Modeling and Optimization in Science and Technologies

Fatos Xhafa
Leonard Barolli
Admir Barolli
Petraq Papajorgji *Editors*

Modeling and Processing for Next-Generation Big-Data Technologies

With Applications and Case Studies

Springer

Modeling and Optimization in Science and Technologies

Volume 4

About this Series

The book series *Modeling and Optimization in Science and Technologies (MOST)* publishes basic principles as well as novel theories and methods in the fast-evolving field of modeling and optimization. Topics of interest include, but are not limited to: methods for analysis, design and control of complex systems, networks and machines; methods for analysis, visualization and management of large data sets; use of supercomputers for modeling complex systems; digital signal processing; molecular modeling; and tools and software solutions for different scientific and technological purposes. Special emphasis is given to publications discussing novel theories and practical solutions that, by overcoming the limitations of traditional methods, may successfully address modern scientific challenges, thus promoting scientific and technological progress. The series publishes monographs, contributed volumes and conference proceedings, as well as advanced textbooks. The main targets of the series are graduate students, researchers and professionals working at the forefront of their fields.

More information about this series at http://www.springer.com/series/10577

Fatos Xhafa · Leonard Barolli
Admir Barolli · Petraq Papajorgji
Editors

Modeling and Processing for Next-Generation Big-Data Technologies

With Applications and Case Studies

Springer

Editors
Fatos Xhafa
Universitat Politècnica de Catalunya
Barcelona
Spain

Admir Barolli
University of Salerno
Salerno
Italy

Leonard Barolli
Fukuoka Institute of Technology (FIT)
Fukuoka
Japan

Petraq Papajorgji
Canadian Institute of Technology
Tirana
Albania

ISSN 2196-7326 ISSN 2196-7334 (electronic)
ISBN 978-3-319-09176-1 ISBN 978-3-319-09177-8 (eBook)
DOI 10.1007/978-3-319-09177-8

Library of Congress Control Number: 2014953522

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface by Editors

Nowadays, we are witnessing an exponential growth in data sets, coined as Big Data era. The data being generated in large Internet-based IT systems is becoming a cornerstone for cyber-physical systems, administration, enterprises, businesses, academia and all human activity fields. Indeed, there is data being generated everywhere: in IT systems, biology, genomics, financial, geospatial, social networks, transportation, logistics, telecommunications, engineering, digital content, to name a few. Unlike recent past where the focus of IT systems was on functional requirements and services, now data is seen as a new asset and data technologies are needed to support IT systems with knowledge, analytics and decision support systems.

Researchers and developers are facing challenges in dealing with this data deluge. Challenges arise due to the extremely large volumes of data, their heterogenous nature (structured & unstructured) and the pace at which data is generated requiring both offline and online processing of large streams of data as well as storing, security, anonymicity, etc. Obviously, most traditional database solutions may not be able to cope with such challenges and non-traditional database and storage solutions are imperative today. Novel modelling, algorithms, software solutions and methodologies to cover full data cycle (from data gathering to visualisation and interaction) are in need for investigation.

This Springer book brings together nineteen chapter contributions on new models and analytic approaches for the modelling of large data sets, efficient data processing (online/offline) and analysis (analytics, mining, etc.) to enable next generation data aware systems offering quality content and innovative services in a reliable and scalable way. The book chapters critically analyze the state of the art and envision the road ahead on modelling, analysis and optimisation models for next generatation big data technologies. Finally, benchmarking, frameworks, applications, case studies and best practices for big data are also included in the book.

Main contributions of this book

Specifically, the contributions of this book focus on the following research topics and development areas of big data:

Data Modelling and Analysis

Data modelling is among the foremost building blocs in today's data technologies. Because data comes in a variety of formats and from various sources and could be structured, semi-structured or unstructured, the mission of the data modelling is to bring structure to the data in a transparent way for efficient storage and further processing by the applications and systems consuming the data. These issues are dealt with in three first chapters of the book:

[1] Rodolfo da Silva Villaça, Rafael Pasquini, Luciano Bernardes de Paula and Maurício Ferreira Magalhães. Exploring the Hamming distance in distributed infrastructures for similarity search.

[2] Radu-Ioan Ciobanu, Ciprian Dobre, and Fatos Xhafa. Data Modeling for Socially-Based Routing in Opportunistic Networks.

[3] Petra Perner. Decision Tree Induction Methods and Their Application to Big Data.

Specifically, in the chapter by **Villaça *et al***. is presented a vector space model for structuring data and support efficient similarity search techniques. In the second chapter, **Dobre *et al.*** deal with data modelling from the emerging paradigm of opportunistic networks aiming to support efficient data routing and dissemination algorithms. The presented model can found application to many other scenarios from information-centric networks to the Internet of Things. **Perner**, in the third chapter, introduces the decision tree induction model and its use for data mining/learning algorithms. The author then gives an outlook on the application of decision tree induction to Big Data scenarios.

Data Gathering, Aggregation and Replication

The need for big data solutions has led to the definition of a whole data life cycle. In such cycle, data gathering, aggregation, structuring, storing and replication are fundamentals steps. Indeed, unlike traditional approaches when most data is stored into databases in a straighforward way, big data sources emerge from various wireless and mobile networked devices, known as "world sensing data". Additionally, not all data is or should be stored as is, but can be classified and grouped/gathered into larger data entities or objects. In many such scenarios, data is also replicated to increase availability and reliability. Several approaches to such issues are presented in the next three chapters, as follows:

[4] Suchetana Chakraborty, Sandip Chakraborty, Sukumar Nandi, and Sushanta Karmakar. Sensory Data Gathering for Road-Traffic Monitoring: Energy Efficiency, Reliability and Fault-tolerance

[5] Kazuya Matsuo, Keisuke Goto, Akimitsu Kanzaki, Takahiro Hara, Shojiro Nishio. Data aggregation and forwarding route control for efficient data gathering in dense mobile wireless sensor networks

[6] Evjola Spaho, Admir Barolli, Fatos Xhafa, and Leonard Barolli. P2P Data Replication: Techniques and Applications

Respectively, **Chakraborty *et al.*** present a novel tree based data gathering scheme for data gathering in Vehicular Sensor Networks. The presented approach enables an efficient design of data collection protocol through which delay sensitivity and reliability of the large volume of application data as well as the scarcity of sensor resources can be addressed. **Matsuo *et al.*** present a data gathering method considering geographical distribution of data values for reducing traffic in dense mobile wireless sensor networks. Data replication and the usefulness of P2P techniques to increase availability and reliability are presented in the chapter by **Spaho *et al.***

High Performance and MapReduce Processing

Efficient data processing has become a must for big data approaches to meet the timely processing needs (both online/real-time and offline), due to large volumes of data and the continuous increasing amounts of data. Two chapters cover the challenges and present solutions to achieve high performance processing:

[7] Alexey Cheptsov, Bastian Koller. Leveraging High Performance Computing Infrastructures to Web Data Analytic Applications by means of Message-Passing Interface.

[8] Jia-Chun Lin, Fang-Yie Leu, Ying-ping Chen. ReHRS: A Hybrid Redundant System for Improving MapReduce Reliability and Availability.

Chepstov and Koller present an approach to parallelise data-centric applications based on the known Message-Passing Interface. Although MPI is a long standing parallel computing framework, the authors have introduced novel features to achieve high utilisation rate and low costs of using productional high performance computing and Cloud computing infrastructures. They also discuss on OMPIJava –Java bindings for Open MPI—as an alternative to MapReduce Hadoop framework. On the other hand, **Lin *et al.*** make an indepth analysis of MapReduce framework and present techniques to improve the availability and reliability to support robust data processing applications.

Data Analytics and Visualisation

With the emergence of the Big Data, data analysis and data analytics are taking great importance as means to support knowledge extraction and decision support systems. Under the name of data analytics are comprised various specific forms of analytics such as business analytics, health analytics, learning analytics, security analytics, etc. On the other hand, closely related to analytics is the visualisation, which becomes challenging due to the size and multi-dimensionality of the data. The three next chapters bring issues, challenges and appoaches to data analytics and data visualisation.

[9] Patricia Morreale, Allan Goncalves, and Carlos Silva. Analysis and Visualization of Large Scale Time Series Network Data

[10] Yoshihiro Okada. Parallel Coordinates Version of Time-tunnel (PCTT) and Its Combinatorial Use for Macro to Micro Level Visual Analytics of Multidimensional Data

[11] Martin Strohbach, Holger Ziekow, Vangelis Gazis, Navot Akiva. Towards a Big Data Analytics Framework for IoT and Smart City Applications

Concretely, **Morreale *et al.*** present a methodology for data cleaning and preparation to support big data analysis along with a comparative examination of three widely available data mining tools. The proposed methodology is used for analysis and visualisation of a large scale time series dataset of environmental data. The research issues related to visualisation of multidimensional data is studied in the chapter by **Okada**, where the author introduces an interactive visual analysis tool for multidimensional data and multi-attributes data. **Strohbach *et al.*** analyse the high level requirements of big data analytics and then provide a Big Data Analytics Framework for IoT and their application to smart city. Their approach is exemplified through a case study in the smart grid domain. A prototype of the framework addressing the volume and velocity challenges is also presented.

Big Data, Cloud Computing and Auditing

Big Data and Cloud computing are penetrating and impacting each time more the businesses and enterprises in various forms, including business intelligence, decision making, business processes, innovation, etc. However, the adoption of these new technologies by businesses and enterprises is facing many challenges. The two chapters below analyse such challenges in the interdisciplinary field of ICT, businesses, innovation auditing and control.

[12] Antonio Scarfò and Francesco Palmieri. How the big data is leading the evolution of ICT technologies and processes

[13] David Simms. Big Data, Unstructured Data and the Cloud: Perspectives on Internal Controls

Scarfò and Palmieri highlight the most important innovation and development trends in the new arising scenarios of Big Data and its impact on the organisation of ICT-related companies and enterprises. The authors make a critical analysis and address the missing links in the ICT big Picture and present the emerging data-driven reference models for the modern information-empowered society. **Simms** analyses the increasing awareness by businesses and enterprises about the value of Big Data to the world of corporate information systems. Through his analysis it is shown nevertheless that inspite of the potential advantages brought by Big Data and Cloud computing, like the use of outsourcing, their adoption requires addressing issues of confidentiality, integrity and availability of applications and data. An appropriate understanding of risk and control issues is advocated in the chapter as a need for a successful adoption of these new technologies.

Big Data, Mobile Computing and IoT

With the fast increase in the number of smartphones, tablets as well as all sorts of smart devices, the mobile computing and Internet of Things have fast become among most important Big Data sources. It is expected that processing of such data will significantly contribute to smart applications and environments, with a human-centric focus. The fol-

lowing two chapters address the challenges to be faced for achieving user-centric aware IoT that brings together people and devices into a sustainable eco-system.

[14] María V. Moreno-Cano, José Santa, Miguel A. Zamora-Izquierdo, and Antonio F. Skarmeta. Future Human-Centric Smart Environments

[15] Tor-Morten Grønli, Gheorghita Ghinea, Muhammad Younas, Jarle Hansen. Automatic Configuration of Mobile Applications using Context-Aware Cloud Based Services

Moreno-Cano *et al.* bring a user-centric perspective of IoT and present a management platform for smart environments. Their platform is based on a layered architecture and uses artificial intelligent methods to transform raw data into semantically-meaningful information used by services. Their approach is exemplified with real use cases from smart buildings. **Grønli *et al.*** discuss several challenges in the area of context-awareness in the cloud setup, whereby context-aware information is harvested from several dimensions to build a rich foundation for context-aware computation. The authors have combined and exploited the Cloud and Mobile computing paradigms to create a new user experience and a new way to invoke control over user's mobile phone.

Social Networking and Crowd-sourcing

Social networking is yet another important source for Big Data, referred to also as Social Big Data, Social Data Sensing and Crowd-sourcing Big Data, which consist of user activity data collected by social networks and crowd-sourced, participatory activities. As the number of users of social networks keeps increasing, pushed by advances in mobile computing and smartphones, this kind of big data is seen as a real asset that could bring value to companies such as by providing users with personalized content. Nevertheless, extracting the real value from such data is challenging due to the data volume and growth and the various data formats, which make the data not ready for processing straightway. The two chapters below discuss these issues.

[16] Ryoichi Shinkuma, Yasuharu Sawada, Yusuke Omori, Kazuhiro Yamaguchi, Hiroyuki Kasai, Tatsuro Takahashi. Socialized system for enabling to extract potential 'values' from natural and social sensing data

[17] G. Piro, V. Ciancaglini, R. Loti, L.A. Grieco, and L. Liquori. Providing crowd-sourced and real-time media services through a NDN-based platform

The chapter by **Shinkuma *et al***. mainly considers two problems faced when extracting values from sensing data, namely, dealing with raw/unprocessed sensing data and the inefficiency in terms of management costs to keep all sensing data usable. The authors propose a relational graph-based approach to encode the characteristics of sensing data. On the other hand, **Piro *et al***. use crowd-sourcing for providing real-time media contents. To that aim, the design of a network architecture, based on the emerging Named Data Networking is used to support crowd-sourced real-time media contents.

Open Data, Benchmarking, Frameworks, Best Practices and Experiences

As in the case of other emerging research fields, Big Data calls for benchmarking, standardisation and evaluation frameworks. The book includes two chapters bringing a set of best practices and experiences in open data projects and frameworks addressing the needs and challenges in this regard.

[18] Mikel Emaldi, Oscar Peña, Jon Lázaro, Diego López-de-Ipiñaa. Linked Open Data for Smarter Cities

[19] Franck Le Gall, Sophie Vallet Chevillard, Alex Gluhak, Nils Walravens, Zhang Xueli, Hend Ben Hadji. Benchmarking Internet of Things Deployment: Frameworks, Best Practices and Experiences

The chapter by **Emaldi *et al***. proposes the use of Linked Open Data together with a set of best practices to publish data on the Web recommended by the W3C, in a new data life cycle management model. Their approach is exempflied for the case of open data for smart cities, namely, smart data, enabling automatic consumption of big amounts of data, providing relevant and high quality data to end users with low maintenance costs. Finally, **Le Gall *et al***. critically analyse existing gaps in assessing the utility and benefits of IoT deployments and propose a novel benchmarking framework for IoT deployments. The proposed framework is complementary to the emerging tools for the analysis of Big Data and allows a better decision making for policy makers for regulatory frameworks.

Emerging Applications

Altogether, the chapters of the book bring a variety of big data applications from *IoT systems, smart cities, traffic control, energy efficient systems, disaster management, etc.* sheding light on the great potential of Big Data but also envisioning the road ahead in this exciting field of the data science.

Targeted Audience and Last Words

The contributions of the chapters of the book are researchers and practitioners from academia and industry, who bring their expertise and experience in the Big Data, comprising fundamental approaches, implementations and experimental approaches as well as benchmarking and best practices. The variety of approaches, examples and applications along the chapters makes the book interesting to ample audiences of academics, instructors and senior students, researchers and senior graduates from academia, networking and software engineers, data analysts, business analysts from industries and businesses.

We hope that the readers and practitioners of Big Data will find this book useful in their academic, research and professional activities!

Acknowledgements

The editors of this book wish to sincerely thank all the authors of the chapters for their interesting contributions to this Springer volume, for taking aboard all comments and feedback from editors and reviewers and for their timely efforts to provide high quality chapter manuscripts. We are grateful to the reviewers of the chapters for their generous time and for giving useful suggestions and constructive feedback to the authors. We would like to acknowledge the encouragement received from Prof. Srikanta Patnaik, the editor in chief of the Springer series *"Modeling and Optimization in Science & Technology"* and the support from Dr. Leontina Di Cecco, Springer Editor, and the whole Springer's editorial team during the preparation of this book.

Finally, we wish to express our gratitude to our families for their understanding and support during this book project.

February 2014

The Editors

Fatos Xhafa
Leonard Barolli
Admir Barolli
Petraq Papajorgji

Contents

List of Contributors

Luigi Alfredo Grieco
DEI - Politecnico di Bari, Italy
a.griecog@poliba.it

Navot Akiva
AGT International, Germany
nakiva@agtinternational.com

Admir Barolli
University of Salerno, Italy
admir.barolli@gmail.com

Leonard Barolli
Fukuoka Institute of Technology, Japan
barolli@fit.ac.jp

Hend Ben Hadji
Centre d'Etudes et des Recherches des Télécommunications, Tunisie
hend.benhji@cert.mincom.tn

Luciano Bernardes de Paula
Federal Institute of Education, Science and Technology of São Paulo, Brazil
lbernardes@ifsp.edu.br

Suchetana Chakraborty
Indian Institute of Technology, India
suchetana@iitg.ernet.in

Sandip Chakraborty
Indian Institute of Technology, India
c.sandip@iitg.ernet.in

Ying-ping Chen
National Chiao Tung University,
Taiwan
ypchen@cs.nctu.edu.tw

Alexey Cheptsov
High Performance Computing Center Stuttgart,
Germany
cheptsov@hlrs.de

Vincenzo Ciancaglini
INRIA - Sophia Antipolis, France
fvincenzo.ciancaglini@inria.fr

Jia-Chun Lin
National Chiao Tung University,
Taiwan
kellylin1219@gmail.com

Radu-Ioan Ciobanu
University Politehnica of Bucharest,
Romania
radu.ciobanu@cti.pub.ro

Rodolfo da Silva Villaça
Federal University of Espírito
Santo, Brazil
rodolfo.villaca@ufes.br

Ciprian Dobre
University Politehnica of Bucharest,
Romania
ciprian.dobre@cs.pub.ro

Mikel Emaldi
University of Deusto, Spain
fm.emaldi@deusto.es

Maurício Ferreira Magalhães
State University of Campinas, Brazil
mauricio@dca.fee.unicamp.br

Vangelis Gazis
AGT International, Germany
vgazis@agtinternational.com

Gheorghita Ghinea
Brunel University, UK
george.ghinea@brunel.ac.uk

Alex Gluhak
University of Surrey, UK
a.gluhak@surrey.ac.uk

Allan Goncalves
Kean University, Union, NJ USA
goncalal@kean.edu

Keisuke Goto
Osaka University, Japan
goto.keisuke@ist.osaka-u.ac.jp

Tor-Morten Grønli
Norwegian School of IT, Norway
tmg@nith.no

Jarle Hansen
Systek AS, Norway
jarle@jarlehansen.net

Takahiro Hara
Osaka University, Japan
hara@ist.osaka-u.ac.jp

Akimitsu Kanzaki
Osaka University, Japan
kanzaki@ist.osaka-u.ac.jp

Sushanta Karmakar
Indian Institute of Technology, India
sushanatak@iitg.ernet.in

Hiroyuki Kasai
The University of Electro-
communications, Japan
kasai@is.uec.ac.jp

Bastian Koller
High Performance Computing Center
Stuttgart, Germany
koller@hlrs.de

Jon Lázaro
University of Deusto, Spain
jlazaro@deusto.es

Franck Le Gall
Easy Global Market, France
franck.le-gall@eglobalmark.com

Fang-Yie Leu
TungHai University, Taiwan
leufy@thu.edu.tw

Luigi Liquori
INRIA - Sophia Antipolis, France
luigi.liquorig@inria.fr

Diego López-de-Ipiña
University of Deusto, Spain
dipinag@deusto.es

Riccardo Loti
INRIA - Sophia Antipolis, France
riccardo.loti@inria.fr

Kazuya Matsuo
Osaka University, Japan
matsuo.kazuya@ist.osaka-u.ac.jp

María V. Moreno-Cano
University of Murcia, Spain
mvmoreno@um.es

Patricia Morreale
Kean University, Union, NJ USA
pmorreal@kean.edu

Sukumar Nandi
Indian Institute of Technology, India
sukumar@iitg.ernet.in

Shojiro Nishio
Osaka University, Japan
nishio@ist.osaka-u.ac.jp

Yoshihiro Okada
Kyushu University, Japan
okada@inf.kyushu-u.ac.jp

Yusuke Omori
Kyoto University, Japan
omori@cube.kuee.kyoto-u.ac.jp

Francesco Palmieri
Second University of Naples, Italy
francesco.palmieri@unina.it

Rafael Pasquini
Federal University of Uberlândia, Brazil
pasquini@facom.ufu.br

Oscar Peña
University of Deusto, Spain
oscar.pena@deusto.es

Petra Perner
Institute of Computer Vision and applied Computer Sciences, Germany
pperner@ibai-institut.de

Giuseppe Piro
DEI - Politecnico di Bari, Italy
fg.piro@poliba.it

José Santa
University of Murcia, Spain
josesanta@um.es

Antonio Scarfò
MaticMind SpA, Italy,
ascarfo@maticmind.it

Yasuharu Sawada
Kyoto University, Japan
sawada@cube.kuee.kyoto-u.ac.jp

Ryoichi Shinkuma
Kyoto University, Japan
shinkuma@i.kyoto-u.ac.jp

Carlos Silva
Kean University, Union, NJ USA
salvadca@kean.edu

David Simms
Haute Ecole de Commerce, University of Lausanne, Switzerland
david.simms@unil.ch

Antonio F. Skarmeta
University of Murcia, Spain
skarmeta@um.es

Evjola Spaho
Fukuoka Institute of Technology, Japan
evjolaspaho@hotmail.com

Martin Strohbach
AGT International, Germany
mstrohbach@agtinternational.com

Tatsuro Takahashi
Kyoto University, Japan
ttakahashi@i.kyoto-u.ac.jp

Sophie Vallet Chevillard
inno TSD, France
s.valletchevillard@inno-group.com

Nils Walravens
Vrije Universiteit Brussel, Belgium
nils.walravens@vub.ac.be

Fatos Xhafa
Technical University of Catalonia, Spain
fatos@lsi.upc.edu

Zhang Xueli
Chinese Academy of Telecom Research, China
zhangxueli@catr.cn

Kazuhiro Yamaguchi
Kobe Digital Labo, Inc, Japan
k-yamaguchi@kdl.co.jp

Muhammad Younas
Oxford Brooks University, UK
m.younas@brookes.ac.uk

Miguel A. Zamora-Izquierdo
University of Murcia, Spain
mzamora@um.es

Holger Ziekow
AGT International, Germany
hziekow@agtinternational.com

Exploring the Hamming Distance in Distributed Infrastructures for Similarity Search

Rodolfo da Silva Villaça[1], Rafael Pasquini[2], Luciano Bernardes de Paula[3], and Maurício Ferreira Magalhães[4]

[1] Department of Computing and Electronics (DCEL),
Federal University of Espírito Santo (UFES),
São Mateus/ES, Brazil
rodolfo.villaca@ufes.br
[2] Faculty of Computing (FACOM),
Federal University of Uberlândia (UFU),
Uberlândia/MG, Brazil
pasquini@facom.ufu.br
[3] Federal Institute of Education, Science and Technology of São Paulo (IFSP),
Bragança Paulista/SP, Brazil
lbernardes@ifsp.edu.br
[4] School of Computing and Electrical Engineering (FEEC),
State University of Campinas (UNICAMP),
Campinas/SP, Brazil
mauricio@dca.fee.unicamp.br

Abstract. Nowadays, the amount of data available on the Internet is over Zettabytes (ZB). Such condition defines a scenario known in the literature as Big Data. Although traditional databases are very efficient for finding and retrieving specific content, they are inefficient on Big Data scenario, since the great majority of such data are unstructured and scattered across the Internet. In this way, new databases are required in order to support similarity search. In order to handle such challenging scenario, the proposal in this chapter is to explore the Hamming similarity existent between content identifiers that are generated using the Random Hyperplane Hashing function. Such identifiers provide the basis for building distributed infrastructures that facilitate the similarity search. In this chapter, we present two different approaches: a P2P solution (Hamming DHT) and a Data Center solution (HCube). Evaluations are presented and indicate that both are capable of improving the recall in a similarity search.

1 Introduction

In the current Big Data scenario, users have become data sources; companies store uncountable information from clients; millions of sensors monitor the real world, creating and exchanging data in the Internet of things. According to a study from the International Data Corporation (IDC) published in May 2010 [1], the amount of data available in the Internet surpassed 2 ZB in 2010, doubling every 2 years, and might surpass 8 ZB in 2015. The study also revealed that approximately 90% of them are composed

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_1

of unstructured, heterogeneous, and variable data in nature, such as texts, images, and videos.

Emerging technologies, such as Hadoop [2] and MapReduce [3], are examples of solutions designed to address the challenges imposed by Big Data in the so-called three Vs: Volume, Variety, and Velocity. Through parallel computing techniques in conjunction with grid computing or, recently, taking advantage of the infrastructure offered by the cloud computing concept, IT organizations offer means for handling large-scale, distributed, and data-intensive jobs. Usually, such technologies offer a distributed file system and automated tools for adjusting, on the fly, the number of servers involved in the processing tasks. In such case, large volumes of data are pushed over the networking facility connecting the servers, transferring $< key, value >$ pairs from mappers to reducers in order to obtain the desirable results. In this scenario it is desirable to minimize the need for moving data across the network in order to speedup the overall processing task.

While the current solutions are unquestionably efficient for handling traditional applications, such as batch processing of large volumes of data, they do not offer adequate support for the similarity search [4], whose objective is the retrieval of sets of similar data given by a similarity level. As an example, similarity between data may be used in a recommender system based on users' social profiles. In an example, a user profile can be defined as a set of characteristics that uniquely influence how users make their decisions. Users with similar characteristics are more likely to have similar interests and preferences.

In this way, to make a similarity search system based on users' profiles, the characteristics of a user in a social network can be placed in a vector and, using a Vector Space Model (VSM), the similarity between users can be measured through the use of vector distance metrics, such as Euclidean distance, cosine, and Hamming distance. But, except for the Hamming distance, all the other metrics are affected by the curse of dimensionality [4]. The high computational cost due to the dimensionality problem is a challenge to be faced by the new similarity search systems in a Big Data scenario.

This chapter presents how to support similarity search using the Hamming distance as similarity metric. To achieve that, data are indexed in a database using the Locality Sensitive Hashing (LSH) function called Random Hyperplane Hashing (RHH) function [5]. RHH is a family of LSH functions that uses the cosine similarity between vectors and Hamming as the distance metric between the generated binary strings, i.e., the greater the cosine similarity between a pair of content vectors, the lower the Hamming distance between the binary strings. These binary strings represent a data identifier whose similarity can be measured using the Hamming distance. Each query in this database is evaluated through the use of the Hamming distance between the query identifier and each data identifier.

In the similarity search, a query to be evaluated is composed by the same set of characteristics of content indexed in the database. Each user of the similarity search system enters the desired characteristics and a similarity level according to the desired volume of answers to the query. The greater the similarity level, the lower the number of data retrieved because the more specific is the query. The query is indexed in the

database using the RHH function and the Hamming similarity between the query and all data identifiers in the database is calculated. All those profiles whose Hamming similarity satisfies the desired similarity level are returned as the query response.

To evaluate the similarity search system, the following tests were done: the correlation of the cosine similarity between content vectors and the Hamming similarity of their identifiers is presented using four different similarity levels (0.7, 0.8, 0.9, and 0.95), the frequency distribution of the Hamming distance between content identifiers according to their similarity level; and, finally, some results of selected queries, and responses are presented. In our experiments, content vectors represent users profile in the Adult Data Set of the UCI Repository [6].

In our previous work [7] an overlay solution for this similarity search system was developed on top of a Distributed Hash Table (DHT) structure. Essentially, it was shown the possibility of storing similar data in servers close to the logical space of the overlay network by using a $put(k,v)$ primitive, and that it is also possible to efficiently recover a set of similar data by using a single $get(k,sim)$ primitive. In another previous work [8] the HCube was shown, a Data Center solution designed to support similarity searches in Big Data scenarios, aiming to reduce the distance to recover similar contents in a similarity search. In HCube similar data are stored in the same hosting server or in servers nearly located in the Data Center.

This chapter is organized as follows: Section 2 presents some background on the technologies used in the Hamming DHT and the HCube. Section 3 contains a literature review on related work on similarity search in Peer-to-Peer (P2P) networks and Data Center. Section 4 briefly presents the Hamming DHT and HCube solutions. Section 5 evaluates the proposed similarity search system in distributed scenarios. Section 6 provides some final remarks and future work.

2 Background

This section presents the concept of the VSM, a model to represent data as vectors in a multidimensional space; the RHH Function, an LSH function used to generate data identifiers preserving similarity between content vectors; and the Hamming similarity function, a similarity function that is used to compare the Hamming distance between binary identifiers.

2.1 Vector Space Model

VSM is an algebraic model for representing objects as vectors. In general, each dimension of these vectors is related to a characteristic of the content itself, such as keywords in a text, color histogram in a picture, or profile attributes in a social network.

A set of adult vectors, extracted from the Adult Data Set of the UCI Repository [6], is used to describe the procedure of transforming such attributes in a vector that can be measured and compared to other vectors using algebraic operations. In essence, this data set contains information about adult citizens living in the US including the following attributes: age, work-class, education level, years in school, marital status, occupation, relationship, race, sex, capital gain and loss over the last year, hours worked per week, native country, and annual salary. Samples of theses profiles are shown in the sequence:

- ADULT1 - 43; Self-emp-not-inc; 5th-6th; 3; Married-civ-spouse; Craft-repair; Husband; White; Male; 0; 4700; 20; United-States; $\leq$50K
- ADULT2 - 56; Private; 10th; 6; Married-civ-spouse; Craft-repair; Husband; White; Male; 0; 0; 0.45; France; $\leq$50K
- ADULT3 - 50; Self-emp-inc; Prof-school; 15; Married-civ-spouse; Prof-specialty; Husband; White; Male; 0; 0; 36; United-States; $\geq$50K
- ADULT4 - 30; Private; Prof-school; 15; Married-civ-spouse; Prof-specialty; Husband; White; Male; 0; 0; 30; United-States; $\geq$50K

For the experiments, some adaptations must be done in the adult vectors. Numerical attributes present in the vectors, such as "age", had to be normalized to the range [0..1]. Such normalization was done by dividing the value by the highest one in the set. Coordinates that represent discrete attributes (for example, "sex" that may be "male" or "female") were divided into different ones, each one in a separated dimension corresponding to the possible values. For the "sex" attribute, two dimensions were created: "male" and "female". If the person is a man, his vector has a value "1" for the "male" dimension and "0" for the female dimension and vice versa in case of a woman being represented.

As stated in [9], this procedure was necessary because the notion of similarity or distance for discrete informations is not as straightforward as for the numerical ones and was a major challenge faced here. This is due to the fact that different values taken by a discrete attribute are not inherently ordered and hence a notion of ordering between them is not possible. Also, the notion of similarity can differ depending on the particular domain. Due to this fact, each attribute of a vector had to be extended in a number of dimensions equal to all the values it contains. Using this procedure, the adult vector had to be extended from 14 to 103 dimensions.

A possibility to measure the similarity between vectors that represent some data is to calculate the cosine of the angle between them (sim_{cos}). The cosine similarity produces high quality results across several domains as presented in [10]. To illustrate this, Figure 1 presents an application in which user profiles are represented by two-dimensional vectors, each dimension describing the user's interest in Sports and Literature, using a scale in which "0" means no interest and "10" means total interest in a topic. Consider four user profiles represented by the tuples PROFILE1(3,4) - rank 3 for Sports and 4 for Literature, PROFILE2(4,4) - rank 4 for both Sports and Literature, PROFILE3(5,3) - rank 5 for Sports and 3 for Literature, and PROFILE4(7, 3) - rank 7 for Sports and 3 for Literature.

In order to provide insights into the use of similarity by cosine (sim_{cos}), consider the development of a friendship recommender system based on users' profiles responsible for indicating friends with similar interests to a new user profile and represented by the tuple NEW_PROFILE(8,1) - rank 8 for Sports and 1 for Literature. In this case, the recommender system would suggest to NEW_PROFILE the following order of preference for the establishment of new friendships: 1) PROFILE4 whose $sim_{cos} \approx 0.96$, 2) PROFILE2 whose $sim_{cos} \approx 0.9$, 3) PROFILE3 whose $sim_{cos} \approx 0.8$, and 4) PROFILE1 whose $sim_{cos} \approx 0.7$.

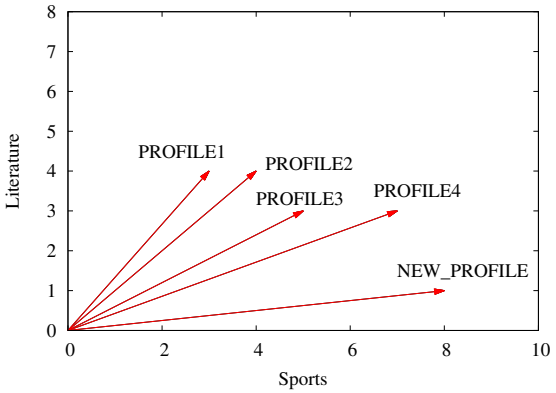


Fig. 1. Graphical representation of profiles vectors

2.2 Random Hyperplane Hashing and Hamming Similarity

The LSH functions reduce the dimensions of vectors representing data while ensuring that the more similar two objects are, the more similar the hash values of their vectors will be [4]. Each family of LSH functions is related to some similarity function. The RHH is an example of a family of LSH functions related to the cosine similarity. In this context, Charikar [5] presents a hashing technique summarized in this section.

Given a set $\overrightarrow{r}_1, \overrightarrow{r}_2, \ldots, \overrightarrow{r}_m$ of m vectors $\overrightarrow{r} \in \Re^d$, each of their coordinates randomly drawn from a standard normal distribution, and a vector $\overrightarrow{u} \in \Re^d$, a hash function $h_{\overrightarrow{r}}$ is defined as follows:

$$h_{\overrightarrow{r}}(\overrightarrow{u}) = \begin{cases} 1, \text{if } \overrightarrow{r} \cdot \overrightarrow{u} \geq 0 \\ 0, \text{if } \overrightarrow{r} \cdot \overrightarrow{u} < 0 \end{cases}$$

For each $h_{\overrightarrow{r}}(\overrightarrow{u})$ one bit is generated, and the results of m $h_{\overrightarrow{r}}(\overrightarrow{u})$ are concatenated to compose an m-bit hash key for this vector $\overrightarrow{u}$. For two data vectors $\overrightarrow{u}, \overrightarrow{v} \in \Re^d$, the probability of generating similar keys is the value of the cosine of the angle between $\overrightarrow{u}$ and $\overrightarrow{v}$. Consequently, the greater the cosine similarity, the more likely the generated keys will share common bits, leading to two identifiers close in the Hamming distance (i.e., the number of different bits in two binary strings). Inversely, the Hamming similarity is possible to be calculated by measuring $sim_h = \frac{m - D_h}{m}$, in which sim_h is the Hamming similarity, D_h is the Hamming distance, and m is the number of bits in the string representing the user profile identifier.

As an example, assuming that an application uses an identifier of 8-bits, a sequence of $m = 8$ random vectors $\overrightarrow{r}$ must be generated, and the returned bits of the m $h_{\overrightarrow{r}}$ are concatenated in order to generate an 8-bit identifier. Table 1 shows an example comparing the cosine similarity (sim_{cos}), Hamming distance (D_h) and similarity (sim_h) between 4 generic 8-bits user profiles identifiers. In this example, "PROFILE B", which is the most similar profile to "PROFILE A" based on the cosine similarity, also has the smallest Hamming distance (D_h) corresponding to a Hamming similarity (sim_h) of 0.875.

Table 1. 8-bits profile identifiers, Hamming distance (D_h), Hamming similarity (sim_h) and Cosine similarity (sim_{cos}) of user profiles

User profiles	8-bits identifier	D_h	sim_h	sim_{cos}
PROFILE B, PROFILE A	01001010, 01101010	1	0.875	0.99
PROFILE C, PROFILE A	01001000, 01101010	2	0.75	0.95
PROFILE D, PROFILE A	01011000 01101010	3	0.625	0.85

3 Literature Review

Among the related works on similarity search available in the literature, most of them are based on some kind of indexing schemes such as hashing functions [4] or Space Filling Curve (SFC) [11]. Both are motivated by the nearest neighbors problem [4], i.e., how to retrieve the most similar data in an indexing space? The adoption of the Hamming similarity property of the RHH function brings us an advantage: it is not necessary the use of the Hilbert SFC to aggregate similar identifiers.

As said in [4], SFCs are affected by the curse of dimensionality [4]. To address the dimensionality problem, Indyk [4] proposed to use LSH functions. These functions may reduce the number of dimensions of a vector creating an identifier for it, represented by a binary string of size m ($m \geq 0$). The distance between two binary strings generated by the application of any LSH function in a pair of content vectors is inversely proportional to the similarity between them. Our proposal of using the Hamming distance to measure the similarity among content mirroring the corresponding similarities among user profiles is, to the best of our knowledge, a new approach in the design of similarity search systems.

In the first implementation of the similarity search prototype it was necessary to search the whole database. We are aware that it is not adequated in the actual Big Data scenario which motivated this proposal. To address this Big Data challenging scenario, we propose two distributed architectures, Hamming DHT [7] and HCube [8], that can serve as infrastructure for similarity search in a Big Data context. The Hamming DHT is a P2P network which exploits the Hamming similarity to facilitate the search and retrieval of similar profiles. The HCube is a data center infrastructure specialized in the similarity search using the Hamming similarity.

The overlay approaches appear as solutions to help managing large volumes of data. In general, these solutions are based on Distributed Hash Tables (DHT), sharing data among peers in an overlay network. According to [12], P2P multidimensional indexing methods emerge as a whole new paradigm over the last few decades. In this scenario, DHT need to be equipped with multidimensional queries and similarity search processing capabilities.

Hycube [13] is an example of DHT that uses Hamming as a distance metric and organizes peers in a unit size hypercube. However, the costs involved in the maintenance of the hypercube under churn and incomplete cubes are greater than the costs in a consistent hashing DHT, such as the proposed Hamming DHT.

pSearch, proposed by Tang et al. [14], is a P2P network that also uses content vectors and cosine similarity, but it differs from the Hamming DHT since it is specialized for similarity search of text documents in a P2P network. Also, pSearch discusses a way to build a distributed term dictionary in order to index text documents.

Bhattacharya [15] uses cosine similarity and LSH functions to propose a framework for similarity search in distributed databases. It extends the $get(k)$ primitive, provided by any DHT implementation, to support a similarity level $get(k, D_h)$ in the Hamming distance D_h from k. This proposal is not specialized in similarity searches, since it is a patch to be applied over any existent DHT to support similarity search.

From this brief survey we can detach that the Hamming DHT is focused in the reduction of the distance (in hops) necessary to recover similar contents and in the increase of the recall in similarity search systems. The use of LSH functions to index similar contents is not new, but the exploration of the Hamming similarity in the organization of content identifiers is an original idea with no similar proposals. This modification makes the Hamming DHT much simpler than other solutions.

When compared to these works, HCube is a distributed solution using techniques such as LSH functions and SFCs to support similarity searches, but HCube is not based on an overlay solution. HCube presents a server-centric data center structure specialized for similarity search, where similar data are stored in servers physically near, recovering such data in a reduced number of hops and with reduced processing requirements.

On the other hand, players such as Google [16] and Amazon [17] developed data centers specialized in storing and processing large volumes of data through MapReduce solution, but none of them consider the similarity search. In this way, HCube opens up a new research field, where applications can benefit from its similarity search structure, avoiding processing-intensive tasks as occur in MapReduce, since similar data are organized in servers nearly located during the storing phase.

In the next section is presented how to perform searches on a distributed infrastructure based on Hamming distance, considering the similarity in their results.

4 Similarity Search Based on Hamming Distance

In the proposed similarity search system, a user issuing a query must select the desired characteristics and a similarity level, in the [0..1] interval, for the query. Then, the content database is searched for the entries that satisfy the filled characteristics. In this first implementation, only complete queries are allowed, with all attributes filled.

Figure 2 shows an example of the main interface used to select the desired characteristics for the search. In the same interface the user inputs a similarity level. As said before, the greater the similarity level, the lower the number of obtained results, because less profiles in the database satisfy the query. The results are put in a list as the output of the similarity search.

Figure 3 summarizes the prototype operation. The user of the proposed similarity search selects all the desired characteristics for a query, as well as the desired similarity level. After selecting the desired characteristics, a content vector is built according to the procedure described in Section 2.1. This vector $\overrightarrow{v}$ is used as input for the RHH function that generates the query hash identifier. The Hamming similarities between $\overrightarrow{v}$ and all

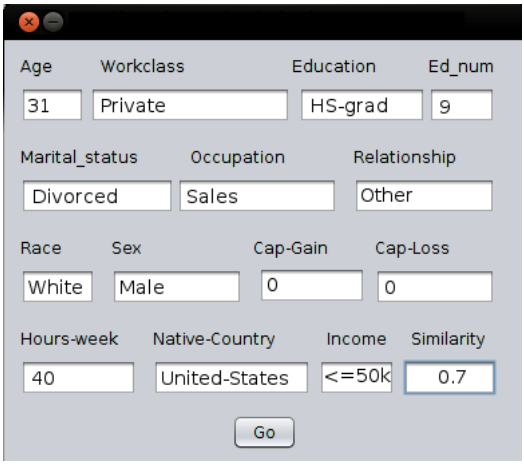


Fig. 2. Prototype interface for a query

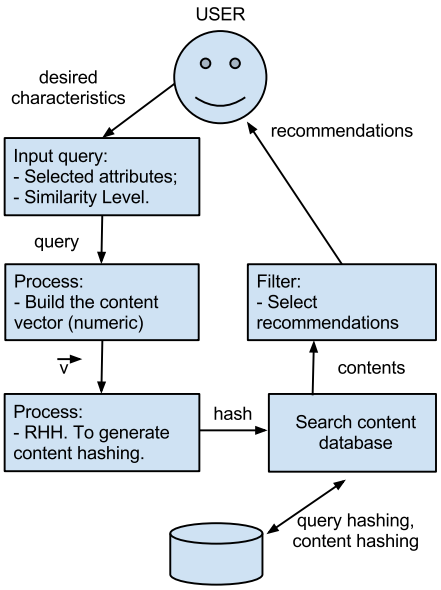


Fig. 3. Prototype operation

entries in the database are calculated, and all entries satisfying the desired similarity level are listed to the user. A second level filter can be implemented to improve the results or to select the most recent ones, as an example, but this filter is not present in this implementation.

Some examples of results, using the adult set described in Section 2.1, are presented and analyzed. The prototype interface was used to get results for all similar profiles

in the database according to a desired similarity level. In the evaluation we use four similarity levels to illustrate the results: 0.7, 0.8, 0.9, and 0.95.

Table 2 presents some of the results. In all cases the input to the search was a profile with the following characteristics: "31" years old; working in the "Private" sector; High School graduates, "HS-grad"; "9" years in school; "Divorced"; working on a "Sales" department; relationship status equal to "Other-relative"; "White"; "Male"; born in the "United-States"; annual income "<=50K" U.S. dollar. It was used a similarity level of 0.7, which means that results greater than or equal to 0.7 were considered.

Table 2. Some results of the prototype, and their Hamming similarity (sim_h), using the same user profile q as input. q: <31; Private; HS-grad; 9; Divorced; Sales; Other-relative; White; Male; United-States; <=50K>.

R_1, $sim_h = 0.75$	<41; Private; HS-grad; 9; Married-civ-spouse; Adm-clerical; Wife; White; Female; United-States; >50K>
R_2, $sim_h = 0.7266$	<46; Private; Some-college; 10; Married-civ-spouse; Sales; Husband; White; Male; United-States; >50K>
R_3, $sim_h = 0.7734$	<36; Private; Some-college; 10; Married-civ-spouse; Craft-repair; Husband; White; Male; United-States; <= 50K>
R_4, $sim_h = 0.8359$	<22; Private; HS-grad; 9; Never-married; Other-service; Other-relative; White; Male; United-States; <=50K>
R_5, $sim_h = 0.8516$	<53; Private; HS-grad; 9; Married-civ-spouse; Craft-repair; Husband; White; Male; United-States; <=50K>
R_6, $sim_h = 0.8438$	<37; Private; HS-grad; 9; Separated; Handlers-cleaners; Not-in-family; White; Male; United-States; <=50K>
R_7, $sim_h = 0.9062$	¡48; Private; Assoc-adm; 12; Never-married; Craft-repair; Not-in-family; White; Male; United-States; <=50K>
R_8, $sim_h = 0.9219$	¡62; Private; HS-grad; 9; Never-married; Craft-repair; Not-in-family; White; Male; United-States; <=50K>
R_9, $sim_h = 0.9141$	<28; Private; HS-grad; 9; Never-married; Craft-repair; Other-relative; White; Male; United-States; <=50K>
R_{10}, $sim_h = 0.9766$	<38; Private; HS-grad; 9; Divorced; Sales; Not-in-family; White; Male; United-States; <=50K>
R_{11}, $sim_h = 0.9688$	<70; Private; HS-grad; 9; Never-married; Craft-repair; Other-relative; White; Male; United-States; <=50K>
R_{12}, $sim_h = 0.9844$	<33; Private; HS-grad; 9; Divorced; Sales; Not-in-family; White; Male; United-States; <=50K>

Not all results are presented in Table 2 but there were 48842 profiles in the database. In the following tests, 48842 queries were done in the same 48842 profiles, leading to 2385540964 results to be evaluated according to the selected similarity level. The total number of results is variable and depends on the size of the database and the similarity level. The greater the similarity level, the shorter is the total number of profiles retrieved.

In the next subsections we present the Hamming DHT and the HCube, two different ways of distributing the proposed similarity search system.

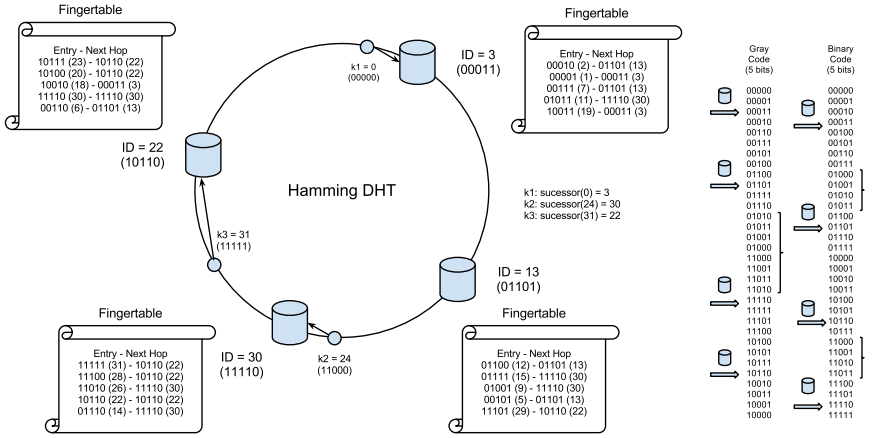


Fig. 4. Example of the Hamming DHT ring with $m = 5$

4.1 Hamming DHT

The Hamming DHT inherits from Chord [18] the consistent hashing approach and the join and leave procedures, but proposes two new features aimed at extracting the maximum benefits from the proposed mechanism for generating similar identifiers for similar contents. In short, the two new features are: 1) the use of Gray codes in the organization of the identifiers in the ring and 2) the establishment of fingers based on the Hamming distance of peers' identifiers.

This section details the storage and retrieval aspects of the proposed system, whose content classification mechanism is developed using the RHH function meeting the following properties:

- $\forall c_1, c_2 \in \mathscr{C} : sim_{cos}(c_1, c_2) \rightarrow [0..1]$, where c_1 and c_2 are content vectors in a content vector space $\mathscr{C}$.
- $\forall c_1, c_2 \in \mathscr{C} : D_h(RHH(c_1), RHH(c_2)) \propto 1/sim_{cos}(c_1, c_2)$.

In essence, as shown in [19], the properties inherent to the RHH functions can represent, with high accuracy level, the similarity between contents measured as their Hamming similarity. Such characteristic, in conjunction with the Gray code organization of the identifiers in the Hamming DHT and the establishment of fingers based on the Hamming distance between peers, provide an efficient system for the similarity search reducing the distance (in hops) between peers storing similar contents.

4.1.1 Building the Ring

All peers composing the virtual ring of the Hamming DHT have an m-bit node identifier. In order to obtain such m-bit identifiers, peers may use any base hash function, like MD5 or SHA-1, applied over, for example, their IP address and/or a private key, assuring the uniqueness of the identifiers.

In the sequence, after obtaining their identifiers, peers join the ring, which is organized according to the Gray code sequence, differently from other DHTs like Chord, in which the ring is organized in a crescent natural order of identifiers. Figure 4 shows an example of the proposed Hamming DHT using $m = 5$. As can be seen in this figure, there are four peers (3 - 00011_2, 13 - 01101_2, 30 - 11110_2, and 22 - 10110_2) and three contents (0 - 00000_2, 24 - 11000_2, and 31 - 11111_2). Observe on the right side of Figure 4 the differences between the Gray code sequence and the crescent natural order of identifiers. From the Gray code sequence we have $3 < 13 < 30 < 22$.

4.1.2 Consistent Hashing

In order to store the contents in the Hamming DHT, the key k of a content is assigned to the first peer whose identifier is subsequent or equal to k, following the Gray code sequence. This peer is called the *successor* of key k and it is denoted as $successor(k)$. In short, the $successor(k)$ is the first peer clockwise from k in the m-bit Gray ring. For example, peer 3 is responsible for storing the content of $k = 0$, the peer 30 is responsible for storing the content of $k = 24$, and the peer 22 is responsible for storing the content of $k = 31$.

In a comparison between the Gray code sequence and the natural binary code (also shown on the right side of Figure 4) it is possible to realize some benefits of the Gray code sequence for aggregating similar contents. As an example, all the occurrences of contents with identifiers *10** are consecutively positioned and stored in peer 30 (11110), while in the binary natural order the occurrences are stored in two different places such as peers 13 (01101) and 30 (11110).

4.1.3 Establishing Fingers

Once the ring is organized, it is possible to store and retrieve information on/from the DHT if each peer has a connection to its successor on the ring. Based on these circular relationships, the actions of storing and retrieving a given key k require that the messages be routed around the Gray ring passing through the list of successor peers until finding the peer responsible for that content key (the $successor(k)$).

However, to obtain a better routing performance, each peer may maintain a routing table, also called finger table, with (at most) m entries. When the required finger entry does not point to a peer in the DHT it is mapped to its successor.

The ith entry (f_i) in the finger table of peer p corresponds to the identifier obtained by switching the ith bit of its identifier. This entry (f_i) maps to the peer $successor(f_i)$ on the Gray code ring, i.e., $f_i = (p \oplus 2^{i-1}) \rightarrow sucessor(f_i), 1 \leq i \leq m$. This finger table also includes the locator (for example, the IP address allocated to this peer) of $successor(f_i)$. As an example, in Figure 4 is shown the finger tables of peers 3, 13, 30, and 22. In this example, the finger table of peer 13 points to the successor of each of its m entries: 12, which points to 13; 15, which points to 30; 9, which points to 30; 5, which points to 13; and 29 which points to 22.

When a peer p does not have a finger directly established with the $successor(k)$ of key k, it forwards the message to the peer p' available in its finger table whose identifier better precedes k in the Gray ring. Such process is repeated until it arrives at the $successor(k)$, corresponding to the situations where the number of hops between peers are bigger than one.

An overall description of the HCube architecture is provided in the next section, exhibiting the organization of servers inside the Data Center structure and defining how the identifiers of the servers are assigned according to the Gray code sequence and presenting the eXclusive OR (XOR) routing responsible for traffic forwarding.

4.2 HCube

As seen in Figure 5, the logical organization of HCube is composed of two layers, the Admission Layer and the Storage Layer.

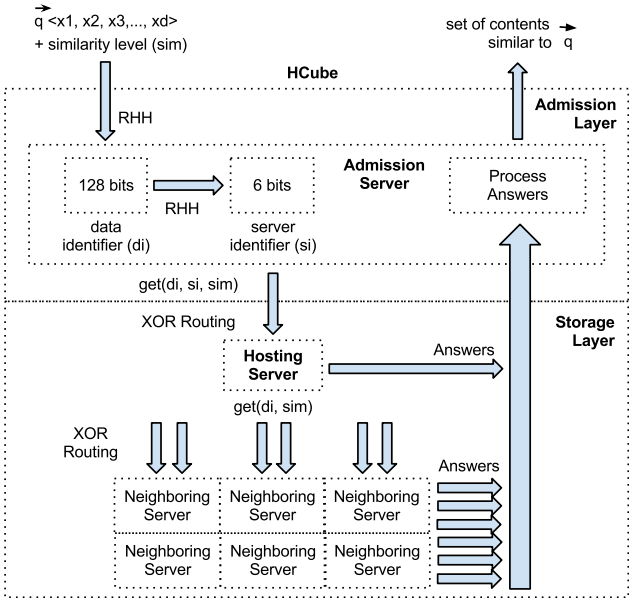


Fig. 5. H-cube operation in a query

The Admission Layer provides the interface between the external world and the HCube structure. This layer is composed of a set of Admission Servers, which operate on top of the HCube, receiving the queries from the users/applications and preparing such queries for being injected in the HCube in order to perform the similarity search. For simplicity, Figure 5 shows only one Admission Server at the Admission Layer.

On the top left corner of Figure 5, it is presented a query vector $\overrightarrow{q}$ composed of d dimensions being admitted by the HCube, in conjunction with the desirable similarity level *sim*. The designation of the Admission Server to be in charge of handling such query may be set according to, for example, the geographic region from where the query is originated.

Once the query vector $\overrightarrow{q}$ is received, the Admission Server obtains the data identifier *di* according to the process described in Section 2.2 which has 128-bits length in

Figure 5. The *di* is the data identifier of reference for the similarity search process. In the sequence, the Admission Server reduces the *di*, also using the RHH function, aimed at obtaining the identifier *si*, which indicates the Hosting Server responsible for the reference data *di*. In this example, the *si* has 6-bits length, resulting in a HCube composed of 64 servers (4x4x4 servers).

After the admission of the query $\overrightarrow{q}$, a *get* composed of the reference data identifier *di*, the hosting server *si*, and the desirable *sim* level is issued in the Storage Layer. Such message is routed toward *si* by using the XOR Routing presented later in Section 4.2.3 and, from the hosting server *si*, a series of $get(di, sim)$ messages is triggered, being forwarded to the neighboring servers also using the XOR mechanism.

The generation of such $get(di, sim)$ messages is driven by some factors, including the reference data *di* and the desirable *sim* level. Specifically for the purposes of evaluations, the $get(di, sim)$ messages are gradually sent to all servers, following a crescent order of Hamming distance between server identifiers, since the objective is to provide a full analysis of the distance between servers storing similar data and the recall in which similar data is recovered.

As the *get* messages are routed inside HCube, the servers containing data within the desirable *sim* level forward them to the Admission Server responsible for handling such request at the Admission Layer. The Admission Server summarizes the answers and delivers the set of similar data to the requesting user/application, concluding the similarity search process. As an alternative implementation, the Admission Server may return a list of references to the similar data, instead of returning the entire data set. Such option avoids unnecessary movement of huge volumes of data and allows the users to choose what piece of data they really want to retrieve, for example, a doctor may open only a few past diagnoses related to the current treatment.

4.2.1 HCube Structure

The HCube is composed of a set of servers organized in a three-dimensional cube topology, in which servers are the fundamental elements in the Data Center (server-centric paradigm). Under such server-centric scheme, simple COTS (commodity off-the-shelf) switches can be used to simplify the wiring process of the Data Center, or direct links can be established among the servers. The most important characteristic of the server-centric scheme is that network elements, like switches, are not involved in the traffic forwarding decisions, they simple provide connections between servers, which are responsible for all the traffic forwarding decisions.

Each HCube server comprises a general purpose processor, with memory, persistent storage, and six NICs (Network Interface Cards) named from `eth0` to `eth5` and distributed in three axes (x, y, and z). In order to be settled in the HCube topology, each server uses the NICs `eth0` and `eth1` in the x axis, the NICs `eth2` and `eth3` in the y axis, and the NICs `eth4` and `eth5` in the z axis.

In order to provide a greater degree of alternative paths in the HCube, increase the fault tolerance and reduce the maximum distance between any pairs of servers, the adopted topology wraps all the links between edge servers, in all the three axes. Figure 6 presents an HCube topology composed of 27 servers. Note in this figure the existence of six NICs in all servers, the linkage in the three axes, and the wrapping connections between the edge servers.

4.2.2 Allocation of Server Identifiers

In order to facilitate the similarity search, the HCube adopts Gray codes for assigning identifiers to all servers composing its structure. By using the Gray codes [20], two successive identifiers of servers differ in only one bit, i.e., neighboring servers present Hamming distance 1 between their identifiers, tending to store similar data.

The proposed organization follows the Gray Space Filling Curve (SFC) which offers a good ratio between the clustering of data and the computing complexity of the curve [11], and it is specialized in the clustering of data according to their Hamming distance. In Figure 7 it is shown, as an example, an HCube with 64 storage servers (identifiers of 6-bits length) distributed in a 4x4x4 HCube. The figure presents the four layers (L1, L2, L3, and L4) of the HCube, which are connected according to the details presented in Section 4.2.1.

As an example, consider the server 29 (011101_2) located in L2 of Figure 7. The six neighbors of this server are 28 (011100_2, L2) and 31 (011111_2, L2) in the x axis; 25 (011001_2, L2) and 21 (010101_2, L2) in the y axis; 13 (001101_2, L1), and 61 (111101_2, L3) in the z axis. Note that all these neighbors present identifiers whose Hamming distance to server 29 is equal to 1.

As another example, in which links between edge servers are established, consider server 60 (111100_2) located in L3 of Figure 7. The six neighbors of this server are 62 (111110_2, L3) and 61 (111101_2, L3) in the x axis; 52 (110100_2, L3) and 56 (111000_2, L3) in the y axis; 44 (101100_2, L4), and 28 (011100_2, L2) in the z axis. All of them also present Hamming distance 1 to the identifier of server 60.

The HCube presented in Figure 7 is a special case, since the number of NICs of the servers matches the length of the identifiers (6-bits). In this particular case, all the neighbors of all servers present Hamming distance equal to 1. However, for HCubes with size bigger than 64 servers the identity space for defining the identifiers of servers must be bigger than 6-bits, resulting in more than 6 servers whose Hamming distance is equal to 1. In short, it means that 6 servers presenting Hamming distance 1 will always

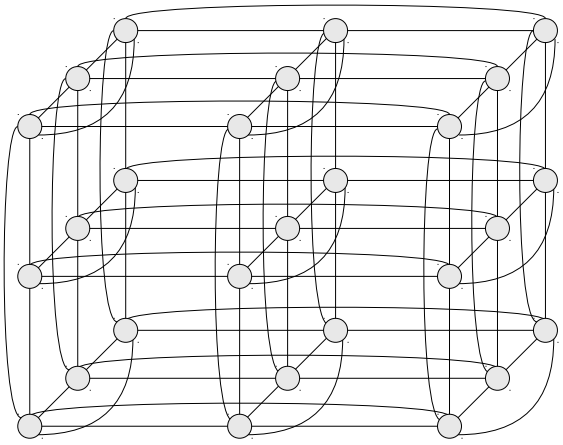


Fig. 6. HCube with 27 servers ($x = 3$, $y = 3$ and $z = 3$)

$$L1 = \begin{bmatrix} 8 & 9 & 11 & 10 \\ 12 & 13 & 15 & 14 \\ 4 & 5 & 7 & 6 \\ 0 & 1 & 3 & 2 \end{bmatrix} \qquad L3 = \begin{bmatrix} 56 & 57 & 59 & 58 \\ 60 & 61 & 63 & 62 \\ 52 & 53 & 55 & 54 \\ 48 & 49 & 51 & 50 \end{bmatrix}$$

$$L2 = \begin{bmatrix} 24 & 25 & 27 & 26 \\ 28 & 29 & 31 & 30 \\ 20 & 21 & 23 & 22 \\ 16 & 17 & 19 & 18 \end{bmatrix} \qquad L4 = \begin{bmatrix} 40 & 41 & 43 & 44 \\ 44 & 45 & 47 & 46 \\ 36 & 37 & 39 & 38 \\ 32 & 33 & 35 & 34 \end{bmatrix}$$

Fig. 7. Visualization of an HCube with 64 servers (L1, L2, L3, and L4)

be directly connected to a given server, and the remaining servers whose Hamming distance is 1 will be placed in a higher number of hops.

Figure 8 exemplifies an HCube bigger than 6-bits, showing the layers an 8x4x4 (7-bits) HCube with 128 servers. Note the Gray SFC represented by the arrows, and consider server 9 (0001001_2) located in the L1 as an example. There are seven Hamming distance 1 servers, organized as follows: servers 8 (0001000_2, L1) and 11 (0001011_2, L1) in the x axis; servers 25 (0011001_2, L1) and 1 (0000001_2, L1) in the y axis; servers 41 (0101001_2, L2) and 73 (1001001_2, L4) in the z axis and, finally, server 13 (0001101_2, L1) which is located in a higher number of hops from server 9. It is important to highlight that the distance in hops to server 13 is reduced given the wrapped links established between edge servers of HCube.

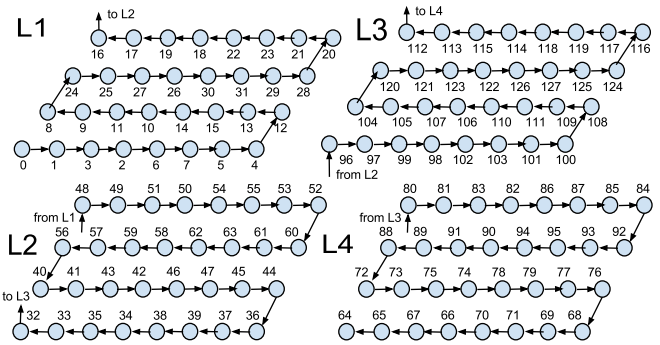


Fig. 8. Gray SFC in a 8x4x4 HCube (128 servers)

A complete Hamming cube must have n dimensions, in which n is the number of bits of the identity space used for servers. Although such structure will have better results in a similarity search using the Hamming similarity, it is difficult to be deployed and far from the real Data Centers wiring structures. In this way, the three-dimensional structure of HCube offers a good trade-off between the complexity of the cube and the recall in a similarity search.

4.2.3 XOR Routing

The XOR metric uses n-bit flat identifiers to organize the routing tables in n columns and route packets through the network. Its routing principle uses the bit-wise exclusive or (XOR) operation between two server identifiers a and b as their distance, which is represented by $d(a,b) = a \oplus b$, being $d(a,a) = 0$ and $d(a,b) > 0, \forall a,b$. Given a packet originated by server x and destined to server z, and denoting $\mathbb{Y}$ as the set of identifiers contained on x's routing table, the XOR-based routing mechanism applied at server x selects a server $y \in \mathbb{Y}$ that minimizes the distance toward z, which is expressed by the following routing policy

$$\mathscr{R} = \underset{y \in \mathbb{Y}}{\operatorname{argmin}}\{d(y,z)\}. \tag{1}$$

In order to support traffic forwarding, the XOR-based routing tables maintained per server are formed by $O(n)$ entries, where the knowledge about neighbor servers is spread into n columns called buckets and represented by $\beta_i, 0 \leq i \leq n-1$. Each time a server a knows a novel neighbor b, it stores the information regarding server b in the bucket β_{n-1-i} given the highest i that satisfies the following condition[1]:

$$d(a,b) \operatorname{div} 2^i = 1, a \neq b, 0 \leq i \leq n-1. \tag{2}$$

In order to exemplify the creation of the routing tables, consider the first example given in Section 4.2.2, assuming server 29 as $a = 011101$ and its neighbor 13 as $b = 001101$. The distance $d(a,b) = 010000$ and the highest i that satisfies the condition (2) is $i = 4$, concluding that the identifier $b = 001101$ must be stored in the bucket $\beta_{n-1-i} = \beta_1$. Basically, condition (2) denotes that server a stores b in the bucket β_{n-1-i}, in which $n-1-i$ is the length of the longest common prefix (*lcp*) existent between both identifiers a and b. This can be observed in Table 3, in which the buckets $\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5$ store the identifiers having *lcp* of length 0, 1, 2, 3, 4, 5 with server 29 (011101).

Table 3. Hypothetical routing table for server 29 (011101) in an HCube with a server identity space in which $n = 6$

β_0	β_1	β_2	β_3	β_4	β_5
111101	**0**01101	**01**0101	**011**001	**0111**11	**01110**0
100000	**0**00000	**01**0000	**011**000	**0111**10	
100010	**0**00100	**01**0100	**011**010		
111111	**0**01111	**01**0111	**011**011		

Such routing tables approach is one of the main advantages of the XOR-based mechanism, since a server only needs to know one neighbor per bucket of the possible 2^n servers available in the network to successfully route packets. If a server has more than one entry per bucket, such additional entries might optimize the routing process, reducing the number of hops in the path from source to destination. Another important

[1] div denotes the integer division operation on integers.

characteristic of this routing table is related to the number of servers that fit in each one of the buckets. There is only one server that fits in the last bucket (β_5), two in β_4, four in β_3, doubling until reaching the first bucket (β_0), where 50% of all servers fit in such buckets (32 servers for $n = 6$). For simplicity of presentation, Table 3 shows examples of neighbor servers limited to four lines.

Afterwards, assuming the Gray code distribution of server identifiers adopted in HCube, filling the buckets is easier, since each one of the Hamming distance 1 neighbors fit in exactly one bucket of the routing table, assuring all the traffic forwarding inside the HCube. Note in the first line of Table 3 the intentional presence of all servers whose identifiers present Hamming distance 1 to server 29 (011101): server 111101 in β_0, 001101 in β_1, 010101 in β_2, 011001 in β_3, 011111 in β_4, and 011100 in β_5. As mentioned before, for HCubes bigger than 64 servers (6-bits), only 6 servers with Hamming distance 1 will be physically connected to a given server. In this case, a signaling process [21] is used to discover the other servers located at distances bigger than 1 hop.

In the next section is presented some evaluations of the proposed similarity search system in distributed environments.

5 Evaluations

In the first place, the correlation between Hamming and cosine similarity was evaluated, showing that it is possible to use the RHH function to generate content identifiers preserving its similarity in the Hamming distance between them.

After that, some evaluations regarding the Hamming DHT and HCube are presented. More results can be found in [7] and [8]. These evaluations show that it is possible to reduce the distance between similar data in distributed environments using both proposals.

5.1 Hamming Similarity

This section evaluates the correlation between the cosine similarity of the Adult profiles used in the tests and their corresponding Hamming similarity. To perform the tests, adult profiles were indexed using RHH resulting in a 128-bits binary identifier. These identifiers were used to index the adult profiles in the database. Each profile $\overrightarrow{q}$ in the Adult set was used as a query vector in the same database. The cosine and Hamming similarity between $\overrightarrow{q}$ and all other profiles were calculated and selected those presenting a cosine similarity equal greater than or equal to 0.7.

Figure 9 plots the correlation between cosine and Hamming similarity in the tests using similarity levels 0.7, 0.8, 0.9, and 0.95. From Figure 9(a) it is possible to notice that, for a cosine similarity greater than or equal to 0.7, the corresponding Hamming similarity varies in the range $[0.62..0.85]$. From Figure 9 it is possible to notice that the correlation is improved in the highest similarity levels. For example, for a cosine similarity of 0.95 (Figure 9(d), the corresponding Hamming similarity varies in the approximate range $[0.9..1]$.

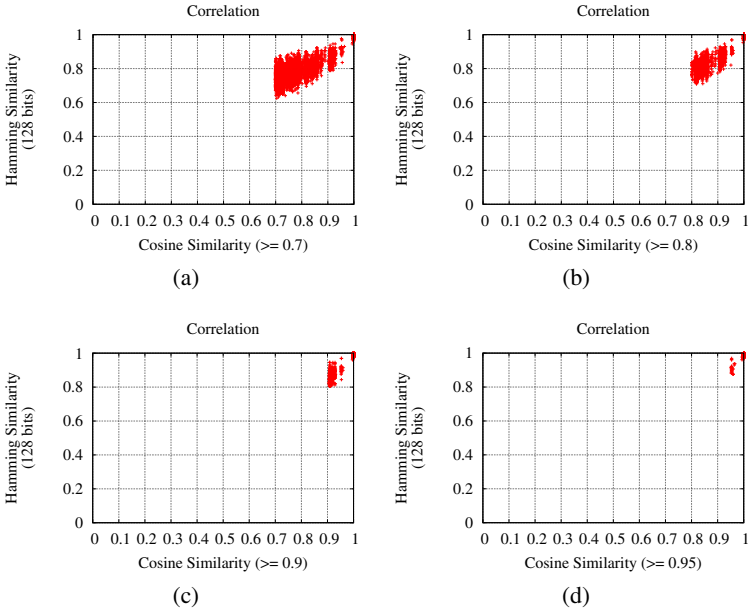


Fig. 9. Correlation between cosine and Hamming similarity of their 128-bits identifiers with cosine similarity greater than or equal to 0.7 (a), 0.8 (b), 0.9 (c), and 0.95 (d)

The greater the cosine similarity, the greater the correlation. In Table 4 is shown the average of the correlation between cosine similarity (sim_{cos}) of pairs of queries and adult profiles, and the Hamming similarity (sim_{ham}) of their 128-bits identifiers. These samples were sorted according to these similarity levels: 0.7, 0.8, 0.9, and 0.95.

Table 4. Average of the correlation between cosine and Hamming similarity

sim_{cos}	Average of the Correlation
≥ 0.7	0.84
≥ 0.8	0.90
≥ 0.9	0.95
≥ 0.95	0.98

The results show a strong correlation between cosine and Hamming similarity for 128-bits pairs of adult profiles, especially for the highest similarity levels. Still examining this table, the greater the similarity level, the greater the correlation between cosine similarity and profile identifiers. The standard deviations of these averages for the lowest similarity levels are high due to the probabilistic characteristic of the RHH function, but the confidence intervals of the samples were calculated and are low and acceptable. For the highest similarity levels the standard deviations are negligible, as the confidence interval of the samples.

Another test to express the feasibility of the similarity search in this scenario is the evaluation of the frequency distribution of the Hamming distance of pairs of content identifiers, according to their cosine similarity. As indicated in Section 5.3.1, each profile in the Adult data set was used as a query vector in the entire database. For each pair of adults, the cosine similarity and the corresponding Hamming distance of their identifiers were measured. The evaluation was done using four different similarity intervals: [0.7..0.8), [0.8..0.9), [0.9..0.95), [0.95..1).

Figure 10(a) shows the frequency distribution of the Hamming distance of the 128-bits user's identifiers, which have a cosine similarity greater than or equal to 0.7 and less than 0.8 [0.7..0.8). As depicted, the results tends to a normal distribution in which most of the distances are between 20% (25 bits) and 30% (38 bits) of the total length of the identifier. The same behavior can be observed in the other ranges: [0.8..0.9) (Figure 10(b)), [0.9..0.95) (Figure 10(c)), and [0.95..1.0) (Figure 10(d)).

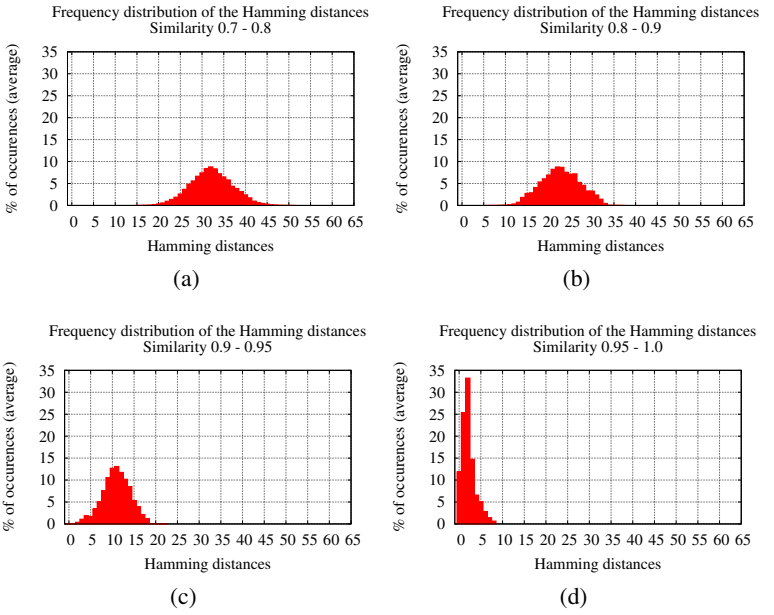


Fig. 10. Frequency distribution. Similarity levels 0.7 (a), 0.8 (b), 0.9 (c) and 0.95 (d).

5.2 Hamming DHT

This section describes some experiments aiming to evaluate the Hamming DHT. The main idea is to validate such DHT as a valuable approach in order to support the searching for similar contents in a distributed environment. To evaluate the Hamming DHT proposal, the following experiments were performed:

- Adult profile identifiers of 128-bits length were generated using RHH;
- These profiles' identifiers were indexed and distributed in the Hamming and Chord DHTs with 1000 and 10000 peers;

- Each profile in this set was used as a query vector for similar profiles in all profiles of the same data set. There are a total of 48842 adult profiles in the data set, which means that 48842 x 48842 queries were realized over the Adult Data Set;
- An identifier was generated for each query and a lookup having such identifier as argument was executed in both DHTs. The lookup message is forwarded to the peer which is responsible for the query identifier (the hosting peer), i.e., the identifier's successor on the ring;
- From the hosting peer, each profile identifier within the query's similarity level is retrieved, and the distance in number of hops from the hosting peer to the other peers hosting each similar contents is measured.

These tests permitted us to highlight the following aspects:

- The frequency distribution of the number of hops to retrieve all similar profiles in the set according to their similarity level. This evaluation shows that the Hamming DHT aggregates more than a normal ring-style DHT, such as Chord, reducing the distance between similar profiles in the number of hops;
- The query recall, corresponding to the fraction of the profiles that is relevant to the query and successfully retrieved. This evaluation shows that it is possible to build a more efficient search engine on top of the Hamming DHT, at lower cost, measured in the number of hops to complete the query.

To perform the tests, we have implemented a Chord and a Hamming DHT simulator, including a feature to generate random peers and join them in a ring-style DHT. Also, the developed simulator indexes and stores each profile identifier k using the $put(k,v)$ operation. The $lookup(k)$ operation returns the successor of the key k on the ring which represents the peer responsible for storing the profile associated to this key, the hosting peer. The $get(k)$ primitive was extended to handle the proposed similarity level assuming the format $get(k,sim)$: given a key (k) and a similarity level (sim), all similar profiles stored in the hosting peer are returned. This search is extended to the neighbors of the hosting peer with distance of 1 hop (or longer distances) aiming to improve the searching results.

5.2.1 Distribution of the Number of Hops

The graphs in Figure 11 show the frequency distribution of the number of hops to retrieve all similar contents. The graphs exhibit the percentage of recovered contents, which is relative to the total number of similar contents in the set, and their corresponding distance to the hosting peer. The tests were performed varying the number of peers in each DHT to analyze this influence in the results. It was simulated 1000 peers in Chord and in the Hamming DHT with similarity levels 0.7 (Fig. 11(a)), 0.8 (Fig. 11(c)), and 0.9 (Fig. 11(e)). Also, Fig. 11(b), 11(d), and Fig. 11(f) show the results obtained from the same tests simulated with 10000 peers. The size of the keys used in these tests is 128-bits.

From these results, it is possible to notice that the Hamming DHT is able to cluster more similar profiles in lower number of hops, i.e., shorter distances between them. The confidence interval for 95% of the samples are negligible. The process of acquiring

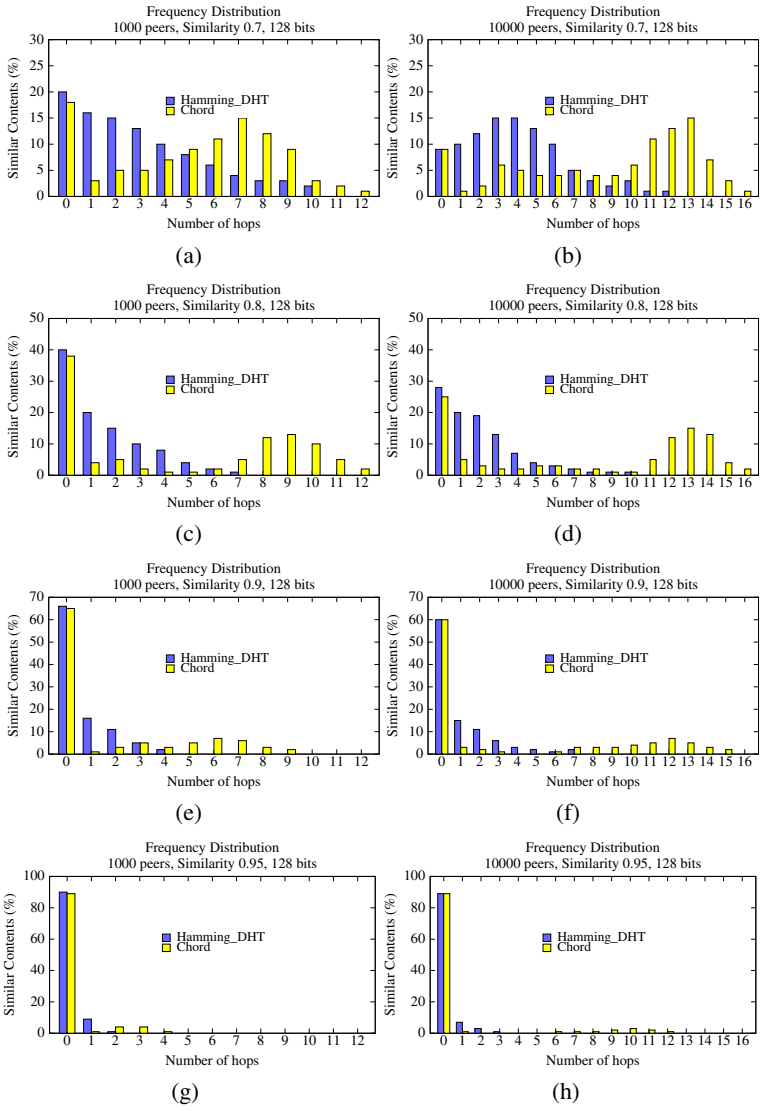


Fig. 11. Frequency distribution of the number of hops and 128-bit keys. 1000 peers with similarity levels 0.7, 0.8, 0.9, and 0.95. 10000 peers with similarity levels 0.7, 0.8, 0.9, and 0.95.

fingers of the Hamming DHT, which privileges the Hamming distance between peers, and the organization of the identifiers according to the Gray code sequence in the ring, contributes to the results presented in these tests.

5.2.2 Recall

Figures 12(a), 12(c), and 12(e) show the recall in a similarity search for the Chord and the Hamming DHT with 1000 peers, similarity levels 0.7, 0.8, and 0.9 and 128-bit.

Figures 12(b), 12(d), and 12(f) show the results obtained from the same tests simulated with 10000 peers.

From the results of Figure 12, it is possible to see that using the Hamming DHT as an infrastructure to support similarity search is a valuable approach. As an example, from Figure 12(d), it is possible to see that a search engine designed over the Hamming DHT having a $get(k, 0.8)$ function with a depth of 4 hops, about 91% of all similar profiles can be retrieved, while in Chord, only about 50% of them can be retrieved. The better frequency distribution shown in Figure 11 justifies the better recall obtained by the Hamming DHT when compared to Chord.

5.3 HCube

This section describes the experiments performed in the evaluation of the HCube proposal. The main idea is to validate our proposal as a valuable approach in order to support the searching for similar data. To evaluate the HCube the following experiments were performed:

- Adult profile identifiers with 128-bits were generated using RHH, indexed and distributed in HCubes with 1024, 2048, 4096, and 8192 servers;
- Profiles in the query set were used as query vectors in the Adult set. Pairs of queries and adults were randomly selected to be evaluated considering similarity levels greater than or equal to 0.7, 0.8, 0.9, and 0.95;
- A 128-bits identifier was generated for each query. Lookups having these identifiers and a Hamming similarity level were performed. To map the hosting server for each query, each 128-bits identifier was reduced to a 10-, 11-, 12-, and 13-bits identifier using RHH. These reduced identifiers correspond to the hosting servers of the queries for each evaluated size of HCubes;
- From the hosting server, all profiles that fit within the Hamming similarity level are retrieved.

These tests allow us to highlight the following aspects:

- The correlation between cosine and Hamming similarity of adult identifiers (128-bits), and between cosine and Hamming similarity of hosting server identifiers;
- The frequency distribution of the number of hops to retrieve all similar profiles in the set according to their similarity level;
- The query recall, corresponding to the fraction of all relevant profiles that was successfully retrieved.

To perform the tests, adult profiles were indexed and distributed in the HCube using the $put(k, v)$ primitive. The $get(k, sim)$ primitive was used to retrieve all similar profiles in a hosting server and its neighbors given an identifier (k) and a similarity level (sim). Only to permit us to evaluate the recall and the number of hops, this search was extended to all neighbors of the hosting server in a configurable depth.

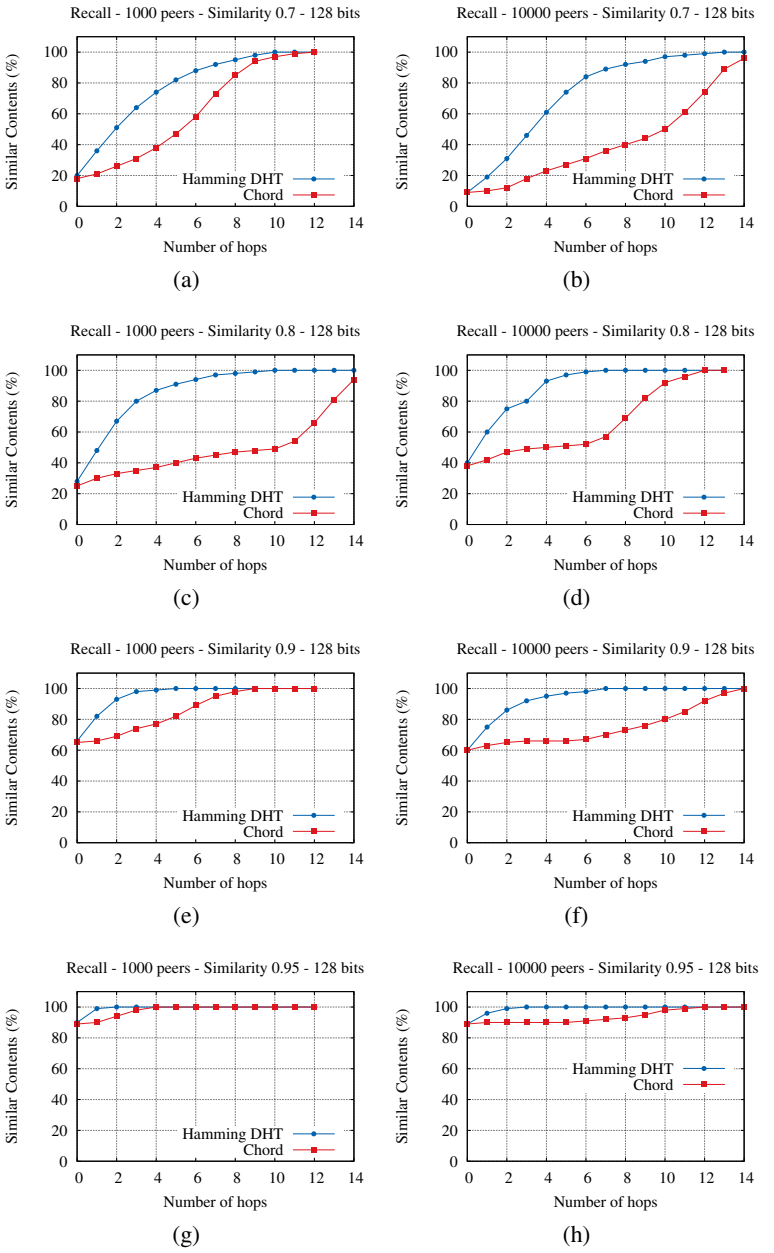


Fig. 12. Recall. 1000 peers, similarity levels 0.7, 0.8, 0.9, and 0.95. 10000 peers, similarity levels 0.7, 0.8, 0.9, and 0.95.

5.3.1 Correlation

Table 5 shows the average of the correlation between cosine similarity (sim_{cos}) of pairs of queries and adult profiles, and the Hamming similarity (sim_{ham}) of their 128-bits identifiers. Also, the correlation between cosine similarity and the Hamming similarity of 10-, 11-, 12-, and 13-bits hosting server identifiers is presented. To calculate these correlations we randomly selected 10 samples of queries and adults profiles, each sample containing about 1% of the total number of pairs in the Adult set. These samples were sorted according to the similarity levels: 0.7, 0.8, 0.9, and 0.95.

Table 5. Average of the correlation between cosine and Hamming similarity

sim_{cos} Level	128-bits sim_{ham}	10-bits sim_{ham}	11-bits sim_{ham}	12-bits sim_{ham}	13-bits sim_{ham}
≥ 0.7	0.84	0.44	0.48	0.50	0.52
≥ 0.8	0.90	0.48	0.52	0.63	0.68
≥ 0.9	0.98	0.79	0.84	0.86	0.90
≥ 0.95	$\approx$1.00	0.98	0.99	0.98	$\approx$1.00

The results show a strong correlation between cosine and Hamming similarity for 128-bits pairs of adult profiles, especially for the highest similarity levels. When the similarity level falls bellow 0.9 it is possible to notice a moderate correlation (< 0.8) between cosine and Hamming similarity of pairs of adults and their hosting server identifiers. Still examining this table, the greater the similarity level, the greater the correlation between cosine similarity of pairs of adults and Hamming similarity of their hosting server identifiers.

The standard deviation of these averages for the lowest similarity levels are high due to the probabilistic characteristic of the RHH function, but the confidence intervals of the samples were calculated and are low and acceptable. For the highest similarity levels the standard deviations are negligible, as the confidence interval of the samples.

5.3.2 Distribution of the Number of Hops

The results in Figure 13 show that the higher is the similarity level of a query, the lower is the effort necessary to recover similar data. As an example, in Figure 13(a), using 10-bits for a server identifier and a Hamming similarity level equal to 0.7, most of the similar profiles of a given query are located between 2 and 6 hops away from the hosting server. It is important to notice here that using 10-bits in a server identifier, a Hamming similarity of 0.7 is equivalent to a Hamming distance of 3, on average. In an HCube with 6-bits (a complete HCube), having 3 as the Hamming distance implies in 3 hops from source to target. As another example, in Figure 13(c), also using 10-bits for the server identifier and a similarity level of 0.9, most of the similar data associated to a given query are between 0 and 3 hops. A Hamming similarity of 0.9 in this case means that the Hamming distance between server identifiers storing the similar data is equal to 1, on average.

In a complete Hamming cube with n dimensions, in which n is the number of bits of the server identifier, with a similarity level sim the maximum number of hops necessary

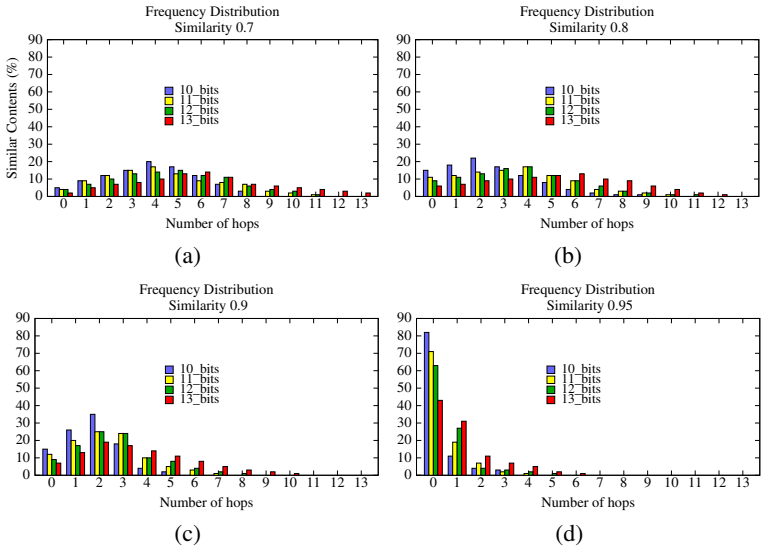


Fig. 13. Frequency distribution. Similarity levels 0.7 (a), 0.8 (b), 0.9 (c), and 0.95 (d).

to recover similar data is *n*-(*sim***n*). As an example, with *sim*=0.7 and *n*=10, the maximum number of hops necessary to recover all 0.7 similar data is 10-(0,7*10) = 3. The highest values presented in the HCube results are due to the reduced number of dimensions necessary to build an incomplete Hamming cube. The average of the number of hops to retrieve all similar profiles in a given query is presented in Table 6:

Table 6. Average of the number of hops

Similarity Level	1024 servers (10-bits)	2048 servers (11-bits)	4096 servers (12-bits)	8192 servers (13-bits)
0.7	3.88	4.35	4.71	5.75
0.8	2.56	3.47	3.72	5.07
0.9	1.76	2.32	3.65	3.48
0.95	0.28	0.43	0.57	1.10

5.3.3 Recall

An important metric to indicate the efficiency of any information retrieval system is the recall, which represents the fraction of the profiles that is relevant to the query and successfully retrieved in a similarity search. This evaluation shows that it is possible to build an efficient search engine on top of the HCube. In Figure 14 the results of the evaluation of the recall are presented. They represent the average of all queries and results in the Adult set.

From these results, it is possible to see that the best results occur for the highest similarity levels. As an example, taking an HCube with 4096 servers (12-bits) and similarity

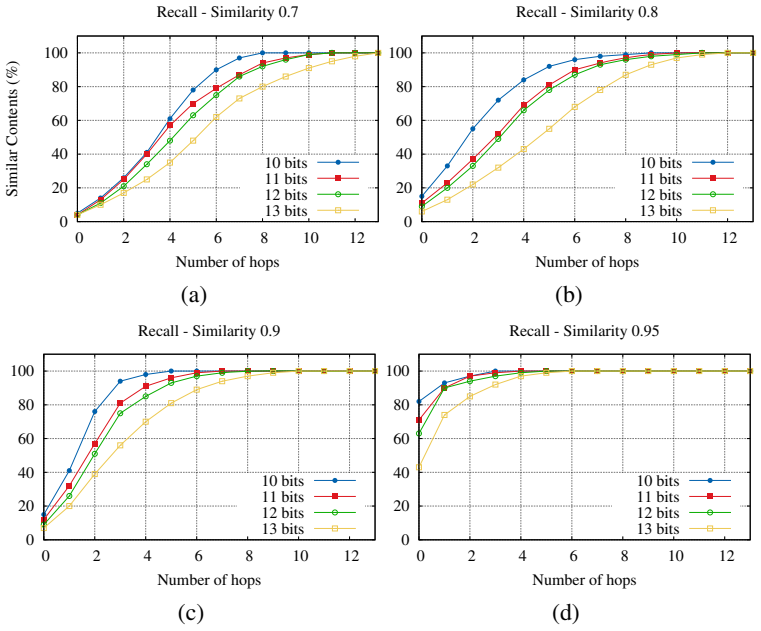


Fig. 14. Recall. Similarity levels 0.7 (a), 0.8 (b), 0.9 (c), and 0.95 (d).

level equal to 0.8, 20% of the similar adult profiles are retrieved using 2 hops and 50% of them are retrieved with 4 hops. In the same scenario, using similarity level of 0.95, 60% of all similar profiles are stored in the hosting server, and expanding the search for its immediate neighbors (1 hop), 90% of all similar profiles are retrieved.

In the next section the chapter is concluded and future works are discussed.

6 Conclusions and Further Research Issues

In short, the similarity search based on Hamming metric is an innovative proposal. The basic assumption is that users with similar profiles have similar interests. A user profile can be obtained in several ways, such as by crawling social networks, extracting demographic informations, curriculum vitae, or user accounts. Compared to other related works in the literature, the main advantages of the proposed similarity search are: is does not suffer from the curse of dimensionality; the similarity is evaluated independently from the type and semantics of each dimension of the content vector; it involves low cost computational operations, such as the XOR function, to compute the similarity between two identifiers. Also, the similarity search is well adapted to be indexed in virtually [7] or physically distributed systems [8] and can be extended to be a hybrid approach.

The use of weights in the similarity search proposal is a work to investigate in the future. In this initial experimentations, the use of different scales in some profile attributes causes them to contribute at different levels to estimate the similarity between profile

vectors. As an example in the Adult set, if the user sets "age" with a greater weight than the other characteristics, in Table 2, result R_9 could be more similar to the query q than R_{10}. The way users express their preferences can be implemented in a collaborative way.

Another aspect of the similarity search is that it becomes easily adapted to be a dissimilarity search system. A dissimilarity system can be used in several applications to contrast different profiles in a recommendation. It is only necessary to search for the negate identifier of a profile. As an example, a dissimilarity search for the following profile <27; Private; Some-college; 11; Never-married; Adm-clerical; Own-child; White; Female; United-States; >=50K> using 0.9 as a dissimilarity level, returns <38; Private; HS-grad; 9; Married-civ-spouse; Craft-repair; Husband; White; Male; Cuba; <=50K> and <37; Private; HS-grad; 9; Married-civ-spouse; Sales; Husband; White; Male; Haiti; <=50K>, among others.

The results show that the Hamming DHT can be a useful tool to serve as an overlay infrastructure for similarity search. The evaluation compares the Hamming DHT and Chord because it can be used as a reference element in the DHT literature, even if it has not been proposed for similarity search. Also, to the best of our knowledge, no other chapter in the DHT literature explores the Hamming similarity of content identifiers to propose a DHT specialized for similarity search, which makes difficult our comparisons with other approaches.

HCube is a Data Center solution designed to support similarity searches in Big Data scenarios, aiming to reduce the distance and to improve the recall of similar content. This chapter shows that the union of the VSM representation, the RHH function, the Gray SFC, the three-dimensional structure used in a server-centric Data Center and the XOR-based routing solution provide the necessary substrate to efficiently achieve the objectives of HCube. As future work, alternatives for HCubes bigger than 8192 servers will be investigated. A practical approach could be the increase in the number of servers' NICs. However, as it is a physically limited approach, we believe that an interesting option can be the introduction of an HCube hierarchy.

References

1. Gantz, J., Reinsel, D.: The Digital Universe Decade - Are You Ready? `http://www.emc.com/collateral/analyst-reports/idc-digital-universe-are-you-ready.pdf` (2010) (Online; Acesso em 2 de Março de 2013)
2. The Apache Software Foundation: Apache® Hadoop, `http://hadoop.apache.org/` (2013) (Online; Acesso em 5 de Março de 2013)
3. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Commun. ACM 51(1), 107–113 (2008)
4. Indyk, P., Motwani, R.: Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In: STOC 1998: Proceedings of the 30th Annual ACM Symposium on Theory of Computing, pp. 604–613. ACM, New York (1998)
5. Charikar, M.S.: Similarity Estimation Techniques from Rounding Algorithms. In: STOC 2002: Proceedings of the 34th Annual ACM Symposium on Theory of Computing, New York, NY, USA, pp. 380–388 (2002)

6. Frank, A., Asuncion, A.: UCI machine learning repository (2010), http://archive.ics.uci.edu/ml
7. Villaça, R., de Paula, L.B., Pasquini, R., Magalhães, M.F.: Hamming DHT: Taming the Similarity Search. In: Proceedings of the 10th Annual IEEE Consumer Communications and Networking Conference, CCNC 2013. IEEE Communications Society, Las Vegas (2013)
8. Villaça, R., Pasquini, R., de Paula, L.B., Magalhães, M.F.: HCube: A Server-centric Data Center Structure for Similarity Search. In: Proceedings of the 27th International Conference on Advanced Information Networking and Applications, AINA 2013. IEEE Computer Society, Barcelona (2013)
9. Desai, A., Singh, H., Pudi, V.: DISC: Data-Intensive Similarity Measure for Categorical Data. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011, Part II. LNCS (LNAI), vol. 6635, pp. 469–481. Springer, Heidelberg (2011)
10. Lee, D., Park, J., Shim, J., Lee, S.: Efficient Filtering Techniques for Cosine Similarity Joins. INFORMATION-An International Interdisciplinary Journal 14, 1265 (2011)
11. Lawder, J.: The application of Space-filling Curves to the Storage and Retrieval of Multidimensional Data. PhD thesis, University of London, London (December 1999)
12. Zhang, C., Xiao, W., Tang, D., Tang, J.: P2P-based multidimensional indexing methods: A survey. J. Syst. Softw. 84(12), 2348–2362 (2011)
13. Olszak, A.: Hycube: a dht routing system based on a hierarchical hypercube geometry. In: Wyrzykowski, R., Dongarra, J., Karczewski, K., Wasniewski, J. (eds.) PPAM 2009, Part II. LNCS, vol. 6068, pp. 260–269. Springer, Heidelberg (2010)
14. Tang, C., Xu, Z., Mahalingam, M.: psearch: information retrieval in structured overlays. SIGCOMM Comput. Commun. Rev. 33, 89–94 (2003)
15. Bhattacharya, I., Kashyap, S., Parthasarathy, S.: Similarity Searching in Peer-to-Peer Databases. In: Proceedings of the 25th IEEE International Conference on Distributed Computing Systems, ICDCS 2005, pp. 329–338 (June 2005)
16. Chang, F., Dean, J., Ghemawat, S., Hsieh, W.C., Wallach, D.A., Burrows, M., Chandra, T., Fikes, A., Gruber, R.E.: Bigtable: a distributed storage system for structured data. In: Proc. of the 7th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2006, vol. 7. USENIX, Berkeley (2006)
17. DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, P., Vogels, W.: Dynamo: amazon's highly available key-value store. SIGOPS Oper. Syst. Rev. 41(6), 205–220 (2007)
18. Stoica, I., Morris, R., Liben-Nowell, D., Karger, D.R., Kaashoek, M.F., Dabek, F., Balakrishnan, H.: Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications. IEEE/ACM Trans. Netw. 11(1), 17–32 (2003)
19. de Paula, L.B., Villaça, R.S., Magalhães, M.F.: Analysis of Concept Similarity Methods Applied to an LSH Function. In: COMPSAC 2011: Computer Software and Applications Conference. IEEE, Munich (2011)
20. Faloutsos, C.: Gray Codes for Partial Match and Range Queries. IEEE Trans. Software Eng. 14(10), 1381–1393 (1988)
21. Pasquini, R.: Proposta de Roteamento Plano Baseado em uma Métrica de OU-Exclusivo e Visibilidade Local. Phd. thesis, Faculdade de Engenharia Eletrica e Computação. Universidade Estadual de Campinas, Campinas, SP (June 2011)

Data Modeling for Socially Based Routing in Opportunistic Networks

Radu-Ioan Ciobanu[1], Ciprian Dobre[1], and Fatos Xhafa[2]

[1] University Politehnica of Bucharest, Splaiul Independentei 313, Bucharest, Romania
radu.ciobanu@cti.pub.ro, ciprian.dobre@cs.pub.ro
[2] Universitat Politècnica de Catalunya, C/Jordi Girona Salgado 1-3, 08034 Barcelona, Spain
fatos@lsi.upc.edu

Abstract. Opportunistic networks are the next step in the evolution of mobile networks, especially, since the number of human-carried mobile devices such as smartphones and tablets has greatly increased in the past few years. They assume unselfish communication between devices based on a store-carry-and-forward paradigm, where mobile nodes carry each other's data through the network, which is exchanged opportunistically. In this chapter, we present opportunistic networks in detail and show various real-life scenarios where such networks have been successfully deployed or are about to be, such as disaster management, smart cities, wildlife tracking, context-aware platforms, etc. We highlight the challenges in designing successful data routing and dissemination algorithms for opportunistic networks, and present some of the most important techniques and algorithms that have been proposed in the past few years. We show the most important issues for each of them, and attempt to propose solutions for improving opportunistic routing and dissemination. Finally, we present what the future trends in this area of research might be, from information-centric networks to the Internet of Things.

1 Introduction

Opportunistic networks are extensions of the legacy Mobile Ad Hoc Networks (MANETs) concept. Legacy MANETs are composed of mobile nodes that collaboratively set up a network plane by running a given routing protocol. Therefore, the sometimes implicit assumption behind MANETs is that the network is well connected, and nodes' disconnection is an exception to deal with. Most notably, if the destination of a given message is not connected to the network when the message is generated, then that message is dropped after a short time (i.e., the destination is assumed to not exist). Opportunistic networks are mobile wireless networks in which the presence of a continuous path between a sender and a destination is not assumed, since two nodes may never be connected to the network at the same time. The network is assumed to be highly dynamic, and the topology is, thus, extremely unstable and sometimes completely unpredictable. Nevertheless, the network must guarantee end-to-end delivery of messages despite frequent disconnections and partitions.

The opportunistic networking paradigm is particularly suitable to those environments that are characterized by frequent and persistent partitions. In the field of wildlife tracking, for example, some kinds of sensor nodes are used to monitor wild species. In these

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_2

cases it is not easy (nor possible sometimes) to have connectivity among a source sensor node and a destination data collector node. This happens because the animals to be monitored move freely and there is no possibility to control them in such a way to favor connectivity. Opportunistic networks may also be exploited to bridge the digital divide. In fact, they can support intermittent connectivity to the Internet for underdeveloped or isolated regions. This can be obtained by exploiting mobile nodes that collect information to upload to the Internet as well as requests for Web pages or any kind of data that need to be downloaded from the Internet. Both data and requests are uploaded to and downloaded from the Internet once the mobile data collector node reaches a location where connectivity is available.

This chapter focuses on the particular problem of data gathering in such challenged networks. We describe different alternatives and solutions for routing the data from a source to its destination. Today different techniques could be employed for mobile data gathering. A basic strategy would be to only allow data delivery when mobile devices are in direct proximity of the sinks. This technique has very little communication overhead, given that messages are only sent directly from the sensor node generating messages to the sinks. However, depending on how frequently mobile nodes meet the sinks, the delivery of the data might be very poor. This is particularly true if the sinks are very few and spread out. More refined techniques include epidemically inspired approaches, which would randomly spread the data over the network, so that eventually a sink could be reached. We analyze both these worlds, highlighting specific problems and solutions to solve them in concrete case scenarios.

The rest of this chapter is organized as follows. We first introduce in more detail the domain of opportunistic mobile communication. We present case studies where opportunistic networks are already being successfully deployed and used, and highlight specific issues regarding their implementation in particular scenarios. In Section 3 we perform an analysis of the latest advances in data routing and dissemination solutions based on the use of opportunistic mobile networks. We highlight the main issues of each proposal, and attempt to give solutions to some of them in Section 4. Section 5 presents future trends, and Section 6 concludes our study.

2 Opportunistic Networks

This section presents a detailed view regarding opportunistic networks (including definitions, benefits, and challenges), as well as a presentation of real-life use cases where they can bring a significant contribution.

2.1 Definition

Opportunistic networks (ONs) are a natural evolution of MANETs, where most (or sometimes all) of the nodes are mobile wireless devices. These devices range from small wireless-capable sensors to smartphones and tablets. The evolution from MANETs to ONs was necessary because opportunistic networks help transmit data horizontally, i.e., using costless inter-device transmissions, taking advantage of the already-existent device interaction. Moreover, ONs help disseminate data and decongest currently existing

backend protocols by using short-range communication over IEEE 802.11, Bluetooth, ZigBee, etc. They are also considered to be the solution that will offer vehicle-to-vehicle communication for future Vehicular Ad Hoc Networks (VANETs). The composing nodes of an ON have no knowledge of the shape of the network when they join it. They are only aware of other nodes that they come in close proximity to, depending on the radius of their communication mechanism (e.g., WiFi, Bluetooth, NFC, etc.). Thus, no assumptions are made regarding the existence of paths between nodes, since a network topology is not known by any node (and even if were known, it would much too dynamic to be used in routing mechanisms, since nodes are highly mobile and they hardly stay in the same place for long periods of time).

ONs are based on a paradigm entitled store-carry-and-forward, or SCF [38]. This implies that a node begins by storing some local data, which was either generated by itself, or was received from another node. Since ONs are characterized by a high degree of mobility, nodes move around a lot, thus carrying the stored data around the network. Finally, when the destination for a particular data item[1] is encountered, a forwarding process is started in order to send the data to the destination. However, encountering a message's destination is not the only situation where a data forward occurs. Opportunistic networks are based on the altruism of nodes: it is not enough for a node to see only to its interest, instead it must help other encountered nodes transfer their data (in exchange for them carrying the node's messages as well). This mutual help is the key to ONs and helps ensure that the data are spread through the network as much as possible, thus, increasing the probability of a message to reach its intended destination, and decreasing the time it takes to do so.

There are two important parameters that describe the shape and behavior of an opportunistic network: contact time and inter-contact time [9]. The contact time is the duration of a contact between two encountering nodes, and it represents the time window when the nodes may exchange data. Higher contact times lead to the opportunity of exchanging more data between the nodes, but may also mean that the two nodes are static and the chance of them encountering other nodes and spreading the data is lower. Inter-contact time is the duration between two consecutive contacts of the same pair of nodes, and it offers an indication of the familiarity between the two nodes. If they have low inter-contact times, they meet each other very often, so they probably move around in the same geographical areas. The two parameters can be extended to the entire network, as any-contact time and inter-any-contact time [28]. The any-contact time is the duration of a node's encounter with any other node, whereas the inter-any-contact time represents the time between a node's sighting of any two nodes. The sparser the network, the lower the inter-any-contact time is, which leads to few forwarding opportunities, so the messages have a lower chance of reaching their intended destinations.

Opportunistic networks have been thoroughly studied in [38]. The authors offer a definition of ONs and present several realistic case studies of opportunistic networks that have been successfully deployed, including pocket-switched networks (PSNs) in the Haggle project[2], wildlife monitoring, and ONs for intermittent Internet connectivity

[1] In opportunistic networks, data items are generally called messages. From this point on, we will refer to them thus, making a distinction only where the situation requires it.

[2] `http://www.haggleproject.org`.

(which we discuss in more detail in Section 2.3). Furthermore, the paper also analyzes several ON routing and forwarding algorithms, while proposing a taxonomy used to classify them. They are split into algorithms with and without infrastructure. The infrastructure-based algorithms may be based on dissemination or on context, whereas the algorithms without an infrastructure are split based on their infrastructure type (fixed or mobile). Another detailed study of ONs has been performed by Conti et al. [17]. Their paper describes opportunistic networks, stating that the understanding of human mobility is paramount to designing efficient protocols for opportunistic networking. Conti et al. also discuss ON architecture, forwarding algorithms, data dissemination, security, and applications and conclude their work by observing that there is a strong link between opportunistic networking and mobile social networks. The authors also show that ONs can be used for both point-to-point communication as well as data dissemination.

2.2 Challenges

Aside from the benefits of opportunistic networks and their applicability in real-life (presented in Section 2.3), there are several challenges that must be taken into consideration when designing an ON-based solution. The first and most important caveat of ONs is that the lack of connectivity at all times leads to a potential lack of end-to-end paths. In other words, deploying an opportunistic network means accepting the fact that not all messages may successfully reach their destinations, or when they do, reach them with high delays. As stated in Section 2.1, there are many factors that can affect an ON's hit rate[3], ranging from the number of devices in the network to the behavior and social grouping of the device's owners (if we are dealing with an ON where the nodes are humans carrying mobile devices). Opportunistic network administrators must be aware of this and only use such networks where delays and loss of messages are acceptable. The purpose of every ON routing or dissemination algorithm is to increase the hit rate, because higher hit rates make ONs much more likely to be successfully used in real-life scenarios.

Closely related to the first challenge is the decision of selecting a message's next hop. Ideally, each node should have access to the future behavior of the entire network, and thus, choose the shortest path between it and a message's destination, similar to what is done for classical static networks. Unfortunately, this is not the case for opportunistic networks, and this is why researchers are still proposing new methods of deciding whether a message should be exchanged when two nodes are in range of each other. Aside from selecting the next hop, decisions should be made regarding the amount of copies a message should have in the network, and whether it should be kept or deleted by the originating node. Various methods for routing or dissemination algorithms have been proposed over the years, and the most successful ones are presented in Section 3, along with their (still existing) issues.

The main bottlenecks of data dissemination in ONs are large inter-contact times and slow nodes. If a large period of time passes before a node is encountered by any other

[3] The percentage of messages that reach their destinations, out of the total number of messages sent in the network.

device, then there is no chance of that node receiving the information it is interested in. Therefore, opportunistic networks should be as dense as possible, so that each node gets a fair chance of receiving data from others. Furthermore, slower nodes may also act as bottlenecks, especially, if they are central nodes with lots of contacts. If this is the case, they are considered as suitable forwarders by other nodes and data are sent to them. Since they download slowly, they prevent the other nodes from being able to transfer many data items at once, thus slowing down the entire network. This is why an opportunistic network should be carefully configured to use suitable algorithms, that are able to detect bottlenecks caused by slow and selfish nodes, and which can take advantage of a contact between two nodes optimally.

Another aspect that should be taken into consideration when deploying an opportunistic network is that, since the nodes are generally mobile devices, they have a limited life until they need to be recharged. The more data transfers are performed in a short period of time, the quicker a device's battery consumes, which in turn leads to removing it from the network for a certain period of time (until it is recharged). Furthermore, congestion also leads to quicker energy consumption, since a node has to retry sending messages when the receiving node is flooded with data forwards. Asymmetric data rates can also cause needless power consumption, because the node with the faster connection is blocked by the slower node until all the data is exchanged, so it's working at a slower rate than it actually can.

An area of opportunistic networks that hasn't been researched too much is security. Along with privacy, it is a key condition to people accepting opportunistic networks where their devices are nodes. This would imply that a message sent by a node can only be decrypted by the intended recipient, and that nodes can't enter the network and perform malicious deeds (such as flooding nodes with data, reporting false information, etc.). Moreover, nodes should be stopped from being selfish, using incentive mechanisms.

By taking all these challenges into consideration, the main goal of opportunistic networks is to achieve real mobile computing without the need for a connected network. Researchers are not there yet, but the algorithms and solutions proposed have been better and better over the past few years, so the research is heading into the right direction. It probably won't be too long until we will be able to see and use opportunistic networks for various purposes.

2.3 Use Cases

This subsection presents real-life use cases for ONs. It highlights various areas where opportunistic networks have been or are soon to be deployed.

2.3.1 Disaster Management

One scenario where opportunistic networks may prove to be very important is disaster management. When a disaster such as an earthquake, a tsunami, or an explosion occurs, legacy communication might collapse, due to potential damage to the physical components of the network, such as switches or cables. Therefore, need arises for a method of ensuring more efficient and dependable solutions that can be employed for

security missions. One such solution is proposed by Bruno et al. [8], and it implies using ONs to create an overlay infrastructure for rescue and crisis management services. The proposed solution uses the unaffected components of the static infrastructure (i.e., the ones that have not been damaged by the disaster), by making them act as nodes in opportunistic networks. Aside from these ad hoc nodes, special networks and connectors are also deployed, and even singular mobile devices belonging to the survivors or to people nearby the disaster spot may be used. The main goal is to offer connectivity where otherwise there would be none, which leads to a higher efficiency in finding survivors or organizing the rescue efforts. Moreover, another goal is to lower the congestion rate, since regular network infrastructures tend to become very crowded during such incidents.

A similar solution is proposed by Lilien et al. [34], where systems that were not originally nodes of an opportunistic network dynamically join it, with the purpose of aiding with communication in disaster situations. MAETT (Mobile Agent Electronic Triage Tag) and Haggle-ETT [36] are two similar methods that are used to collect the triage data (i.e., location) of a disaster victim, when regular communication systems are down. They allow it to be collected and represented in an electronic format, which can then be transmitted to coordination points where it is processed and made available for the rescue missions. The difference between the two algorithms is that MAETT uses mobile agents for storing the electronic triage tag, whereas Haggle-ETT is based on the Haggle architecture.

2.3.2 Smart Cities

Cities are areas where Big Data is having a real impact. Town planners and administration bodies just need the right tools at their fingertips to consume all the data points that a town or city generates and then be able to turn that into actions that improve people's lives. In this case, Big Data is definitely a phenomenon that has a direct impact on the quality of life for those that choose to live in a town or city. Smart cities of tomorrow will rely not only on sensors within the city infrastructure, but also on a large number of devices that will willingly sense and integrate their data into technological platforms used for introspection into the habits and situations of individuals and city-large communities. Predictions say that cities will generate over 4.1 Terabytes per day per square kilometer of urbanized land area by 2016. Efficiently handling such amounts of data is already a challenge.

Smart cities monitor and integrate the conditions of all their critical infrastructures (such as roads, bridges, tunnels, rails, subways, airports, sea-ports, communications, water, power, etc.) in order to better optimize their resources and plan their preventive maintenance activities [10]. They connect the physical, IT, social, and business infrastructures, for the purpose of leveraging the collective intelligence of the cities. Opportunistic networks are the logical means of achieving at least a part of a smart city infrastructure, since they can be employed to perform communication between various parts of a smart city. For example, the traffic lights system can be opportunistically connected to a service that offers information about traffic jams, crowded roads, accidents, etc., so it can adapt to the conditions of the environment.

Moreover, mobile devices belonging to a smart city's citizens may also be opportunistically employed as sensor nodes, as shown by Le et al. [33]. The authors propose

a new routing algorithm for a heterogeneous architecture composed of various types of nodes, as opposed to existing routing algorithms, which are based on the idea that the nodes and technologies are homogeneous and can easily work together. Using mobile devices as nodes in a smart city leads to a much better knowledge of the conditions in various parts of the town, ranging from traffic information, to data such as temperature or weather conditions. The more information is available, the better the decisions made by the various subsystems of the smart cities are.

2.3.3 Floating Content

Another potential practical use of opportunistic networks is in regard to floating content in areas such as open city squares [18], which are geographical zones (also known as anchor zones) where mobile nodes enter, spend a certain amount of time, and leave. In this case, the nodes are mobile devices belonging to humans, and the anchor zones are relatively small-sized areas where many people congregate, which represent the boundaries of the ad hoc opportunistic network. While inside the anchor zone, the nodes may copy the data either if they need it for themselves, or if they transport it for the benefit of other nodes. If a node is interested in a certain data item floating around the anchor zone, it replicates it. Content availability in the anchor zone is probabilistic and best-effort, since the information can disappear from the area (i.e., it can sink) if unoptimal data sharing algorithms are employed. Nodes exiting the open city square delete the zone-specific content, since it is of no relevance outside the area in which it was generated, which means that the availability of the floating content is probabilistic. When dealing with floating content, the main requirement is that the content should easily be available, which happens when a certain fraction of the nodes in the anchor zone carry it. This is why ON-specific algorithms are employed and adapted for making the replication decisions. The best nodes should be chosen for replicating data, while at the same time congestion should be avoided. Desta et al. [18] attempt to increase the proportion of nodes having the content, and to have a high probability of bootstrapping the system (i.e., avoiding early extinction of the content during the initial phase).

2.3.4 Advertising

More recently, ONs have been used for other purposes, such as advertising. One such example is MobiAd [25], an application that presents the user with local advertisements in a privacy-preserving manner. The ads are selected by the phone from a pool of advertisements which are broadcast on the local mobile base station or received from local WiFi hotspots. Information about ad views and clicks is encrypted and sent to the ad channel in an opportunistic way, via other mobile devices or static WiFi hotspots. This helps ensure privacy, since the other nodes won't discover which ads were viewed, and the ad provider isn't able to know which user saw what ad.

Another way of applying opportunistic networking to advertising is proposed by Heinemann and Straub [26]. They base their proposition on word-of-mouth (WOM) communication or viral marketing, which means facilitating advertising where none of the participants are marketing sources. In classical WOM communication, a user A passes next to a store and sees a promotion for a certain item. Knowing that a friend B is interested in those types of items, A lets B know about the offer, thus advertising the product through word-of-mouth, even though there is no commercial interest from

user A. Heinemann and Straub propose using this mechanism over an ON infrastructure. Instead of showing their promotions in the window, stores have mobile devices that broadcast information about promotions and offers on a certain radius. When people with mobile devices pass near the store, their own device receives the offer and attempts to match it with the current user's preferences. If the user is interested, then an alert is shown. If not, the information is stored for opportunistic transmission. Assuming that node A received an offer it is not interested in, it carries it around until it encounters node B. If node B is interested, then the offer is forwarded to it, and its owner is notified. This mechanism would imply having an application where users can store their preferences, in order to help match the received offers.

2.3.5 Sensor Networks

Sensor networks [4] are the backbone of infrastructures such as smart cities or smart houses. They are composed of various types of sensors which are able to monitor environmental conditions, such as temperature, light, sound, etc. The data thus collected is sent to a main location, which analyzes it and may decide to act upon it (e.g., when the temperature is too high, the cooling system is started, or when the light is too low, the lights are turned on, etc.). Applications of sensor networks in real-life include environmental monitoring (air quality/pollution, forest fires, landslide, water quality, natural disasters, etc.), industrial monitoring (machine health, data logging, water, waste), or agriculture. When dealing with complex sensor network, where there are many heterogeneous sensors, opportunistic networks, are employed in order to improve the data-collecting process. Instead of every sensor sending data to a singular central processing unit, the data is collected by various mobile collectors, which preprocess the data before sending it to the central unit. This way, there are much fewer data transfers, and the central unit becomes less of a bottleneck. Moreover, it is able to respond more quickly to changes in the environment, leading to a more optimal behavior of the network.

2.3.6 Interplanetary Internet

Interplanetary Internet [1] is a specialized type of opportunistic network that would allow fast communication between the Earth and other planets (or even spaceships or probes sent way beyond the planets in our solar system). The first point of focus is on fast Mars communication, and this network has three basic components: NASA's Deep Space Network (DSN)[4], a six-satellite constellation around Mars together with a large Marsat satellite placed in low Mars orbit, and a new protocol for transferring data. The DSN would act as the portal between Earth and the Interplanetary Internet, the six smaller satellites and the Marsat would provide a full-time connection between the DSN and Mars, and the new protocol would be used to ensure that data is transferred optimally through the satellites. Communication with Mars would be the first step, with the planet acting as a node in the network, passing data further on to other planets. One proposed idea for the communication protocol is the Parcel Transfer Protocol (PTP), which assumes that data is transferred using the six smaller satellites opportunistically, by selecting the appropriate one according to the position of the satellites in orbit and

[4] An international network of antennas that is able to track data and control the navigation of interplanetary spacecraft, used by NASA.

the position of the point that communication must be performed with. Moreover, the satellites can relay data between each other before it reaches Mars.

2.3.7 Crowd Management

Another situation where ONs and DTNs may be used is in crowded areas, e.g., at locations where groups of individuals get separated and need to locate each other. In such scenarios (e.g., an amusement park where a child gets lost from his parents, a concert where a group of friends split and they need to find each other, a football match, etc.), contacts between mobile devices happen often and for longer periods of time, but (due to the high number of nodes in a very small space), classic communication means such as 3G and mobile telephony may not work properly. However, since the ON created ad hoc by leveraging the participants' mobile devices is very dense, the mobile devices may be used to opportunistically broadcast the location of mobile device owners to interested receivers. For example, a child at an amusement park might carry a smartphone that opportunistically broadcasts the child's encrypted location, by leveraging the neighboring devices. The child's parents have a smartphone of their own which receives the broadcast from encountered nodes, thus being aware of the child's location at any point in time. Similarly, non-emergency scenarios may also benefit from this solution, where events regarding the bands that will be playing or any other announcements can be disseminated to the participants of a music concert or any other type of social event with high participation. This would gradually replace the classic printed event programmes, offering up the possibility for more detailed and interactive information for attendees.

An existing crowd management project is Extreme Wireless Distributed Systems (EWIDS)[5], which attempts to use wireless sensor technology to monitor people's behavior, using crowd management as their application domain. Proximity graphs are generated using body-worn sensors carried by thousand of people, and the data is extracted and processed, either in real-time or offline. The extracted data is a series of graphs that evolve, used to provide feedback to a crowd of people, leading to crowd management.

2.3.8 Context-Aware Platforms

Another area that benefits from ONs is represented by platforms that support generic context-aware mobile applications. CAPIM (Context-Aware Platform using Integrated Mobile services) [19] is such a framework, and it is designed to support the construction of next-generation applications. It performs an integration of context data-collecting services, such as location, user's profile and characteristics, environment, in order to offer a centralized framework for context information. The smart services are dynamically loaded by mobile clients, which take advantage of the sensing capabilities provided by modern smartphones, possibly augmented with external sensors. A framework such as CAPIM can be employed in academic environments as a means to facilitate the interaction between students, professors, and other faculty members. The communication could be performed opportunistically, instead of using fixed infrastructures such as WiFi or 3G, in order to limit the bandwidth used and to save battery life (since Bluetooth

[5] `http://www.distributed-systems.net/index.php?id=extreme-wireless-distributed-systems.`

consumes less power than both WiFi and 3G). CAPIM would be used to disseminate announcements to students, signal their location, the classes they take, the interactions they have, etc. Other examples of context-aware platforms that might benefit from an ON-based framework include PACE [27], SOCAM [22] or CoWSAMI [3].

2.3.9 Wildlife Tracking

One of the first applications of opportunistic networks in real-life has been wildlife tracking, i.e., recording the behavior and movement paths of animals in their natural habitat, while being nonintrusive. This assumes that animals are equipped with special tags that are able to communicate with other similar tags and with the researchers' base stations. The tags are able to record the geographical coordinates of the animal carrying it, as well as the encounters with other animals wearing similar tags. Having the tags able to communicate with the base stations means that researchers do not have to capture the animal in order to retrieve the collected information anymore. Instead, tracked animals exchange collected data between each other, and upload it whenever they are in range of the base stations. These stations may either be mobile (such as cars/ships belonging to researchers, caretakers from the national parks, fishermen, etc.), or fixed. Opportunistic routing protocols are employed when two tracked animals meet each other, in order to decide which data will be exchanged between the two. Generally, history-based algorithms are used, since flooding is not feasible due to lack to storage space.

Two examples of such wildlife tracking applications are ZebraNet [32] and Shared Wireless Infostation Model (SWIM) [41]. ZebraNet was deployed in the Kenyan savanna to help track zebras wearing special collars. In this case, the base station was mobile, namely the researchers' vehicle that periodically moved around the savanna to collect data. SWIM, on the other hand, was used to track whales' movement, and the base stations were either mobile (seabirds) or fixed (buoys).

2.3.10 Internet Access in Limited Conditions

Another application of ONs, according to Pelusi et al. [38], is offering Internet access in limited conditions where no infrastructure exists. Two examples of such projects are the DakNet project [39] and the Saami Network Connectivity (SNC) [20] project. The DakNet project and the similar, but improved, KioskNet system [24], have the purpose of creating a low-cost asynchronous infrastructure that is able to provide connectivity in rural areas where the deployment of standard Internet access is not feasible or cost-effective (such as Indian villages). The two projects propose building kiosks in villages that are equipped with digital storage and wireless communication devices, which interact with mobile base stations periodically. Such stations are mounted on buses, motorcycles and bicycles, and collect the data from the kiosks and deliver it to Internet access points located in the cities (and vice versa). SNC provides a similar solution, but to Saami herders in Lapland, in order to protect their heritage and cultural identity, while still helping them integrate into the modern societies from their specific countries (Sweden, Norway, Finland).

2.3.11 Distributed Social Networks

Opportunistic networks can also be used to leverage dissemination of information between the members of a social group, without the need (or the expenses) for a

wired static infrastructure. One example of such an implementation is proposed by Thilakarathna et al. [43], which focuses on delay-tolerant applications and services such as content sharing or advertisement propagation between users who are geographically clustered into communities. The authors propose to address important ON issues such as lack of trust, privacy, or latency of delivery through combining the advantages of distributed decentralized storage and opportunistic communications. A community-based greedy heuristic algorithm is proposed, which is shown to maximize the content dissemination with limited number of replications. Another distributed social network application that uses ONs is DroidOppPathFinder [2], which generates and shares content about paths for fitness activity in a city, recommending the best paths from specific geographical areas through the analysis of user preference and context information collected by various sensing devices.

3 Data Routing and Dissemination

Ever since opportunistic networks were first presented, many data routing and dissemination algorithms have been proposed, ranging from very simple ones to more complex techniques that use prediction or social knowledge when performing routing decisions. There are many ways in which these algorithms may be classified (a taxonomy for routing algorithms is presented in [38], and one for dissemination techniques is proposed in [11]). However, we have split them here into basic, socially based, and history-and prediction-based algorithms, for simplicity. This section presents the most important algorithms in each of these three categories, highlighting their advantages, as well as their issues.

3.1 Basic Algorithms

The basic algorithms presented in this section are among the first algorithms proposed for opportunistic networks, and this is why they are simpler, not taking into consideration many of the aspects that ON routing and dissemination algorithms take for granted nowadays.

3.1.1 Epidemic

The Epidemic algorithm [44] is based on the way a virus spreads: when two potential carriers meet, the one with the virus infects the other one, if it isn't already infected. Thus, when an ON node A encounters a node B, A downloads all the messages from B that it does not already contain, and vice versa. The simplest version of this algorithm assumes that a node's data memory is unlimited, so that it can store all the messages that can be at once in the opportunistic network. However, this is unfeasible in real-life, especially as the network grows ever larger, so a modified Epidemic version exists, where the data memory of a node is limited. Thus, when node A's memory is full and it encounters node B, first it has to drop the oldest messages in its memory, in order to make room for the messages that it will download from node B. This also makes the algorithm somewhat inefficient, since some older messages may be important (e.g., they may be addressed to nodes that A is about to encounter) and some new ones totally irrelevant (e.g., their destinations may be nodes that A will never meet).

3.1.2 Spray-and-Wait

Spray-and-Wait [42] is an improvement to Epidemic that attempts to treat the congestion (and consequently the energy consumption) problem by limiting the total number of messages sent in the network, while keeping a high hit rate. As the name states, the algorithm is split into two phases. The Spray phase assumes that, for each message originating at a source node, a predefined number of copies are transferred to the encountered nodes, in the order that they are seen. The nodes that received the message will then do the same with the nodes that they encounter. Secondly, the Wait phase occurs after the end of the Spray phase (i.e., after the message has been transmitted for the given number of times) and it implies that, if a message's destination has not been encountered yet, then the message will only be relayed when (and if) the destination is encountered. Thus, Spray-and-Wait combines the speed of Epidemic routing with the simplicity of direct transmission. However, unlike the basic Epidemic algorithm, Spray-and-Wait doesn't guarantee the maximum hit rate. Moreover, it still does not take into account any context information in its routing decisions.

3.2 Socially Based Algorithms

Since the initial algorithms proposed for routing and data dissemination in ONs proved to be inefficient for large scale networks, new algorithms were required. Opportunistic networks were considered generally to be formed of mobile devices carried by humans, so human mobility and behavior were studied in detail, in order to find patterns. Thus, it has been shown that users tend to interact more with nodes that they have a strong social connection with [9,13] (i.e. nodes belonging to the same social community). This led to the creation of socially based algorithms, which base their decision of whether two nodes should exchange a message or not on the communities of the two nodes, as well as the communities of the message's source and destination.

3.2.1 Socio-Aware Overlay

The Socio-Aware Overlay algorithm [45] is a data dissemination technique that creates an overlay for an ON with publish/subscribe communication, composed of nodes having high centrality values that have the best visibility in a community. Each of these nodes, called hubs or brokers, represents a community, and thus leverages the communication between nodes pertaining to its community and other nodes. The Socio-Aware Overlay algorithm is based on a publish/subscribe approach, with nodes subscribing to channels that publish data. When two nodes meet, subscriptions and unsubscriptions with the destination of community broker nodes are exchanged, as well as a list of centrality values with a time stamp. When a broker node changes upon calculation of its centrality, the subscription list is transferred to the new broker. When a publication reaches the broker, it is propagated to all other brokers, and then the broker checks its own subscription list. If there are members in its community that must receive the publication, the broker floods the community with the information. Being socially aware, the algorithm has its own community detection method. This method assumes a community structure that is based on a classification of the nodes in an opportunistic network, from the standpoint of another node. A first type of node is one from the same community, having a high number of contacts of long/stable durations. Another type of node is

called a familiar stranger and has a high number of contacts with the current node, but the contact durations are short. There are also stranger nodes, where the contact duration is short and the number of contacts is low, and finally friend nodes, with few contacts, but high contact durations. In order to construct an overlay for publish/subscribe systems, community detection is performed in a decentralized fashion. Thus, each node must detect its own local community. The disadvantage of this method is that broker nodes tend to be congested, given that all data directed to their community must first pass through them. If the members of a community are subscribed to many channels, it would be more suitable to be able to have multiple brokers, in order to increase the efficiency.

3.2.2 BUBBLE Rap

BUBBLE Rap [30] is a routing algorithm for opportunistic networks that uses knowledge about nodes' social communities to deliver messages. It assumes that a mobile device carrier's role in the society is also true in the network. Therefore, the first step performed by BUBBLE Rap is to forward data to more popular nodes than the current node. The second assumption made in BUBBLE Rap is that the communities people form in their social lives are also observed in the network layer, so the second part of the algorithm is to identify the members of the destination community and pass them the message. Thus, a message is bubbled up the hierarchical ranking tree using a global popularity level, until it reaches a node that is in the same community as the destination. Then, the message is bubbled up using a local ranking until it reaches its target. The popularity of a node is given by its betweenness centrality, which is the number of times a node is on the shortest path between two other nodes in the network. Community detection is done using k-CLIQUE [31], which dynamically detects the community of a node by analyzing its contacts with other devices. A distributed version of BUBBLE Rap entitled DiBuBB is also proposed by the authors. It uses distributed k-CLIQUE for community detection, together with a cumulative or single window algorithm for distributed centrality computation. The single window (S-window) algorithm computes centrality as the number of encounters the current node has had in the last time window (chosen usually to be six hours), while the cumulative window (C-window) algorithm counts the number of individual nodes encountered for each time window and then performs an exponential smoothing on the cumulated values. The same issue that can appear at the Socio-Aware Overlay may be present at BUBBLE Rap: popular nodes tend to get congested, because they have to carry messages for many other nodes.

3.2.3 SRSN

The SRSN algorithm [6] is based on the assumption that ad hoc detected communities may miss important aspects of the true organization of an opportunistic network, where, for example, a node might have a strong social link to another node that is encountered rarely. In such a situation, a detected social network might omit this tie and thus yield suboptimal forwarding paths. Therefore, two types of social networks are considered. First of all, there is a detected social network (DSN) as given by a community detection algorithm such as k-CLIQUE, an approach similar to the one taken by BUBBLE Rap. Secondly, the authors also propose a self-reported social network (SRSN) as given by social network links (in this case, Facebook relationships). The algorithm follows a

few steps: nodes generate data, carry it around the network and, when they encounter another node, they only exchange information if the two nodes are in the same network (either DSN of SRSN). Therefore, there are two versions of this algorithm: one that uses the DSN, and another one that uses SRSN. Through extensive experiments, the authors show that using SRSN information instead of DSN decreases the delivery cost and produces comparable delivery ratio. This happens because the two social networks differ in terms of structural and role equivalence, with the better approximation being obtained through the SRSN. The results presented by the SRSN algorithm are relevant in terms of highlighting the importance of using readily available information (such as Facebook, Google+, Twitter, or LinkedIn social relationships) for approximating a user's social relationships. However, in situations where the social network does not correctly approximate the network's behavior, or where social network information is not available, such an algorithm can't be used.

3.2.4 ContentPlace

ContentPlace [7] deals with data dissemination in resource-constrained ONs, by making content available in regions where interested users are present, without overusing available resources. To optimize content availability, it exploits learned information about users' social relationships to decide where to place user data. The design of ContentPlace is based on two assumptions: users can be grouped together logically, according to the type of content they are interested in, and their movement is driven by social relationships. When a node encounters another node, it decides what information seen on the other node should be replicated locally. Thus, ContentPlace defines a utility function by means of which each node can associate a utility value to any data object. When a node encounters another peer, it selects the set of data objects that maximizes the local utility of its cache. Due to performance issues, when two nodes meet, they do not advertise all information about their data objects, but instead they exchange a summary of data objects in their caches. Finally, the data exchange is accomplished when a user receives a data object it is subscribed to when it is found in an encountered node's cache. To have a suitable representation of users' social behavior, an approach that is similar to the caveman model is used, that has a community structure which assumes that users are grouped into home communities, while at the same time having relationships in acquainted communities. The utility is a weighted sum of one component for each community its user has relationships with. Community detection is done using k-CLIQUE. By using weights based on the social aspect of opportunistic networking, ContentPlace offers the possibility of defining different policies. There are five policies defined: Most Frequently Visited (MFV), Most Likely Next (MLN), Future (F), Present (P), and Uniform Social (US). These policies allow the network manager to change the behavior of the nodes, according to the configuration of the network.

3.3 History-and Prediction-Based Algorithms

There are some situations where social information is not present and algorithms that are able to detect communities are too costly in terms of computing power. Because of such cases (and because social information does not always lead to good approximations of contact behavior), a new type of algorithms has appeared. These algorithms

base their routing decisions on the history of contacts between nodes. If a node A has encountered a node B many times in the recent past, it is assumed that it will encounter it again in the near future, because the two nodes have similar paths. Moreover, based on the shapes of past contacts, various types of distributions and approximations have been employed to predict the future behavior of ON nodes and perform optimal routing decisions.

3.3.1 PROPHET

PROPHET [35] is a prediction-based routing algorithm for ONs which performs probabilistic routing by establishing a metric called delivery predictability (P) at every node A for a known destination B. This probability signifies A's chance to successfully deliver a message to B. When two PROPHET nodes meet, they exchange summary vectors which (among other information) contain the delivery predictability P. When a node receives this information, it updates its internal delivery predictability vector (whose size is equal to the total number of nodes in the ON), and then decides which messages to request from the other node based on the forwarding strategy used. There are three steps performed when computing delivery predictability values. First, whenever a node is encountered, the local value of the metric is updated, which leads to a higher P for nodes that are encountered more often. Secondly, since nodes that are not in contact for long periods of time are not very likely to be good forwarders toward each other, the delivery predictability must age, thus being reduced with the passage of time. The aging process is based on an aging constant and a given unit of time. Finally, the delivery predictability also has a transitive property, based on the fact that, if two nodes A and B meet each other often, and node A also has many encounters with a node C, then C is also a good forwarding node for A. Based on the scaling constant, the transitivity property also impacts the computation of the delivery predictability. The forwarding strategy chosen by the authors is a simple one, and implies that, when two nodes A and B meet, if the delivery probability of the destination of a message at B is higher for A, then the message is transferred (and the other way around as well). The main caveat of this algorithm is that, since nodes are not being split into communities, there exists the risk of flooding a popular node (such as a professor in an academic environment, which interacts with students from various study years). This can be avoided by redirecting a part of the messages destined for a such a node to other less popular nodes.

3.3.2 RANK

Hui and Crowcroft [29] study the impact of predictable human interactions on forwarding in PSNs. By applying vertex similarity on a dataset extracted from mobility traces, they observe that adaptive forwarding algorithms can be built by using the history of past encounters. Furthermore, the authors design a distributed forwarding algorithm based on node centrality and show that it is efficient in terms of hit rate and delivery latency. This greedy algorithm is entitled RANK, and (similarly to BUBBLE Rap) it uses popular nodes to disseminate data. The popularity of a node is quantified by the Freeman betweenness centrality, which is defined as the number of times a node falls on the shortest path of other nodes. The authors assume that each node knows its own centrality and the centrality of the nodes it encounters, but not of the other nodes in the network, so it cannot know the highest centrality in the system. Therefore, the greedy

algorithm pushes traffic on all paths to nodes that have a higher centrality than the current node, until the destination is reached or the messages expire. Since knowing the individual centrality for each node at any point in time is complicated, the authors propose analyzing the past activity of a node to see if it was a good carrier in the past, and then use this information for future forwarding. Therefore, they analyze how well the past centrality can predict the future centrality for a given node, and for this reason they extract three consecutive three-week sessions for a mobility trace and run a set of greedy RANK emulations on the last two data sessions, using centrality values from the first session. The test results show that human mobility is predictable to a certain degree and that past contact information can successfully be used to approximate the future behavior of a node in the ON. However, one of the main limitations of the RANK algorithm is that it only focuses on a day-to-day analysis, whereas a finer-grained predictability may prove to be more useful for messages that have a lower tolerance for delays. Another important caveat of the algorithm is that, although it uses prediction of future node behavior, it does not consider the ON nodes as belonging to communities, which may lead to congestion at the nodes that are most popular in terms of centrality.

3.3.3 dLife

dLife [37] is an opportunistic network routing algorithm that is able to capture the dynamic represented by time-evolving social ties between pairs of nodes. The authors highlight the fact that user behavior is dynamic, the network itself evolves, meaning that network ties are created and broken constantly. This is why dLife focuses on the different behavior users have in different daily periods of time, instead of estimating their behavior per day. The dynamics of social structures are represented as a weighted contact graph, where the weights are used to express how long a pair of nodes is in contact over different periods of time. There are two complementary utility functions employed by dLife: the Time-Evolving Contact Duration (TECD), which is the evolution of social interaction among pairs of users in the same daily interval over consecutive days, and the TECD Importance ($TECD_i$), which is the evolution of a user's importance, based on its node degree and social strength toward its neighbors, in different periods of time. TECD is used to forward messages to nodes that have a stronger social relationship with the destination than the current carrier. Each node computes the average of its contact duration with other nodes during the same set of daily time periods over consecutive days. If the carrier and the encountered node have no social information toward the destination, forwarding is done based on $TECD_i$, where the encountered node gets a message if it has a greater importance than the carrier. The authors also propose a community-based version of dLife, entitled dLifeComm, where the social communities are computed similarly to BUBBLE Rap (i.e., using k-CLIQUE), but the decision whether to forward to a node is done based on TECD and $TECD_i$, thus changing over time. dLife offers the advantage of using both the history of contacts, as well as social information, in performing routing decisions.

4 Potential Solutions

In this section, we present a couple of alternatives to the existing solutions shown in Section 3 and highlight the improvements they bring.

4.1 SPRINT

SPRINT [14] is a novel ON point-to-point routing algorithm that takes advantage of both social knowledge, as well as contact prediction, when making decisions. It uses information about the nodes from the contact history and from existing self-reported social networks. Moreover, it includes a Poisson-based prediction of a node's future behavior. Through extensive experiments, it has been shown that SPRINT performs better that existing socially aware opportunistic routing solutions in terms of hit rate, latency, delivery cost[6], and hop count[7]. This section presents the motivations behind SPRINT and the functionality of the algorithm.

4.1.1 Social Knowledge in Opportunistic Networks

The addition of social information to SPRINT is motivated by the results presented in [13] and [15]. There, an academic environment is analyzed in terms of contact and inter-contact times, as well as the relationships between the social connection strengths and number and duration of contacts between two nodes. First, it is shown that the self-reported social network information (i.e., Facebook friend data gathered from the participants in the network) is a better approximation of the contact behavior of a node than the data obtained by a community-detection algorithm such as *k*-CLIQUE. By taking this information into account, four modified BUBBLE Rap versions are proposed, which use social network information instead of *k*-CLIQUE data.

The first such BUBBLE Rap version is called Social, and it performs the decision of whether to use the local or the global community (as shown in Section 3.2.2) based on social network information, instead of *k*-CLIQUE data. Thus, the communities assumed by BUBBLE Rap are formed through self-reported social networks knowledge, instead of using community detection algorithms. The second method (entitled Max) computes a node's centrality (and thus its importance in its own community) as the maximum between the centrality as reported by the C-window algorithm used by BUBBLE Rap, and a node's popularity in terms of Facebook friends. The other two BUBBLE Rap improvements (Popularity and Popularity Squared) use a weighted sum between the C-window centrality and a node's popularity to compute the final centrality value (Popularity uses a regular sum, whereas Popularity Squared employs a squared sum between the two values).

These four proposed BUBBLE Rap enhancements are tested both on an academic trace, where social connections are expected to exist between students attending the same classes, as well as on a different trace taken in and around the town of St. Andrews [6]. It is shown that all four socially based BUBBLE Rap versions outperform the base implementation for both traces in terms of hit rate. Moreover, the best results are obtained by the Popularity version.

4.1.2 Predicting Opportunistic Nodes' Behavior

As stated in Section 2.2, an important challenge in mobile networks is knowing when and to which node should a message be passed, in order for it to reach its destination

[6] The ratio between the total number of messages exchanged in the network and the number of generated messages.

[7] The number of nodes that carried a message until its destination on the shortest path.

(and do it as fast as possible). Therefore, it would be important if we were able to predict the future behavior of a node in such a network, in regard to its encounters and contact durations. Such a method is proposed in [12], by approximating the time series of a node's contacts as a Poisson distribution.

ON nodes, as previously stated in this chapter, are generally mobile devices that belong to humans, and if there's one thing certain about people, it is that they are creatures of habit. They follow similar daily patterns, from home to work or school, where they spend a generally fixed amount of time, after which they return home. Similarly, in weekends they tend to go to the same places and visit the same locations. This is also valid on a smaller scale, as shown by the analysis in [12], namely an academic scenario where the nodes are the students and professors from a faculty. They have a fixed daily schedule and interact with each other at fixed times in a day, namely when they attend classes. Through analysis of a mobility trace taken in such an academic environment, it has been shown that a node's behavior in an opportunistic network in terms of number of contacts per time unit can be approximated as a Poisson distribution. The shape of the distribution is proven to apply to the mobility trace analyzed by performing a chi-squared test, with only 2.49% of all the Poisson hypotheses rejected. Moreover, the paper shows that, by removing the final week from the trace and attempting to predict each node's number of contacts using a Poisson distribution, a percentage of 98.24% correct predictions is obtained. Similar results are also achieved for the St. Andrews trace mentioned above. We refer the reader to [12] for the complete set of tests and results.

4.1.3 SPRINT Algorithm

The SPRINT algorithm [14] combines socially aware routing (both learned and offline social information) with node behavior prediction, in order to improve the performance of ON routing.

The behavior of SPRINT when two nodes running it get in contact is very similar to ContentPlace's behavior, as shown in Section 3.2.4: each node computes a utility value for both its messages, as well as the ones belonging to the encountered node, and then attempts to maximize its data cache by selecting the messages with the highest utility values. The novel part of SPRINT is its utility function and the way it uses both social, as well as prediction information, to compute the importance of a message. The formula used by a node A to compute the utility of a message M is shown below. w_1 and w_2 are weight values which follow the conditions that $w_1 + w_2 = 1$ and $w_1 > w_2$. U_1 and U_2 are individual utility components.

$$u(M,A) = w_1 * U_1(M,A) + w_2 * U_2(M,A)$$

$$U_1(M,A) = freshness(M) + p(M,A) * (1 - \frac{enc(M,A)}{24})$$

$$U_2(M,A) = c_e(M,A) * \frac{s_n(M) + hop(M) + pop(A) + t(M,A)}{4}$$

The $freshness(M)$ component of U_1 favors new messages, being positive if the message has been created less than a day ago, and 0 otherwise. $p(M,A)$ is the probability

of node A being able to deliver a message M closer to its destination, and is based on predicting a node's behavior, combined with the idea that a node has a higher chance of interacting with nodes it is socially connected with and/or has encountered before. It is computed based on the knowledge that the node contacts follow a Poisson distribution. The first step is to count how many times node A encountered each of the other nodes in the network. If a node has been previously met in the same day of the week or in the same 2h interval as the current time, the total encounters value is increased by 1. For the nodes encountered in the past that are in the same social community as node A, the total number of contacts is doubled. Then, the probabilities of encountering nodes based on past contacts are computed by performing a ratio between the number of encounters per node and the total number of encounters. The next step consists of computing the number of encounters N that node A will have for each of the next 24 hours by using the Poisson distribution probabilities and choosing the value with the highest probability as N. The first N nodes are then picked as potential future contacts for each of the next 24 hours (sorted by probability), and for the rest of them $p(M,A)$ is set to 0. U_1 also uses $enc(M,A)$, which is the time (in hours) until the destination of message M will be met by A according to the probabilities previously computed. If the destination will never be encountered, then $enc(M,A)$ is set to 24 (so the product is 0).

The second component of the utility function is U_2. $c_e(M,A)$ is set to 1 if node A is in the same community as the destination of message M or if it will encounter a node that has a social relationship with M, and 0 otherwise. The prediction information computed for U_1 is used to analyze the potential future encounters of a node. The $s_n(M)$ component is set to 1 (and 0 otherwise) if the source and destination of M do not have a social connection. $hop(M)$ represents the normalized number of nodes that M has visited, $pop(A)$ is the popularity value of A according to its social network information (i.e., number of Facebook friends in the opportunistic network), and finally $t(M,A)$ is the total time spent by node A in contact with M's destination.

SPRINT is compared to BUBBLE Rap and it is shown that it performs better in terms of hit rate, delivery latency, hop count, and delivery cost for three mobility traces and one synthetic mobility model simulation. The complete results, along with their analysis, are presented in detail in [14]. The algorithm's main advantage over other solutions is that it does not rely solely on one method for deciding the next hop. It combines social information, from both offline and online sources, with ad hoc prediction mechanisms (which can be switched on-the-fly, according to the characteristics of the ON), to offer a more complete view of a node's behavior.

4.2 SENSE

SENSE [16] is a collaborative selfish node detection and incentive mechanism for opportunistic networks that is not only able to detect the selfish nodes in an ON, but also has the possibility of improving the network's performance by incentivising the participating nodes into carrying data for other nodes. Altruism is an important component of ONs, since nodes must rely on each other for a successful transmission of their intended messages. Thus, nodes refusing to participate in the routing process are punished by the algorithm, and therefore, have no way to get their messages to be delivered, unless they accept to help other nodes route their data as well.

4.2.1 Selfishness and Altruism in Opportunistic Networks

Most routing and dissemination algorithms proposed so far generally assume that the nodes in an opportunistic network are willing to participate in the routing process at all times. However, in real-life scenarios this is not necessarily true, since a node may be selfish toward a subset of nodes, and unselfish for the rest. Several reasons for this selfishness exist, such as a node being low on resources (battery life, memory, CPU, network, bandwidth, etc.) and trying to save them for future use, fear of malicious data from unknown users, or even lack of interest in helping the nodes from a different social community. The existence of selfish nodes in an opportunistic network might lead to messages having high delays or never being delivered at all. Thus, these nodes should be detected and avoided when routing. Furthermore, incentive mechanisms should reward nodes when they actively take part in the network and punish them when they do not.

There are several altruism models that specify how selfish nodes are spread in the network and how they behave toward the nodes they encounter. SENSE uses the community-biased model, which assumes that people in a community have greater incentives to carry messages for other members of the same community. In this case, altruism is modeled using an intra and an inter-community altruism level. This is one of the most realistic altruism models available, since it is a good approximation of an opportunistic network where the nodes are mobile devices carried by humans that interact based on social relationships. However, altruism values should also be distributed inside a community (i.e., not all nodes in the same community should have the same altruistic values toward each other), using a uniform or normal distribution.

4.2.2 SENSE Algorithm

Each SENSE [16] node has a four-section data memory. First, there is the list of messages that the node has generated in the course of time G. Secondly, each node has a list of messages that it stores, carries, and forwards for other nodes C. Additionally, each node has another two sections of data memory that contain information regarding past transfers: a list of past forwards O and a list of past receives I. O contains information regarding past message forward operations performed either by the current node, or by other nodes. The list of past receives I contains information regarding past message receive operations.

When two nodes A and B running SENSE meet, they each compute an altruism value toward the other node and, based on that value, decide if they will help the other node. If the two nodes decide that they are unselfish toward one another, they exchange I and O and update them with the new information. This way, a node can have a more informed view of the behavior of various nodes in the network, through gossiping.

After two nodes decide to be altruistic toward one another and they finish exchanging knowledge about past encounters, each of them advertises its own specific information, such as battery level and metadata about the messages it carries. Based on the lists of past encounters I and O, each node computes a perceived altruism value for the other node with regard to the messages stored in its own data memory (in other words, it computes how willing the encountered node is to forward a certain type of message). If this value is within certain thresholds, the communication continues and the desired opportunistic routing or dissemination algorithm is applied. If (for example) node B's computed altruism is not within the given limits for any of A's stored messages, then it

is considered selfish by node A, so A does not send it messages for routing and does not accept messages from B, either. Node A then notifies B that it considers it selfish, so B would not end up considering node A selfish. This also functions as an incentive mechanism, because if a node wants its messages to be routed by other nodes, it should not be selfish toward them. Therefore, every time a node is notified that it is selfish in regard to a certain message, it increases its altruism value. If there is a social connection between the selfish node and the source of the message, the inter-community altruism is increased. Otherwise, the intra-community altruism value grows.

The formula for computing altruism values for a node N and a message M based on the list of past forwards O and on the list of past receives I is the following:

$$altruism(N,m) = \sum_{o \in O, i \in I, o.m=i.m}^{N.id=o.d, N.id=i.s} type(m,o.m) * thr(o.b)$$

A past encounter x has a field $x.m$ which specifies the message that was sent or received, $x.s$ is the source of the transfer, $x.d$ is the destination, and $x.b$ is the battery level of the source. *type* is a function that returns 1 if the types of the two messages received as parameters are the same (in terms of communities, priorities, etc.), and 0 otherwise, while *thr* returns 1 if the value received as parameter is higher than a preset threshold, and 0 if it's not the case. Thus, the function counts how many messages of the same type as M have been forwarded with the help of node N, when N's battery was at an acceptable level.

Test results show that SENSE can help improve opportunistic network performance (with metrics such as hit rate, delivery latency, hop count, and delivery cost) when selfish nodes exist. It is demonstrated that SENSE outperforms a scenario where selfish nodes are present, but no selfishness detection and incentive mechanism is available. Moreover, it even performs better than existing similar algorithms, such as IRONMAN [5]. It is also shown that SENSE can successfully differentiate between a node being selfish on purpose, and a node not being able to deliver messages due to low-battery power. For the full set of tests and results, we refer to reader to [16]. The main advantage of SENSE is not only that it can detect and avoid selfish nodes, but also that it can limit the total number of messages sent in the opportunistic network by carefully selecting a message's destination, based on the social connection and history of routing.

5 Future Trends

One of the main limitations of research in this area, so far, is that it has mostly focused on point-to-point communication. However, we believe that the future of opportunistic networks is heading toward data dissemination, where communication is done based on a publish/subscribe paradigm. This is why we are expecting a focus on data dissemination instead of point-to-point routing in the near future. This includes moving toward information-centric networks (ICNs) and the Internet of Things (IoT).

An ICN is a novel method of making the Internet more data-oriented and content-centric [21] and is basically a global-scale version of the publish/subscribe paradigm. The focus changes from referring to data by its location (and an IP address) to requesting Named Data Objects (NDOs) instead. When an ICN network element receives a

request for content, it can respond with the content directly if it has the data cached, or it can request it from its peers otherwise. This way, an end user is not concerned with the location of an object, only with its actual name, thus being able to receive it from any number of hosts. Mobile devices play an important role in ICNs, since they may be used to cache data as closely as possible to interested users, based on context information. Therefore, efficient opportunistic routing and dissemination algorithms have to be employed in order to move the data accordingly and replicate it as needed.

The Internet of Things [23] aims to improve social connectivity in physical communities by leveraging information detected by mobile devices. It assumes a number of such devices being able to communicate between each other to gather context data, which is then used to make automated decisions. The deployment of IoT generally has three steps. The first one is getting more devices onto the network, the second step is making them rely on each other, coordinating their actions for simple tasks without human intervention, and the final step is to understand these devices as a single system that needs to be programmed. The more devices will be connected, the more important will the role of the routing and dissemination protocols be.

It is estimated that IoT will have to accommodate over 50,000 billion objects of very diverse types by 2020 [40]. Standardization and interoperability will thus be absolute necessities for interfacing them with the Internet. New media access techniques, communication protocols, and sustainable standards will need to be developed to make Things communicate with each other and with people. One approach would be the encapsulation of smart wireless identifiable devices and embedded devices in Web services. We can also consider the importance of enhancing the quality of service aspects like response time, resource consumption, throughput, availability, and reliability. The discovery and use of knowledge about services availability and of publish/subscribe/notify mechanisms would also contribute to enhancing the management of complex Thing structures.

Because of the fast increase of mobile data traffic volume being generated by bandwidth-hungry smartphone applications, cellular operators are forced to explore various possibilities to offload data traffic away from their core networks. 3G cellular networks are already overloaded with data traffic generated by smartphone applications (e.g., mobile, TV). With the advent of IoT, the potentially huge number of Things will not be easily incorporated by today's communication protocols and/or Internet architecture. Mobile data offloading may relieve the problem, by using complementary communication technologies (considering the increasing capacity of WiFi), to deliver traffic originally planned for transmission over cellular networks. Here, opportunistic networks can find quick benefits.

Again related to IoT, new services shall be available for persistent distributed knowledge storing and sharing, and new computational resources shall be used for the execution of complicated tasks. Actual forecasts indicate that in 2015 more than 220 Exabytes of data will be stored [40]. At the same time, optimal distribution of tasks between smart objects with high capabilities and the IoT infrastructure shall be found. New mechanisms and protocols will be needed for privacy and security issues at all IoT levels including the infrastructure. Solutions for stronger security could be based on models employing the context-aware capability of Things. New methods are required for energy saving and energy-efficient and self-sustainable systems. Researchers will

look for new power-efficient platforms and technologies and will explore the ability of smart objects to harvest energy from their surroundings.

The large variety of technologies and designs used in the production of Things is a main concern when considering the interoperability. One solution is the adoption of standards for Things intercommunication. Adding self-configuration and self-management properties could be necessary to allow Things to interoperate and, in addition, integrate within the surrounding operational environment. This approach is superior to the centralized management, which cannot respond to difficulties induced by the dimensions, dynamicity, and complexity of the Internet of Things. The autonomic behavior is important at the operational level as well. Letting autonomic Things react to events generated by context changes facilitates the construction and structuring of large environments that support the Internet of Things. Special requirements come from the scarcity of Things' resources, and are concerned with power consumption. New methods of efficient management of power consumption are needed and could apply at different levels, from the architecture level of Things to the level of the network routing. They could substantially contribute to lowering the cost of Things, which is essential for the rapid expansion of the Internet of Things.

Some issues come from the distributed nature of the environment in which different operations and decisions are based on the collaboration of Things. One issue is how Things converge on a solution and how the quality of the solution can be evaluated. Another issue is how to protect against faulty Things, including those exhibiting malicious behavior. Finally, the way Things can cope with security issues to preserve confidentiality, privacy, integrity, and availability are of high interest. For all these, examples of mechanisms designed to cope with such problems by actively using any communication opportunity were presented throughout the chapter.

6 Conclusions

In this chapter, we have provided the definition of opportunistic networks and have shown the challenges facing the deployment of such networks in real-life. However, we have also presented several use cases where ONs have been successfully deployed, and other areas where interesting and valid propositions have been presented. This leads us to believe that opportunistic networks have a good applicability in real-life, especially, if the algorithms and solutions keep evolving, as they have been doing in the past few years.

We have also presented several ON routing and dissemination algorithms. For each of them, we have shown that they have both strengths, as well as weaknesses. Some of these algorithms are suitable for a certain type of situation, other are better for different scenarios. There is no single best algorithm, and this is because opportunistic networks are so varied and can range from large and dense networks with thousands of participants, to small and sparse networks that must make the most of any contacts between nodes. This is why the research area of ONs is so vast and keeps evolving constantly, with the proposed solutions becoming better and better. Moreover, we have shown how some of the issues in opportunistic networking might be fixed by leveraging

social networks, node behavior prediction and selfish node detection, and incentive mechanisms. We concluded our presentation by showing that the future trends in the area of mobile networking are veering toward data dissemination, through information-centric networks and the Internet of Things.

Acknowledgments. This work was partially supported by the project "ERRIC - Empowering Romanian Research on Intelligent Information Technologies/FP7-REGPOT-2010-1", ID: 264207. The work has been cofounded by the Sectoral Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labour, Family, and Social Protection through the Financial Agreement POSDRU/89/1.5/S/62557.

References

1. Akyildiz, I.F., Akan, Ö.B., Chen, C., Fang, J., Su, W.: InterPlaNetary Internet: state-of-the-art and research challenges. Computer Networks Journal 43(2), 75–112 (2003)
2. Arnaboldi, V., Conti, M., Delmastro, F., Minutiello, G., Ricci, L.: DroidOppPathFinder: A context and social-aware path recommender system based on opportunistic sensing. In: Proceedings of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2012 (2012)
3. Athanasopoulos, D., Zarras, A.V., Issarny, V., Pitoura, E., Vassiliadis, P.: CoWSAMI: Interface-aware context gathering in ambient intelligence environments. Pervasive and Mobile Computing Journal 4(3), 360–389 (2008)
4. Bharathidasan, A., An, V., Ponduru, S.: Sensor networks: An overview. Technical report, Department of Computer Science, University of California, Davis (2002)
5. Bigwood, G., Henderson, T.: In: Proceedings of IEEE Third International Conference on Privacy, Security, Risk and Trust, PASSAT 2011 (2011)
6. Bigwood, G., Rehunathan, D., Bateman, M., Henderson, T., Bhatti, S.: Exploiting self-reported social networks for routing in ubiquitous computing environments. In: Proceedings of the 2008 IEEE International Conference on Wireless & Mobile Computing, Networking & Communication, WIMOB 2008, pp. 484–489. IEEE Computer Society, Washington, DC (2008)
7. Boldrini, C., Conti, M., Passarella, A.: Exploiting users' social relations to forward data in opportunistic networks: The HiBOp solution. Pervasive and Mobile Computing Journal 4, 633–657 (2008)
8. Bruno, R., Conti, M., Passarella, A.: Opportunistic networking overlays for ICT services in crisis management. In: Proceedings of the International Conference on Information Systems for Crisis Response and Management, ISCRAM 2008 (2008)
9. Chaintreau, A., Hui, P., Crowcroft, J., Diot, C., Gass, R., Scott, J.: Pocket switched networks: Real-world mobility and its consequences for opportunistic forwarding. Technical report, University of Cambridge Computer Lab (2005)
10. Chourabi, H., Nam, T., Walker, S., Gil-Garcia, J.R., Mellouli, S., Nahon, K., Pardo, T.A., Scholl, H.J.: Understanding smart cities: An integrative framework. In: Proceedings of the 45th Hawaii International Conference on System Science, HICSS 2012, pp. 2289–2297 (2012)

11. Ciobanu, R., Dobre, C.: Data dissemination in opportunistic networks. In: Proceedings of 18th International Conference on Control Systems and Computer Science, CSCS-18, pp. 529–536. Politehnica Press (2012)
12. Ciobanu, R.I., Dobre, C.: Predicting encounters in opportunistic networks. In: Proceedings of the 1st ACM Workshop on High Performance Mobile Opportunistic Systems, HP-MOSys 2012, pp. 9–14. ACM, New York (2012)
13. Ciobanu, R.I., Dobre, C., Cristea, V.: Social aspects to support opportunistic networks in an academic environment. In: Li, X.-Y., Papavassiliou, S., Ruehrup, S. (eds.) ADHOC-NOW 2012. LNCS, vol. 7363, pp. 69–82. Springer, Heidelberg (2012)
14. Ciobanu, R.I., Dobre, C., Cristea, V.: SPRINT: Social prediction-based opportunistic routing. In: Proceedings of IEEE 14th International Symposium and Workshops on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2013, pp. 1–7 (2013)
15. Ciobanu, R.-I., Dobre, C., Cristea, V., Al-Jumeily, D.: Social aspects for opportunistic communication. In: Proceedings of the 11th International Symposium on Parallel and Distributed Computing, ISPDC 2012, pp. 251–258 (2012)
16. Ciobanu, R.-I., Dobre, C., Dascălu, M., Trăusan-Matu, S., Cristea, V.: Collaborative selfish node detection with an incentive mechanism for opportunistic networks. In: Proceedings of IFIP/IEEE International Symposium on Integrated Network Management, IM 2013, pp. 1161–1166 (2013)
17. Conti, M., Giordano, S., May, M., Passarella, A.: From opportunistic networks to opportunistic computing. Communications Magazine 48(9), 126–139 (2010)
18. Desta, M.S., Hyytiä, E., Ott, J., Kangasharju, J.: Characterizing content sharing properties for mobile users in open city squares. In: Proceedings of the 10th Annual Conference on Wireless on-demand Network Systems and Services, WONS 2013, pp. 147–154 (2013)
19. Dobre, C., Manea, F., Cristea, V.: CAPIM: A context-aware platform using integrated mobile services. In: Proceedings of IEEE International Conference on Intelligent Computer Communication and Processing, ICCP 2011, pp. 533–540 (2011)
20. Doria, A., Uden, M., Pandey, D.P.: Providing connectivity to the Saami nomadic community. In: Proceedings of the 2nd International Conference on Open Collaborative Design for Sustainable Innovation, DYD 2002, Bangalore, India (December 2002)
21. Ghodsi, A., Shenker, S., Koponen, T., Singla, A., Raghavan, B., Wilcox, J.: Information-centric networking: seeing the forest for the trees. In: Proceedings of the 10th ACM Workshop on Hot Topics in Networks, HotNets-X, pp. 1:1–1:6. ACM, New York (2011)
22. Gu, T., Pung, H.K., Zhang, D.Q.: A service-oriented middleware for building context-aware services. Journal of Network and Computer Applications 28(1), 1–18 (2005)
23. Guo, B., Yu, Z., Zhou, X., Zhang, D.: Opportunistic IoT: Exploring the social side of the Internet of Things. In: Proceedings of IEEE 16th International Conference on Computer Supported Cooperative Work in Design, CSCWD 2012, pp. 925–929 (2012)
24. Guo, S., Derakhshani, M., Falaki, M.H., Ismail, U., Luk, R., Oliver, E.A., Ur Rahman, S., Seth, A., Zaharia, M.A., Keshav, S.: Design and implementation of the KioskNet system. Computer Networks Journal 55(1), 264–281 (2011)
25. Haddadi, H., Hui, P., Henderson, T., Brown, I.: Targeted advertising on the handset: privacy and security challenges. Human-Computer Interaction Series. Springer (July 2011)
26. Heinemann, A., Straub, T.: Opportunistic networks as an enabling technology for mobile word-of-mouth advertising. In: Pousttchi, K., Wiedmann, D.G. (eds.) Handbook of Research on Mobile Marketing Management, pp. 236–254. Business Scrience Reference, PA (2010)

27. Henricksen, K., Robinson, R.: A survey of middleware for sensor networks: state-of-the-art and future directions. In: Proceedings of the International Workshop on Middleware for Sensor Networks, MidSens 2006, pp. 60–65. ACM, New York (2006)
28. Hui, P., Chaintreau, A., Scott, J., Gass, R., Crowcroft, J., Diot, C.: Pocket switched networks and human mobility in conference environments. In: Proceedings of the ACM SIGCOMM Workshop on Delay-Tolerant Networking, WDTN 2005, pp. 244–251. ACM, New York (2005)
29. Hui, P., Crowcroft, J.: Predictability of human mobility and its impact on forwarding. In: Proceedings of the Third International Conference on Communications and Networking in China, ChinaCom 2008, pp. 543–547 (2008)
30. Hui, P., Crowcroft, J., Yoneki, E.: BUBBLE Rap: social-based forwarding in delay tolerant networks. In: Proceedings of the 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2008, pp. 241–250. ACM, New York (2008)
31. Hui, P., Yoneki, E., Chan, S.Y., Crowcroft, J.: Distributed community detection in delay tolerant networks. In: Proceedings of 2nd ACM/IEEE International Workshop on Mobility in the Evolving Internet Architecture, MobiArch 2007, pp. 7:1–7:8. ACM, New York (2007)
32. Juang, P., Oki, H., Wang, Y., Martonosi, M., Peh, L.S., Rubenstein, D.: Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet. SIGOPS Operating Systems Review 36(5), 96–107 (2002)
33. Le, V.-D., Scholten, H., Havinga, P.: Unified routing for data dissemination in smart city networks. In: Proceedings of the 3rd International Conference on the Internet of Things, IOT 2012, pp. 175–182. IEEE Press, USA (2012)
34. Lilien, L., Gupta, A., Yang, Z.: Opportunistic networks for emergency applications and their standard implementation framework. In: Proceedings of IEEE International Performance, Computing, and Communications Conference, IPCCC 2007, pp. 588–593 (2007)
35. Lindgren, A., Doria, A., Schelén, O.: Probabilistic routing in intermittently connected networks. SIGMOBILE Mobile Computing and Communications Review 7(3), 19–20 (2003)
36. Martín-Campillo, A., Martí, R., Yoneki, E., Crowcroft, J.: Electronic triage tag and opportunistic networks in disasters. In: Proceedings of the Special Workshop on Internet and Disasters, SWID 2011, pp. 6:1–6:10. ACM, New York (2011)
37. Moreira, W., Mendes, P., Sargento, S.: Opportunistic routing based on daily routines. In: IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2012, pp. 1–6 (2012)
38. Pelusi, L., Passarella, A., Conti, M.: Opportunistic networking: data forwarding in disconnected mobile ad hoc networks. Communications Magazine 44(11), 134–141 (2006)
39. Pentland, A., Fletcher, R., Hasson, A.: DakNet: Rethinking connectivity in developing nations. Computer Journal 37(1), 78–83 (2004)
40. INFSO D.4 Networked Enterprise & RFID, INFSO G.2 Micro & Nanosystems, and Working Group RFID of the ETP EPoSS. Internet of Things in 2020. Roadmap for the future (2009), `http://www.caba.org/resources/Documents/IS-2008-93.pdf` (accessed December 20, 2013)
41. Small, T., Haas, Z.J.: The shared wireless infostation model: a new ad hoc networking paradigm (or where there is a whale, there is a way). In: Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2003, pp. 233–244. ACM, New York (2003)
42. Spyropoulos, T., Psounis, K., Raghavendra, C.S.: Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In: Proceedings of the ACM SIGCOMM Workshop on Delay-Tolerant Networking, WDTN 2005, pp. 252–259. ACM, New York (2005)

43. Thilakarathna, K., Viana, A.C., Seneviratne, A., Petander, H.: The power of hood friendship for opportunistic content dissemination in mobile social networks. Technical report, INRIA, Saclay, France (2012)
44. Vahdat, A., Becker, D.: Epidemic Routing for Partially-Connected Ad Hoc Networks. Technical report, Duke University (April 2000)
45. Yoneki, E., Hui, P., Chan, S., Crowcroft, J.: A socio-aware overlay for publish/subscribe communication in delay tolerant networks. In: Proceedings of the 10th ACM Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems, MSWiM 2007, pp. 225–234. ACM, New York (2007)

Decision Tree Induction Methods and Their Application to Big Data

Petra Perner

Institute of Computer Vision and Applied Computer Sciences,
Kohlenstr. 2, 04251 Leipzig, Germany
pperner@ibai-institut.de

Abstract. Data mining methods are widely used across many disciplines to identify patterns, rules, or associations among huge volumes of data. While in the past mostly black box methods, such as neural nets and support vector machines, have been heavily used for the prediction of pattern, classes, or events, methods that have explanation capability such as decision tree induction methods are seldom preferred. Therefore, we give in this chapter an introduction to decision tree induction. The basic principle, the advantageous properties of decision tree induction methods, and a description of the representation of decision trees so that a user can understand and describe the tree in a common way is given first. The overall decision tree induction algorithm is explained as well as different methods for the most important functions of a decision tree induction algorithm, such as attribute selection, attribute discretization, and pruning, developed by us and others. We explain how the learnt model can be fitted to the expert´s knowledge and how the classification performance can be improved. The problem of feature subset selection by decision tree induction is described. The quality of the learnt model is not only to be checked based on the overall accuracy, but also more specific measure are explained that describe the performance of the model in more detail. We present a new quantitative measures that can describe changes in the structure of a tree in order to help the expert to interpret the differences of two learnt trees from the same domain. Finally, we summarize our chapter and give an outlook.

1 Introduction

Data mining methods are widely used across many disciplines to identify patterns, rules, or associations among huge volumes of data. While in the past mostly black box methods, such as neural nets and support vector machines, have been heavily used for the prediction of pattern, classes, or events, methods that have explanation capability such as decision tree induction methods are seldom preferred. Besides, it is very important to understand the classification result not only in medical application more often but also in technical domains. Nowadays, data mining methods with explanation capability are more heavily used across disciplines after more work on advantages and disadvantages of these methods has been done.

Decision tree induction is one of the methods that have explanation capability. Their advantages are easy use and fast processing of the results. Decision tree induction methods can easily learn a decision tree without heavy user interaction while in

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_3

neural nets a lot of time is spent on training the net. Cross-validation methods can be applied to decision tree induction methods while not for neural nets. These methods ensure that the calculated error rate comes close to the true error rate. In most of the domains such as medicine, marketing or nowadays even technical domains the explanation capability, easy use, and the fastness in model building are one of the most preferred properties of a data mining method.

There are several decision tree induction algorithms known. They differ in the way they select the most important attributes for the construction the decision tree, if they can deal with numerical or/and symbolical attributes, and how they reduce noise in the tree by pruning. A basic understanding of the way a decision tree is built is necessary in order to select the right method for the actual problem and in order to interpret the results of a decision tree.

In this chapter, we review several decision tree induction methods. We focus on the most widely used methods and on methods we have developed. We rely on generalization methods and do not focus on methods that model subspaces in the decision space such as decision forests since in case of these methods the explanation capability is limited.

The preliminary concepts and the background are given in Section 2. This is followed by an overall description of a decision tree induction algorithm in Section 3. Different methods for the most important functions of a decision tree induction algorithm are described in Section 4 for the attribute selection, in Section 5 for attribute discretization, and in Section 6 for pruning.

The explanations given by the learnt decision tree must make sense to the domain expert since he often has already built up some partial knowledge. We describe in Section 7 what problems can arise and how the expert can be satisfied with the explanations about his domain. Decision tree induction is a supervised method and requires labeled data. The necessity to check the labels by an oracle-based classification approach is also explained in Section 7 as well as the feature subset-selection problem. It is explained in what way feature subselection can be used to improve the model besides the normal outcome of decision tree induction algorithm.

Section 8 deals with the question: How to interpret a learnt Decision Tree. Besides the well-known overall accuracy different more specific accuracy measures are given and their advantages are explained. We introduce a new quantitative measure that can describe changes in the structure of trees learnt from the same domain and help the user to interpret them. Finally we summarize our chapter in Section 9.

2 Preliminary Concepts and Background

The input to the decision tree induction algorithm is a data set which contains attributes in the column and data entries with its attributes values in each of the lines (see Fig. 1). From that the decision tree induction algorithm can automatically derive a set of rules that generalizes these data. The set of rules is represented as a tree. The decision tree recursively partition the solution space based on the attribute splits into subspaces until the final solutions is reached. The resulting hierarchical representation is very natural to the human problem-solving process. During the construction of the decision tree from

the whole set of attributes are selected only those attributes that are most relevant for the classification problem. Therefore a decision tree induction method can also be seen as a feature selection method.

Once the decision tree has been learnt and the developer is satisfied with the quality of the model the tree can be used in order to predict the outcome for new samples.

This learning method is also called supervised learning, since samples in the data collection have to be labeled by the class. Most decision tree induction algorithms allow using numerical attributes as well as categorical attributes. Therefore, the resulting classifier can make the decision based on both types of attributes.

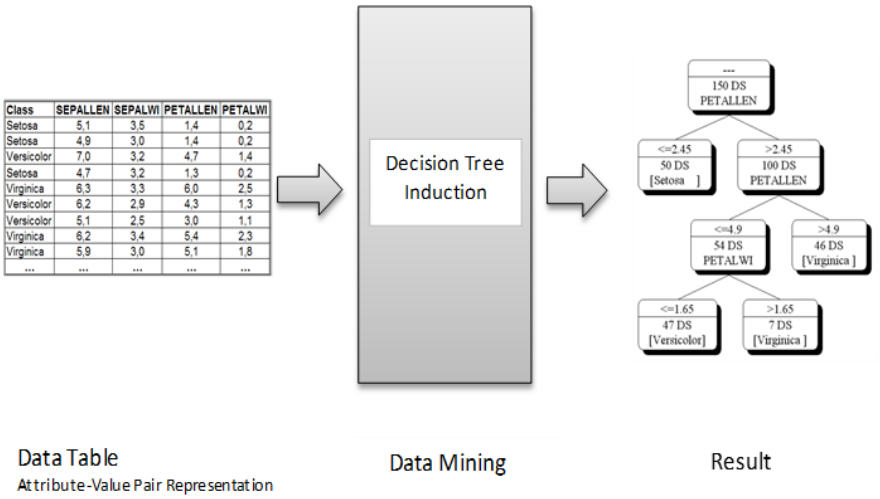


Fig. 1. Basic Principle of Decision Tree Induction

A decision tree is a directed acyclic graph consisting of edges and nodes (see Fig. 2).

The node with no edges entering is called the root node. The root node contains all class labels. Every node except the root node has exactly one entering edge. A node having no successor is called a leaf node or terminal node. All other nodes are called internal nodes.

The nodes of the tree contain the decision rules such as

$$IF\ attribute\ A\ \leq constant\ c\ THEN\ D.$$

The decision rule is a function f that maps the attribute A to D. The above described rule results into a binary tree. The sample set in each node is split into two subsets based on the constant c for the attribute A. This constant v is called cut-point.

In case of a binary tree, the decision is either true or false. In case of an *n-ary* tree, the decision is based on several constant c_i. Such a rule splits the data set into i subsets. Geometrically, the split describes a partition orthogonal to one of the coordinates of the decision space.

A terminal node should contain only samples of one class. If there are more than one class in the sample set we say there is class overlap. This class overlap in each terminal node is responsible for the error rate. An internal node contains always more than one class in the assigned sample set.

A path in the tree is a sequence of edges from (v_1,v_2), (v_2,v_3), ... , $(v_n\text{-}_1,v_n)$. We say the path is from v_1 to v_n and has a length of n. There is a unique path from the root to each node. The depth of a node v in a tree is the length of the path from the root to v. The height of node $\boldsymbol{v}$ in a tree is the length of the largest path from v to a leaf. The height of a tree is the height of its root. The level of a node $\boldsymbol{v}$ in a tree is the height of the tree minus the depth of v.

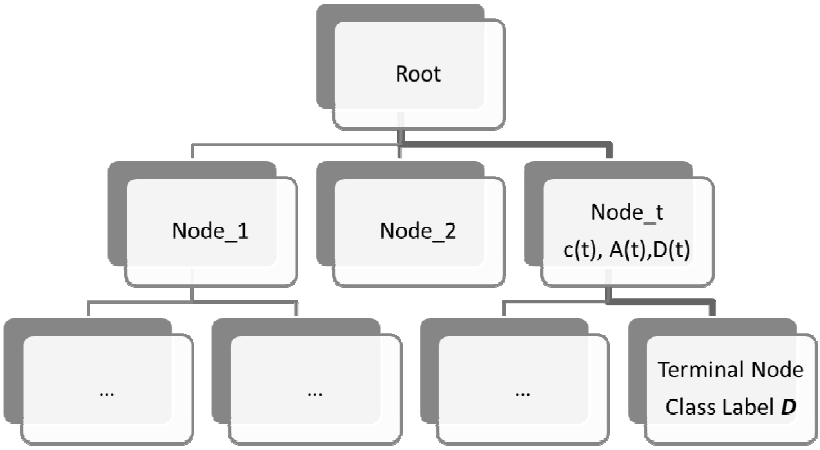


Fig. 2. Representation of a Decision Tree

A binary tree is an ordered tree such that each successor of a node is distinguished either as a left son or a right son. No node has more than one left son, nor has it more than one right son. Otherwise it is an *n-ary* tree.

Let us now consider, the decision tree learnt from Fisher´s Iris data set. This data set has three classes (*1-Setosa, 2-Vericolor, 3-Virginica*) with 50 observations for each class and four predictor variables (*petal length, petal width, sepal length, and sepal width*). The learnt tree is shown in Figure 3. It is a binary tree.

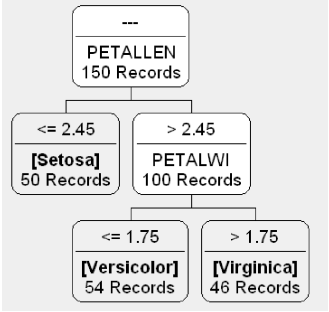


Fig. 3. Decision Tree learnt from Iris Data Set

The average depth of the tree is *(1+3+3+2)/4=9/4=2.25*. The root node contains the attribute *petal_length*. Along a path the rules are combined by the *AND* operator. Following the two paths from the root node we obtain, for example, two rules such as:

$$RULE\ 1:\ IF\ petal\ length\ \leq 2.45\ THEN\ Setosa$$

$$RULE\ 2: IF\ petal\ length < 2.45\ AND\ petal\ length < 4.9\ THEN\ Virginica.$$

In the latter rule we can see that the attribute *petal_length* will be used two times during the problem-solving process. Each time a different cut-point is used on this attribute.

3 Subtasks and Design Criteria for Decision Tree Induction

The overall procedure of the decision tree building process is summarized in Figure 4. Decision trees recursively split the decision space (see Fig. 5) into subspaces based on the decision rules in the nodes until the final stopping criterion is reached or the remaining sample set does not suggest further splitting. For this recursive splitting the tree building process must always pick among all attributes that attribute which shows the best result on the attribute selection criteria for the remaining sample set. Whereas for categorical attributes the partition of the attributes values is given a-priori, the partition of the attribute values for numerical attributes must be determined. This process is called attribute discretization process.

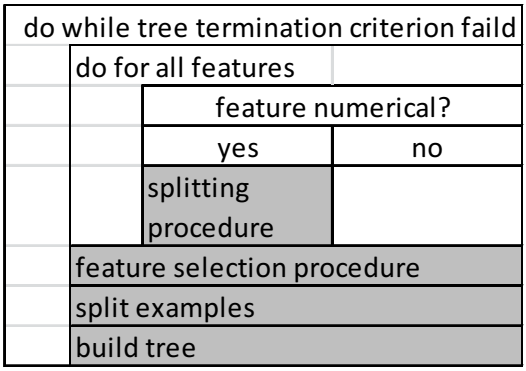


Fig. 4. Overall Tree Induction Procedure

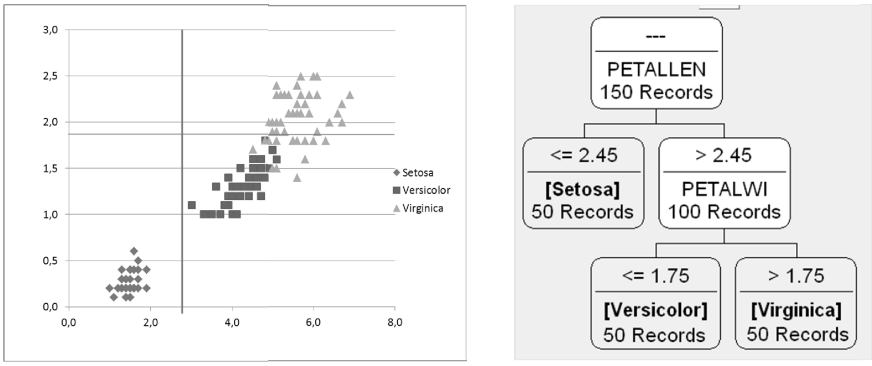


Fig. 5. Demonstration of Recursively Splitting of Decision Space based on two Attributes of the IRIS Data Set

The attribute discretization process can be done before or during the tree building process [1]. We will consider the case where the attribute discretization will be done during the tree building process. The discretization must be carried out before the attribute selection process, since the selected partition on the attribute values of a numerical attribute highly influences the prediction power of that attribute.

After the attribute selection criterion was calculated for all attributes based on the remaining sample set at the particular level of the tree, the resulting values are evaluated and the attribute with the best value for the attribute selection criterion is selected for further splitting of the sample set. Then, the tree is extended by two or more further nodes. To each node is assigned the subset created by splitting on the attribute values and the tree building process repeats.

Attribute splits can be done:

- univariate on numerically or ordinal ordered attributes A such as $A <= a$,
- multivariate on categorical or discretized numerical attributes such as $A \in |a|$, or
- as a linear combination split on numerically attributes $\sum_i a_i X_i \leq c$.

The influence of the kind of attribute splits on the resulting decision surface for two attributes is shown in Figure 6. The axis-parallel decision surface results in a rule such as

$$IF\ petal\ length\ \geq 4.9\ THEN\ Virginica$$

while the linear decision surface results in a rule such as

$$IF - 3.272 + 0.3254 * petal\ with + petal\ length\ \geq 0\ THEN\ Virginica.$$

The latter decision surface better discriminates between the two classes than the axis-parallel one, see Figure 6. However, by looking at the rules we can see that the explanation capability of the tree will decrease in the case of the linear decision surface.

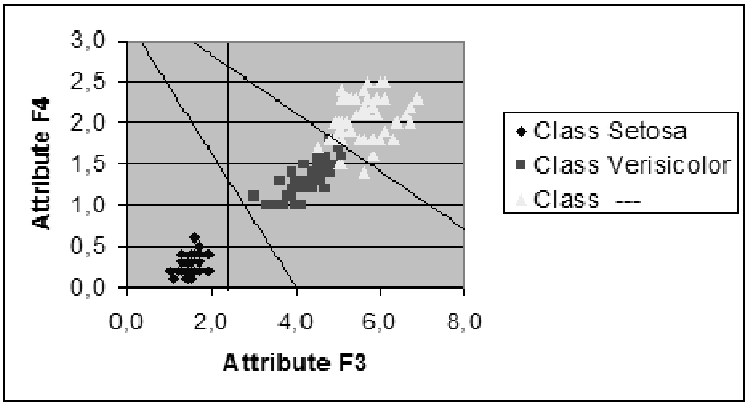


Fig. 6. Axis-Parallel and Linear Attribute Splits Graphically Viewed in the Decision Space

The induced decision tree tends to overfit to the data. This is typically caused due to noise in the attribute values and in the class information present in the training set. The tree building process will produce subtrees that fit to this noise. This causes an increased error rate when classifying unseen cases. Pruning the tree which means replacing subtrees with leaves can help to avoid this problem.

Now, we can summarize the main subtasks of decision tree induction as follows:

- attribute selection (Information Gain [2], X^2-Statistic [3], Gini-Index [4], Gain Ratio [5], Distance measure-based selection criteria [6],
- attribute discretization (Cut-Point [2], Chi-Merge [3], MDL-principle [7] , LVQ-based discretization, Histogram-based discretization, and Hybrid Methods [8],
- recursively splitting the data set, and
- pruning (Cost-Complexity [4], Reduced Error Reduction Pruning [2], Confidence Interval Method [9], Minimal Error Pruning [10]).

Beyond that decision tree induction algorithms can be distinguished in the way they access the data and in non-incremental and incremental algorithms.

Some algorithms access the whole data set in the main memory of the computer. This is insufficient when the data set is very large. Large data sets of millions of data do not fit into the main memory of the computer. They must be assessed from the disk or other storage device so that all these data can be mined. Accessing the data from external storage devices will cause long execution time. However, the user likes to get results fast and even for exploration purposes he likes to carry out quickly various experiments and compare them to each other. Therefore, special algorithms have been developed that can work efficiently although using external storage devices.

Incremental algorithms can update the tree according to the new data while non-incremental algorithms go through the whole tree building process again based on the combined old data set and the new data.

Some standard algorithms are: CART, ID3, C4.5, C5.0, Fuzzy C4.5, OC1, QUEST, CAL 5.

4 Attribute Selection Criteria

Formally, we can describe the attribute selection problem as follows: Let *Y* be the full set of attributes *A*, with cardinality *k,* and let n_i be the number of samples in the remaining sample set *i*. Let the feature selection criterion function for the attribute be represented by $S(A,n_i)$. Without any loss of generality, let us consider, a higher value of *S* to indicate a good attribute *A*. Formally, the problem of attribute selection is to find an attribute *A* based on our sample subset n_i that maximizes our criterion *S* so that

$$S(A,n_i) = \max_{ZCY,|Z|=1} S(Z,n_i) \tag{1}$$

Note that each attribute in the list of attributes (see Fig. 1) is tested against the chosen attribute selection criterion in the sequence it appears in the list of attributes. If two attributes have equal maximal value then the automatic rule picks the first appearing attribute.

Numerous attribute selection criteria are known. We will start with the most used criterion called information gain criterion.

4.1 Information Gain Criterion and Gain Ratio

Following the theory of the Shannon channel [11], we consider the data set as the source and measure the impurity of the received data when transmitted via the channel. The transmission over the channel results in the partition of the data set into subsets based on splits on the attribute values *J* of the attribute *A*. The aim should be to transmit the signal with the least loss on information. This can be described by the following criterion:

$$IF\ I(A) = I(C) - I(C/J) = Max!\ \ THEN\ \ \ Select\ \ Attribute\ A$$

where *I(A)* is the entropy of the source, *I(C)* is the entropy of the receiver or the expected entropy to generate the message $C_1, C_2, ..., C_m$, and *I(C/J)* is the losing entropy when branching on the attribute values *J* of attribute *A*.

For the calculation of this criterion, we consider first the contingency table in Table 4 with *m* the number of classes, *k* the number of attribute values *J*, *n* the number of examples, L_i number of examples with the attribute value J_i, R_j the number of examples belonging to class C_j, and x_{ij} the number of examples belonging to class C_j and having the attribute value J_i.

Now we can define the entropy over all class *C* by:

$$I(C) = -\sum_{j=1}^{m} \frac{R_j}{N} ld \frac{R_j}{N} \tag{2}$$

The entropy of the class given the feature-values, is:

$$I\left(\frac{C}{J}\right) = \sum_{i=1}^{n} \frac{L_i}{N} \sum_{j=1}^{m} -\frac{x_{ij}}{L_i} ld \frac{x_{ij}}{L_i} = \frac{1}{N}\left(\sum_{i=1}^{n} L_i\, ldL_i - \sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij} ldx_{ij}\right) \tag{3}$$

The best feature is the one that achieves the lowest value of (2) or, equivalently, the highest value of the "mutual information" *I(C) - I(C/J).* The main drawback of this measure is its sensitivity to the number of attribute values. In the extreme case a feature that takes N distinct values for the *N* examples achieves complete discrimination between different classes, giving *I(C/J)=0*, even though the features may consist of random noise and be useless for predicting the classes of future examples. Therefore, Quinlan [5] introduced a normalization by the entropy of the attribute itself:

$$G(A) = \frac{I(A)}{I(J)} \tag{4}$$

with

$$I(J) = -\sum_{i=1}^{n} \frac{L_i}{N} ld \frac{L_i}{N}.$$

Other normalizations have been proposed by Coppersmith et. al [13] and Lopez de Montaras [6]. Comparative studies have been done by White and Lui [14].

Table 1. Contingency Table for an Attribute

Class / Attribute values	C_1	C_2	..	C_j	..	C_m	SUM
J_1	x_{11}	x_{12}	..	x_{1j}	..	x_{1m}	L_1
J_2	x_{21}	x_{22}	..	x_{2j}	..	x_{2m}	L_2
⋮	⋮	⋮	⋱	⋮	⋱	⋮	⋮
J_i	x_{i1}	x_{i2}	..	x_{ij}	..	x_{im}	L_i
⋮	⋮	⋮	⋱	⋮	⋱	⋮	⋮
J_n	X_{k1}	X_{k2}	..	X_{kj}	..	X_{km}	L_n
SUM	R_1	R_2	..	R_j	..	R_m	N

4.2 Gini Function

This measure takes into account the impurity of the class distribution.

The selection criterion is defined as:

$$IF\ Gini(A) = G(C) - G(C/J) = Max!\ \ THEN\ \ Select\ \ Attribute\ A.$$

The Gini function for the class is:

$$G(C) = 1 - \sum_{j=1}^{m} \left(\frac{R_j}{N}\right)^2 \tag{5}$$

The Gini function of the class given the feature values is defined as:

$$G(C/J) = \sum_{i=1}^{n} \frac{L_i}{N} G(J_i) \tag{6}$$

with

$$G(J_i) = 1 - \sum_{j=1}^{m} \left(\frac{x_{ij}}{L_i}\right)^2$$

5 Discretization of Attribute Values

A numerical attribute may take any value on a continuous scale between its minimal value x_1 and its maximal value x_2. Branching on all these distinct attribute values does not lead to any generalization and would make the tree very sensitive to noise. Rather we should find meaningful partitions on the numerical values into intervals. The intervals should abstract the data in such a way that they cover the range of attribute values belonging to one class and that they separate them from those belonging to other classes. Then, we can treat the attribute as a discrete variable with *k+1* interval. This process is called discretization of attributes.

The points that split our attribute values into intervals are called cut-points. The cut-points *k* lies always on the border between the distributions of two classes.

Discretization can be done before the decision tree building process or during decision tree learning [1]. Here, we want to consider discretization during the tree building process. We call them dynamic and local discretization methods. They are dynamic since they work during the tree building process on the created subsample sets and they are local since they work on the recursively created subspaces. If we use the class label of each example we consider the method as supervised discretization methods. If we do not use the class label of the samples we call them unsupervised discretization methods. We can partition the attribute values into two *(k=1)* or more intervals *(k>1)*. Therefore we distinguish between binary and multi-interval discretization methods. The discretization process on numerical attribute values belongs to the attribute-value aggregation process, see Figure 7.

In Figure 8, we see the conditional histogram of the attribute values of the attribute *petal_length* of the IRIS data set. In the binary case *(k=1)* the attribute values would be split at the cut-point *2.35* into an interval from *0* to *2.35* and into a second interval from *2.36* to *7*. If we do multi-interval discretization, we will find another cut-point at *4.8*. That groups the values into three intervals *(k=2)*: intervall_1 from *0* to *2.35*, interval_2 from *2.36* to *4.8*, and interval_3 from *4.9* to *7*.

Attribute-value aggregation can also be mean full on categorical attributes. Many attribute values of a categorical attribute will lead to a partition of the sample set into many small subsample sets. This again will result in a quick stop of the tree building process. To avoid this problem, it might be wise to combine attribute values into a more abstract attribute value. We will call this process attribute aggregation. It is also possible to allow the user to combine attributes interactively during the tree building process. We call this process manual abstraction of attribute values, see Figure 7.

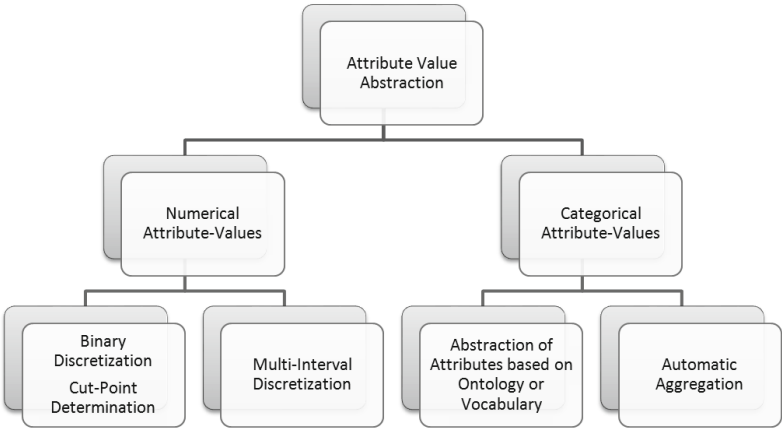


Fig. 7. Attribute Value Abstraction for numerical and categorical attributes

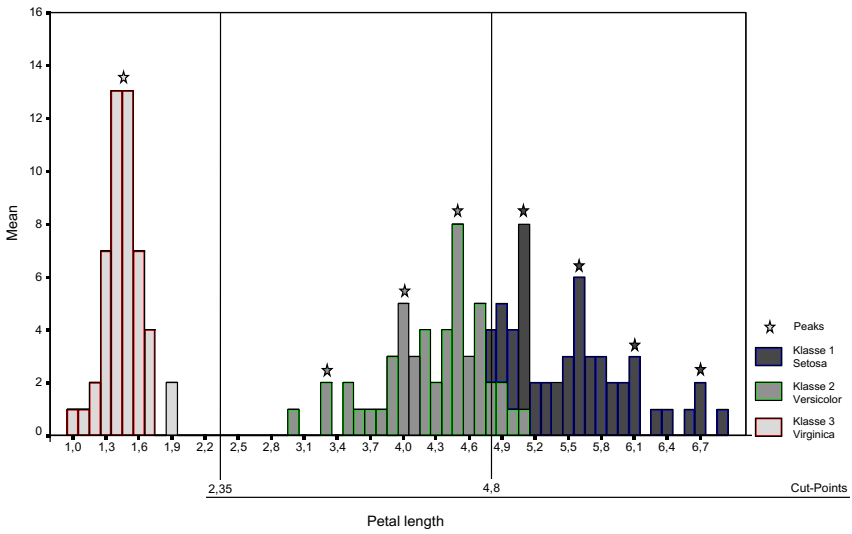


Fig. 8. Histogram of Attribute Petal Length with Binary and Multi-interval Cut-Points

5.1 Binary Discretization

Binary Discretization Based on Entropy

Decision tree induction algorithms like ID3 and C4.5 use an entropy criteria for the separation of attribute values into two intervals. On the attribute range between x_{min} and x_{max} each possible cut-point T is tested and the one that fulfills the following condition is chosen as cut-point T_A:

$$IF\ I(A, T_A; S) = Min!\ \ THEN\ \ Select\ T_A\ \ for\ T$$

with *S* the subsample set, *A* the attribute, and *T* the cut-point that separates the samples into subset S_1 and S_2.

I(A, T; S) is the entropy for the separation of the sample set into the subset S_1 and S_2:

$$I(A;T;S) = \frac{|S_1|}{|S|} I(S_1) + \frac{|S_2|}{|S|} I(S_2) \tag{8}$$

with

$$I(S_1) = -\sum_{j=1}^{m} p(C_j, S_1) ld\, p(C_j, S_1) \tag{9}$$

and $I(S_2)$ respectively.

The calculation of the cut-point is usually a time-consuming process since each possible cut-point is tested against the selection criteria. Therefore, algorithms have been proposed that speedup the calculation of the right cut-point [28].

Discretization Based on Inter- and Intra-Class Variance

To find the threshold we can also do unsupervised discretization. Therefore, we consider the problem as a clustering problem in a one-dimensional space. The ratio between the inter-class variance s_B^2 of the two subsets S_1 and S_2 and the intra-class variance s_w^2 in S_1 and S_2 is used as a criteria for finding the threshold:

$$s_B^2 = P_0(m_0 - m)^2 + P_1(m_1 - m)^2 \tag{10}$$

and

$$s_w^2 = P_0 s_0^2 + P_1 s_1^2 \tag{11}$$

The variances of the two groups are defined as:

$$s_0^2 = \sum_{i=x_1}^{T} (x_i - m_0)^2 \frac{h(x_i)}{N} \text{ and } s_1^2 = \sum_{i=T}^{x_2} (x_i - m)^2 \frac{h(x_i)}{N} \tag{12}$$

with *N* being the number of all samples and $h(x_i)$ the frequency of attribute value x_i. *T* is the threshold that will be tentatively moved over all attribute values. The values m_0 and m_1 are the mean values of the two groups that give us:

$$m = m_0 P_0 + m_1 P_1 \tag{13}$$

where P_0 and P_1 are the probability for the values of the subsets S_1 and S_2:

$$P_0 = \sum_{i=x_1}^{T} \frac{h(x_i)}{N} \text{ and } P_1 = \sum_{i=T+1}^{x_2} \frac{h(x_i)}{N} \tag{14}$$

The selection criterion is:

$$IF\ \frac{s_B^2}{s_w^2} = MAX!\ \ THEN\ Select\ T_A\ \ for\ \ T.$$

5.2 Multi-interval Discretization

Binary interval discretization will result in binary decision trees. This might not always be the best way to model the problem. The resulting decision tree can be very bushy and the explanation capability might not be good. The error rate might increase since the approximation of the decision space based on the binary decisions might not be advantageous and, therefore, leads to a higher approximation error. Depending on the data it might be better to create decision trees having more than two intervals for numerical attributes.

For multi-interval discretization we have to solve two problems:

1. Find multi-intervals and
2. Decide about the sufficient number of intervals.

The determination of the number of the intervals can be done static or dynamic. In the latter case the number of intervals will automatically be calculated during the learning process, whereas in the static case the number of intervals will be given a-priori by the user prior to the learning process. Then the discretization process will calculate as many intervals as it reaches the predefined number regardless of whether the class distribution in the intervals is sufficient or not. It results in trees having always the same number of attribute partitions in each node. All algorithms described above can be taken for this discretization process. The difference of binary interval discretization is that this process does not stop after the first cut-point has been determined, the process repeats until the given number of intervals is reached [8].

The sufficient number of intervals is automatically calculated during the dynamic discretization processes. The resulting decision tree will have different attribute partitions in each node depending on the class distribution of the attribute. For this process we need a criterion that allows us to determine the optimal number of intervals.

Basic (Search Strategies) Algorithm

Generally, we have to test any possible combinations of cut-points k in order to find the best cut-points. This would be computationally expensive. Since we assume that cut-points are always on the border of two distributions of x given class c, we have a heuristic for our search strategy.

Discretization can be done bottom-up or top-down. In the bottom-up case, we will start with a finite number of intervals. In the worst case, these intervals are equivalent to the original attribute values. They can also be selected by the user or estimated based on the maximum of the second-order probability distribution that will give us a hint of where the class borders are located. Starting from that the algorithm merges intervals that do meet the merging criterion until a stopping criterion is reached.

In the top-down case the algorithm first selects two intervals and recursively refines these intervals until the stopping criterion is reached.

Determination of the Number of Intervals

In the simplest case the user will specify how many intervals should be calculated for a numerical attribute. This procedure might become worse when there is no evidence

for the required number of intervals in the remaining data set. This will result in bushy decision trees or will stop the tree building process sooner as necessary. It would be better to calculate the number of intervals from the data.

Fayyad and Irani [7] developed a stopping criterion based on the minimum description length principle. Based on this criterion the number of intervals is calculated for the remaining data set during the decision tree induction. This discretization procedure is called MLD-based discretization.

Another criterion can use a cluster utility measure to determine the best suitable number of intervals.

Cluster Utility Criteria

Based on the inter-class variance and the intra-class variance we can create a cluster utility measure that allows us to determine the optimal number of intervals. We assume that inter-class variance and intra-class variance are the inter-interval variance and intra-interval variance.

Let s_w^2 be the intra-class variance and s_B^2 be the inter-class variance. Then we can define our utility criterion as follows:

$$U = \frac{\sum_{k=1}^{n} s_{Wk}^2 - s_{Bk}^2}{n} \tag{15}$$

The number of intervals n is chosen for minimal U.

MDL-Based Criteria

The MDL-based criterion was introduced by Fayyad and Irani [7]. Discretization is done based on the gain ratio. The gain ratio $I(A, T;S)$ is tested after each new interval against the MDL-criterion:

$$Gain(A,T;S) < \frac{ld\,(F-1)}{F} + \frac{\Delta(A,F;S)}{F} \tag{15}$$

where F is the number of instances in the set of values S,

$$Gain(A,T;S) = I(S) - I(A,T;S), \tag{16}$$

and

$$\Delta(A,T;S) = ld(3^k - 2) - [k * I(S) - k_1 * I(S_1) - k_2 * I(S_2)] \tag{17}$$

and k_i is the number of class labels represented in the set S_i.

One of the main problems with this discretization criterion is that it is relatively expensive. It must be evaluated F-1 times for each attribute. Typically, F is very large. Therefore, it would be good to have an algorithm which uses some assumption in order to reduce the computation time.

LVQ-Based Discretization

Vector quantization methods can also be used for the discretization of attribute values [8]. LVQ [15] is a supervised learning algorithm. This method attempts to define class

regions in the input-data space. At first, a number of codebook vectors W_i labeled by a class are placed into the input space. Usually, several codebook vectors are assigned to each class.

After an initialization of the LVQ, each learning sample is presented one or several times to the net. The input vector X will be compared to all codebook vectors W in order to find the closest codebook vector W_c. The learning algorithm will try to optimize the similarity between the codebook vectors and the learning samples by shifting the codebook vectors in the direction of the input vector if the sample represents the same class as the closest codebook vector. In case of the codebook vector and the input vector having different classes, the codebook vector gets shifted away from the input vector, so that the similarity between these two vectors decreases. All other code book vectors remain unchanged. The following equations represent this idea:

$$for\ equal\ classes: W_c(t+1) = W_c(t) + \alpha[X(t) - W_c(t)] \tag{18}$$

$$for\ different\ classes: W_c(t+1) = W_c(t) - \alpha[X(t) - W_c(t)] \tag{19}$$

$$for\ all\ others: W_j\,(t+1) = W_j(t) \tag{20}$$

This behavior of the algorithms can be employed for discretization. The algorithm tries to optimize the misclassification probability. A potential cut-point might be in the middle of the learned codebook vectors of two different classes. However, the proper initialization of the codebook vectors and the choice of the learning rate $\alpha(t)$ is a crucial problem.

Figure 9 shows this method based on the attribute *petal_length* of the IRIS domain.

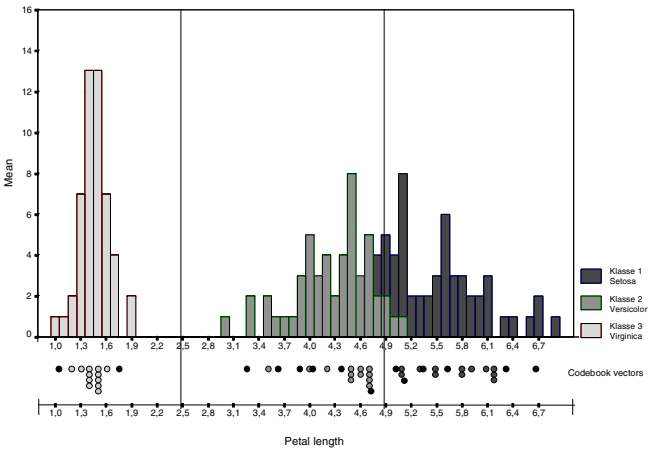


Fig. 9. Class Distribution of an Attribute, Codebook Vectors, and the learnt Cut-Points

Histogram-Based Discretization

A histogram-based method was first suggested by Wu et al. [16]. They used this method in an interactive way during top-down decision tree building. By observing

the histogram, the user selects the threshold which partitions the sample set in groups containing only samples of one class. In Perner et al. [8] an automatic histogram-based method for feature discretization is described.

The distribution $p(a \mid a \in C_k)P(C_k)$ of one attribute a according to classes C_k is calculated. The curve of the distribution is approximated by a first-order polynom with the coefficients a_0 and a_1 and the supporting places x_i that are the attribute values. The minimum square error method is used for approximating the real histogram curve by the first order polynom:

$$E = \sum_{i=1}^{n} (a_1 x_i + a_0 - y_i)^2 \tag{21}$$

The cut-points are selected by finding two maxima of different classes situated next to each other.

We used this method in two ways: First, we used the histogram-based discretization method as described before. Second, we used a combined discretization method based on the distribution $p(a \mid a \in S_k)P(S_k)$ and the entropy-based minimization criterion. We followed the corollary derived by Fayyad and Irani [7], which says that the entropy-based discretization criterion for finding a binary partition for a continuous attribute will always partition the data on a boundary point in the sequence of the examples ordered by the value of that attribute. A boundary point partitions the examples into two sets, having different classes. Taking into account this fact, we determine potential boundary points by finding the peaks of the distribution. If we found two peaks belonging to different classes, we used the entropy-based minimization criterion in order to find the exact cut-point between these two classes by evaluation of each boundary point K with $P_i \leq K \leq P_{i+1}$ between these two peaks. The resulting cut-points are shown in Figure 10.

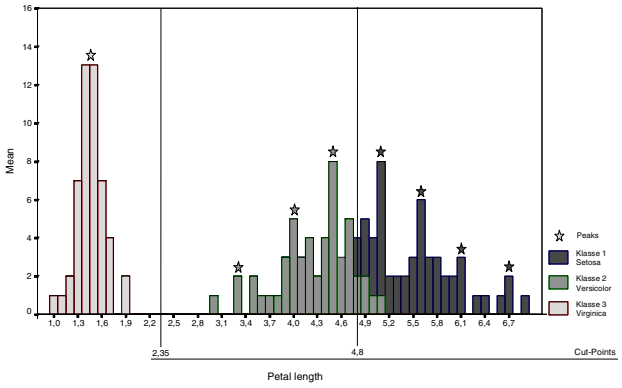


Fig. 10. Examples sorted by attribute values for attribute petal length, labeled peaks, and the selected cut-points

This method is not as time-consuming as the others. Compared to the other methods it gives us reasonably good results (see table5).

Chi-Merge Discretization

The ChiMerge algorithm introduced by Kerber [3] consists of an initialization step and a bottom-up merging process, where intervals are continuously merged until a termination condition is met. Kerber used the ChiMerge method static. In our study, we apply ChiMerge dynamically to discretization. The potential cut-points are investigated by testing two adjacent intervals by the χ^2 independence test. The statistical test is:

$$Chi^2 = \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{(A_{ij} - E_{ij})^2}{E_{ij}} \tag{22}$$

where m is equal two (the intervals being compared), k is the number of classes, A_{ij} is the number of examples in *ith* interval and *jth* class.

The expected frequency E_{ij} is calculated according to:

$$E_{ij} = \frac{R_i * C_j}{N} \tag{23}$$

where R_i is number of examples in *ith* interval; C_j is the number of examples in jth class; and N is the total number of examples.

First, all boundary points will be used for cut-points. In a second step for each pair of adjacent intervals one computes the χ^2-value. The two adjacent intervals with the lowest χ^2-value will be merged together. This step is repeated continuously until all χ^2-values exceed a given threshold. The value for the threshold is determined by selecting a desired significance level and then using a table or formula to obtain the χ^2-value.

The Influence of Discretization Methods on the Resulting Decision Tree

Figures 26-29 show learnt decision trees based on different discretization methods. It can be seen that the kind of discretization method influences the attribute selection. The attribute in the root node is the same for the decision tree based on Chi-Merge discretization (see Figure 26) and LVQ-based discretization (see Figure 28). The calculated intervals are roughly the same. Since the tree generation based on histogram discretization requires always two cut-points and since in the remaining sample set there is no evidence for two cut-points the learning process stops after the first level.

The two trees generated based on Chi-Merge discretization and on LVQ-based discretization have also the same attribute in the root. The intervals are slightly differently selected by the two methods. The tree in Figure 13 is the bushiest tree. However, the error rate (see table 2) of this tree calculated based on leave-one out is not better than the error rate of the tree shown in Figure 12. Since the decision is based on more attributes (see Figure 13) the experts might like this tree much more than the tree shown in Figure 14.

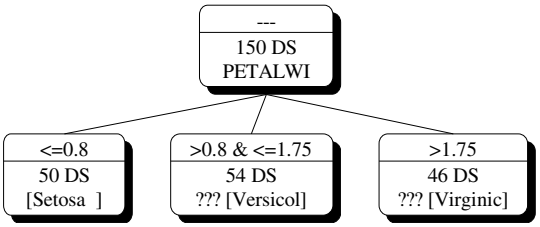


Fig. 11. Decision Tree based on Chi-Merge Discretization (k=3)

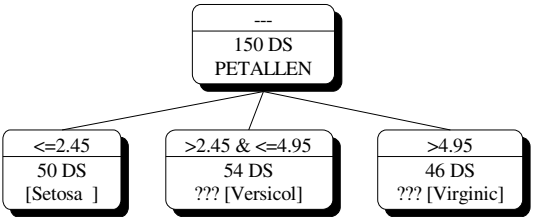


Fig. 12. Decision Tree based on Histogram-based Discretization (k=3)

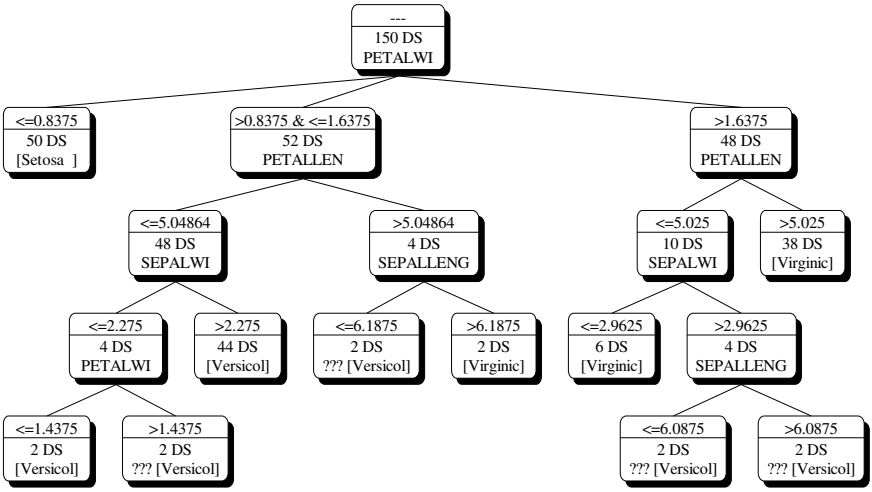


Fig. 13. Decision Tree based on LVQ-based Discretization

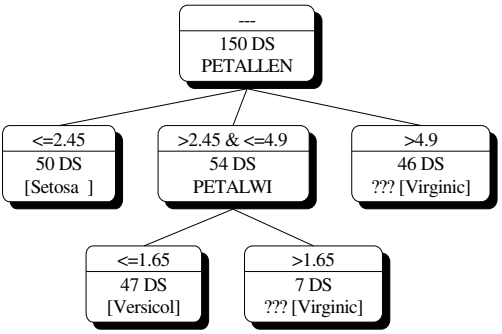


Fig. 14. Decision Tree based on MLD-Principle Discretization

Table 2. Error Rate for Decision Trees based on different Discretization Methods

Descritization Method	Error Rate	
	Unpruned Tree	Pruned Tree
Chi-Merge	7.33	7.33
Histogram-based Discr.	6	6
LVQ-based Discr.	4	5.33
MLD-based Discr.	4	4

5.3 Discretization of Categorical or Symbolical Attributes

Abstraction of Attribute Values Based on Ontology or Vocabulary
In opposition to numerical attributes, categorical, or symbolical attributes may have a large number of attribute values. Branching on such an attribute causes a partition into small subsample sets that will often lead to a quick stop of the tree-building process or even to trees with low explanation capabilities. One way to avoid this problem is the construction of meaningful abstractions on the attribute level at hand based on a careful analysis of the attribute list [17]. This has to be done in the preparation phase. The abstraction can only be done on the semantic level. Advantageous is that the resulting interval can be named with a symbol that a human can understand.

Automatic Abstraction
However, it is also possible to do automatically abstractions on symbolical attribute values during the tree-building process based on the class-attribute interdependence. Then, the discretization process is done bottom-up starting from the initial attribute intervals. The process stops when the criterion is reached.

6 Pruning

If the tree is allowed to grow to its maximum size, it is likely that it becomes overfitted to the training data. Noise in the attribute values and class information will amplify this problem. The tree-building process will produce subtrees that fit to noise. This unwarranted complexity causes an increased error rate when classifying unseen cases. This problem can be avoided by pruning the tree. Pruning means replacing subtrees by leaves based on some statistical criterion. This idea is illustrated in Figures 15 and 16 on the IRIS data set. The unpruned tree is a large and bushy tree with an estimated error rate of 6.67%. Subtrees get replaced by leaves up to the second level of the tree. The resulting pruned tree is smaller and the error rate becomes 4.67% calculated with cross-validation.

Pruning methods can be categorized either as pre- or post-pruning methods. In pre-pruning, the tree growing process is stopped according to a stopping criterion before the tree reaches its maximal size. In contrast, in post-pruning the tree is first developed to its maximum size and afterwards pruned back according to a pruning procedure.

However, pruning methods are always based on some assumptions. If this assumption of the pruning method is true for the particular data set can only be seen on the calculated error rate. There might be data sets where it is better to stay on the unpruned tree and there might be data sets where it is better to use the pruned tree.

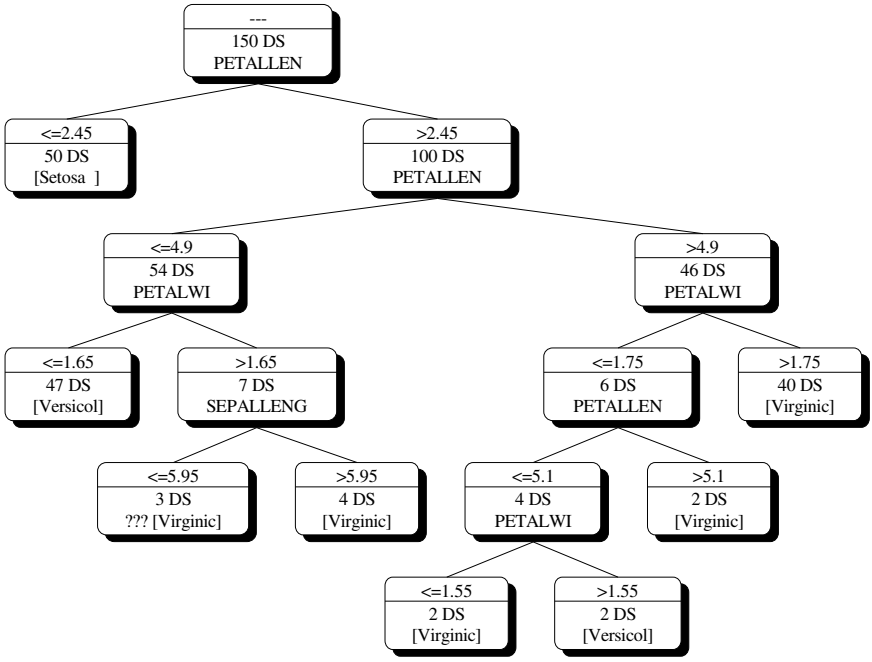


Fig. 15. Unpruned Decision Tree for the IRIS Data Set

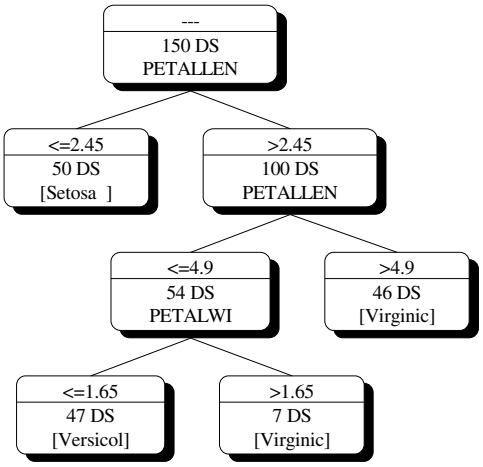


Fig. 16. Pruned Tree for the IRIS Data Set based on Minimal Error Pruning

6.1 Overview about Pruning Methods

Post-pruning methods can mainly be categorized into methods that use an independent pruning set and those that use no separate pruning set, see Figure 17. The latter can be further distinguished into methods that use traditional statistical measures, resampling methods like cross-validation and bootstrapping, and code-length motivated methods. Here, we only want to consider cost-complexity pruning and confidence-interval pruning that belongs to the methods with a separate pruning set. An overview of all methods can be found in Kuusisto [18].

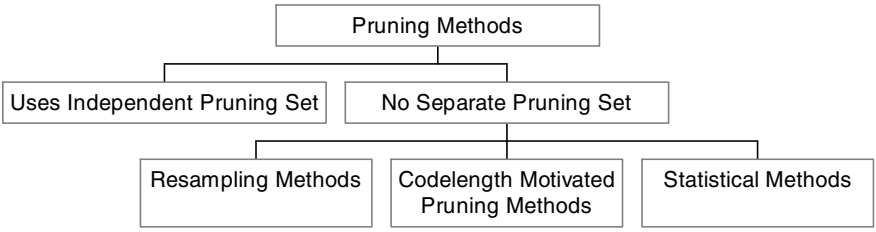


Fig. 17. General Overview of Pruning Methods

6.2 An Example of a Pruning Method - Cost-Complexity Pruning

The cost-complexity pruning method was introduced by Breiman et al. [4]. The main idea is to keep a balance between the misclassification costs and the complexity of the subtree (T) described by the number of leaves. Therefore, Breiman created a cost-complexity criteria as follows:

$$CP(T) = \frac{E(T)}{N(T)} + \alpha * Leaves(T) \tag{24}$$

with *E(T)* being the number of misclassified samples of the subtree *T, N(T)* is the number of samples belonging to the subtree *T, Leaves(T)* is the number of leaves of the subtree *T,* and α is a free defined parameter, which is often called complexity parameter. The subtree whose replacement causes minimal costs is replaced by a leaf:

$$IF\ \alpha = \frac{M}{N(T)*(Leaves(T)-1)} = Min!\ \ THEN\ \ Substitute\ Subtree.$$

The algorithm tentatively replaces all subtrees by leaves if the calculated value for α is minimal compared to the values α of the other replacements. This results in a sequence of trees $T_0 < T_2 <.. < T_i <.. < T_n$ where T_0 is the original tree and T_n is the root. The trees are evaluated on an independent data set. Among this set of tentative trees is selected the smallest tree as final tree that minimizes the misclassifications on the independent data set. This is called the *0-SE* selection method (0-standard error). Other approaches use a relaxed version, called *1-SE* method, in which the smallest tree does not exceed $E_{min}+SE(E_{min})$. E_{min} is the minimal number of errors that yields a decision tree T_i and $SE(E_{min})$ is the standard deviation of an empirical error estimated from the independent data set. $SE(E_{min})$ is calculated as follows:

$$SE(E_{min}) = \sqrt{\frac{E_{min}(N-E_{min})}{N}} \tag{25}$$

with *N* being the number of test samples.

7 Fitting Expert Knowledge into the Decision Tree Model, Improvement of Classification Performance, and Feature Subset Selection

There will be confusion when the domain expert has already built up some knowledge about his domain. Especially when the attribute in the root node is different from his evidence the surprise on the experts side is high. The root node decision should be the most confident decision. The reason for a change can be that there were two competing attributes having the same values for the attribute selection criterion. According to the automatic tree building process the attribute that appears first in the list of attributes is chosen for the desired node. When this happened for the first node of the tree, the whole structure of the tree will change since a different attribute in the first node will result in a different first split of the entire data set.

It is preferable for a decision tee induction tool that this situation is visually presented to the user so that he can judge what has happened. The tool might allow a user to interactively pick the attribute for the node in such a situation. That will not result in an automatic decision tree induction process and might not preferable for very large trees. Therefore, it should be only allowed until a predefined depth of a tree.

It might also be the situation that two significant attributes have slightly different values. The one the user prefers has a slightly lower value for the attribute selection criterion than the ones the decision tree induction algorithm automatically picks. This situation should be displayed to the user for his explanation.

Visualization techniques should show to user the location of the class-specific data distribution dependent on two attributes, as shown in Figure 11. This helps the user to understand what changed in the data. From a list of attributes the user can pick two attributes and the respective graph will be presented.

Decision tree induction is a supervised classification method. Each data entry in the data table used for the induction process needs a label. Noisy data might be caused by wrong labels applied by the expert or by other sources to the data. This might result in low classification accuracy.

The learnt classifier can be used in an oracle-based approach. Therefore, the data are classified by the learnt model. All data sets that are misclassified can be reviewed by the domain expert. If the expert is of the opinion that the data set needs another label, then the data set is relabeled. The tree is learnt again based on the newly labeled data set.

If the user has to label the data, then it might be apparent that the subjective decision about the class the data set belongs to might result in some noise. Depending on the form of the day of the expert or on his experience level, he will label the data properly or not as well as he should. Oracle-based classification methods [19][20] or similarity-based methods [21][22] might help the user to overcome such subjective factors.

The decision tree induction algorithm is also a feature selection algorithm. According to the criterion given in Section 5 the method selects from the original set of attributes Y with cardinality g the desired number of features o into the selected subset X while X *(Zeichen??)*Y.

There are two main methods for feature selection known: the filter approach (see Fig. 18) and the wrapper approach (see Fig. 19). While the filter approach attempts to assess the merits of features from the data alone, the wrapper approach attempts to the best feature subset for use with a particular classification algorithm.

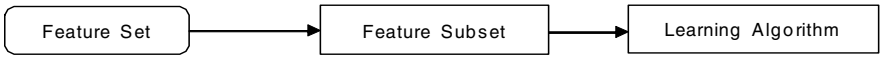


Fig. 18. Filter Approach for Feature Selection

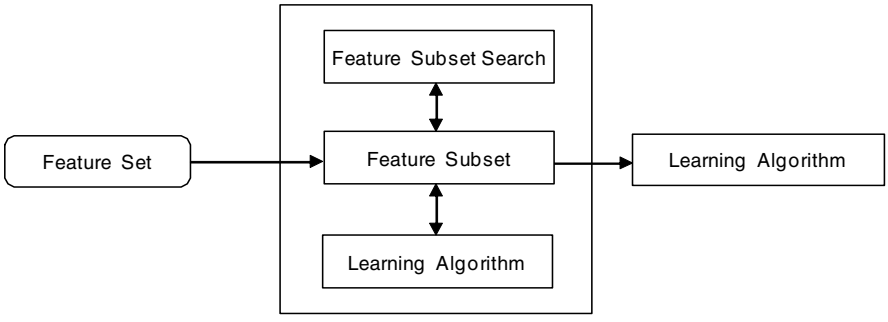


Fig. 19. Wrapper Approach for Feature Selection

In Perner [23], we have shown that using the filter approach before going into the decision tree induction algorithm will result in slightly better error rate.

Contrarily based on our experience, it is also possible to run the induction algorithm first and collect from the tree the chosen subset *X* of attributes. Based on that we can segment the database into a data set having only the subset *X* of attributes and run the decision tree induction algorithm again. The resulting classifier will often have better accuracy than the original one.

8 How to Interpret a Learnt Decision Tree?

For a black box model we only get some quantitative measures such as the error rate as quality criterion. These quantitative measures can also be calculated for decision trees. However, the structure of a tree and the rules contained in a tree are some other information a user can use to judge the quality of the tree. This will sometimes cause problems since the structure and the rules might change depending on the learning data set.

Often the user starts to learn a tree based on a data set DS_n and after some while he collects more data so that he gets a new data set DS_{n+1} (see Figure 20). If he combines data set DS_n and data set DS_{n+1} into a new data set DS' the resulting decision tree will change compared to the initial tree. Even when he learns the new tree only based on the data set DS_{n+1} he will get a different tree compared to the initial tree. Data sampling will not help him to come around this problem. On the contrary, this information can be used to understand what has been changed in the domain over time. For that he needs some knowledge about the tree building process and what the structure of a tree means in terms of generalization and rule syntax.

In this section, we will describe the quantitative measures for the quality of the decision tree and the measures for comparing two learnt trees.

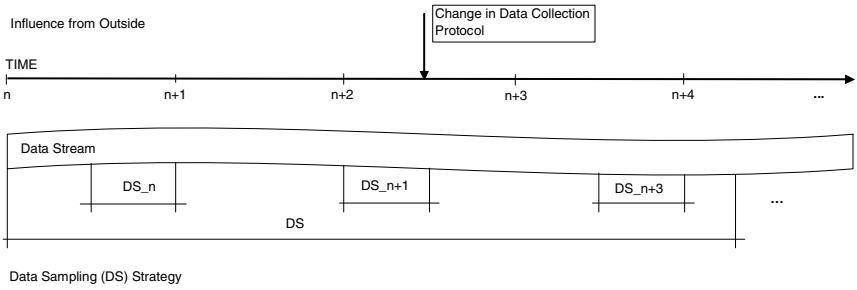


Fig. 20. The Data Collection Problem

8.1 Quantitative Measures for the Quality of the Decision Tree Model

One of the most important measures of the quality of a decision tree is accuracy, respectively, the error rate.

It is the number of false classified samples N_f divided by the whole number of samples N:

$$E = \frac{N_f}{N} \tag{26}$$

This measure is judged based on the available data set. Usually, cross-validation is used for evaluating the model since it is never clear if the available data set is a good representation of the entire domain. Compared to test-and-train, cross-validation can provide a measure statistically close to the true error rate. Especially if one has a small sample set, the prediction of the error rate based on cross-validation is a must. Although this is a well-known fact by now, there are still frequently results presented that are based on test-and-train and small sample sets. If a larger data set is available, cross-validation is also a better choice for the estimation of the error rate since one can never be sure if the data set covers the property of the whole domain. Faced with the problem of computational complexity, n-fold cross-validation is a good choice. It splits the whole data set subsequently into blocks of n and runs cross-validation based that.

The output of cross-validation is the mean accuracy. As you might know from statistics it is much better to predict a measure based on single measures obtained from a data set split into blocks of data and to average over the measure than predict the measure based on a single calculation on the whole data set. Moreover, the variance of the accuracy gives you another hint in regard to how good the measure is. If the variance is high, there is much noise in the data; if the variance is low, the result is much more stable.

The quality of a neural net is often not judged based on cross-validation. Cross-validation requires setting up a new model in each loop of the cycle. The mean accuracy over all values of the accuracy of the each single cycle is calculated as well as the standard deviation of accuracy. Neural nets are not automatically set up but decision trees are. A neural network needs a lot of training and people claim that such a neural net—once it is stable in its behavior is the gold standard. However, the accuracy is judged based on the test-and-train approach and it is not sure if it is the true accuracy.

Bootstrapping for the evaluation of accuracy is another choice but it is much more computationally expensive than cross-validation; therefore, many tools do not provide this procedure.

The mean accuracy and the standard deviation of the accuracy are overall measures, respectively. More detailed measures can be calculated that give a more detailed insight into the behavior of the model [24].

For that we use a contingency table in order to show the quality of a classifier, see Table 3. The table contains the assigned class distribution by the classifier and the real class distribution as well as the marginal distribution $\boldsymbol{c_{ij}}$. The main diagonal is the number of correct classified samples. The last row shows the number of samples assigned to the class assigned to this line and the last line shows the real class distribution in the data set. Based on this table, we can calculate parameters that assess the quality of the classifier in more detail.

Table 3. Contingency Table

		Real Class Index				
		1	**i**	**...**	**m**	**Sum**
Assigned Class Index	**1**	c_{11}	...	...	c_{1m}	
	...	...	...	...	...	
	i	c_{1i}	c_{ii}	...	c_{1m}	
	...	...	...	...	...	
	j	...	c_{ji}	...	...	
	...	...	...	...	...	
	m	c_{m1}	...	...	c_{mm}	
	Sum					

The ***correctness p*** is the number of correct classified samples over the number of samples:

$$p = \frac{\sum_{i=1}^{m} c_{ii}}{\sum_{i=1}^{m}\sum_{j=1}^{m} c_{ji}} \tag{27}$$

We can also measure the classification quality p_{ki} according to a particular class i and the number of correct classified samples p_{ti} for one class i:

$$p_{ki} = \frac{c_{ii}}{\sum_{j=1}^{m} c_{ji}} \text{ and } p_{ti} = \frac{c_{ii}}{\sum_{i=1}^{m} c_{ji}} \tag{28}$$

In the two class cases these measures are known as sensitivity and specifity. Based on the application domain is must be decided if for a class a high correctness is required or not.

Other criteria shown in Table 4 are also important when judging the quality of a model.

Table 4. Criteria for Comparison of Learned Classifiers

Generalization Capability of the Classifier	Error Rate based on the Test Data Set
Representation of the Classifier	Error Rate based on the Design Data Set
Classification Costs	• Number of Features used for Classification • Number of Nodes or Neurons
Explanation Capability	Can a human understand the decision
Learning Performance	Learning Time
	Sensitivity to Class Distribution in the Sample Set

If a classifier has a good representation capability which is judged based on the design data set, it might not mean that the classifier will have a good generalization capability which is judged on the test data set. It will not necessarily mean that the classifier will classify unseen samples with high accuracy.

Another criterion is the cost for classification expressed by the number of features and the number of decisions used during classification. The other criterion is the time needed for learning. We also consider the explanation capability of the classifier as another quality criterion and the learning performance. Decision trees can be fast constructed without heavy user interaction but they tend to be sensitive to the class distribution in the sample set.

8.2 Comparison of Two Decision Trees

Two data sets of the same domain that might be taken at different times might result in two different decision trees when separately used or if combined. Then the question arises: What has been changed in the data? What does it say about the domain? and How to interpret these changes in the structure of the two decision trees? If the models are not similar then something significant has changed in the data set and that also reflects that something is different in the domain. The problem can be seen as knowledge (domain) discovery problem.

We need to have a measure that allows us to judge the situation. Such a measure can be that the two directed graphs with its rules are compared to each other and a similarity measure is calculated.

The path from the top of a decision tree to the leaf is described by a rule like "*IF attribute A<= x and attribute B<=y and attribute C<=z and … THEN Class_1*". The transformation of a decision tree in a rule-like representation can be easily done. The location of an attribute is fixed by the structure of the decision tree. If an attribute appears at the first position in a rule and in another rule in a third or fourth position, then this is another meaning.

Comparison of rule sets is known from rule induction methods in different domains [25]. Here the induced rules are usually compared to the human-built rules [26][27] Often this is done manually and should give a measure about how good the constructed rule set is. We followed this idea and build a similarity measure for decision trees.

The kinds of rules transformed from a decision tree can be automatically compared by substructure mining.

The following questions can be asked:
a) How many rules are identical?
b) How many of them are identical compared to all rules?
c) What rules contain part structures of the decision tree?

We propose a first similarity measure for the differences of the two models as follows:

1. Transform two decision trees d_1 and d_2 into a rule set.
2. Order the rules of two decision tress according to the number n of attributes in a rule.

3. Then build substructures of all l rules by decomposing the rules into their Substructures.
4. Compare two rules i and j of two decision trees d_1 and d_2 for each of the n_j and n_i substructures with s attributes.
5. Build similarity measure SIM_{ij} according to formula 18-23.

The similarity measure is:

$$SIM_{ij} = \frac{1}{n}(Sim_1 + Sim_2 + \cdots + Sim_k + \cdots + Sim_n) \tag{29}$$

with

$$n = \{n_i, n_j\}$$

and

$$Sim_k = \begin{cases} 1 \; if \; substructure \; identity \\ 0 \; if \; otherwise \end{cases} \tag{30}$$

If the rule contains a numerical attribute $A<=k_1$ and $A´<=k_2=k_1+x$ then the similarity measure is

$$Sim_{num} = 1 - \frac{A-A´}{t} = 1 - \frac{k_1-k_1-|x|}{t} = 1 - \frac{|x|}{t} \;\; for \;\; x < t \tag{31}$$

and

$$Sim_k = 0 \;\; for \; x \;\; \geq t. \tag{32}$$

with t a user chosen value that allows x to be in a tolerance range of s % (e.g., 10%) of k_1. That means as long as the cut-point k_1 is within the tolerance range we consider the term as similar, outside the tolerance range it is dissimilar. Small changes around the first cut-point are allowed while a cut-point far from the first cut-point means that something seriously has happened with the data.

The similarity measure for the whole substructure is:

$$Sim_{\;k} = \frac{1}{S}\sum_{z=1}^{S}\{ \begin{matrix} Sim_{num} \\ 1 \;\; for \; A = A´ \\ 0 \; otherwise \end{matrix} \tag{33}$$

The overall similarity between two decision trees d_1 and d_2 is

$$Sim_{d_1,d_2} = \frac{1}{l}\sum_{i=1}^{l} \max_{\forall j} Sim_{ij} \tag{34}$$

for comparing the rules i of decision d_1 with rules j of decision d_2. Note that the similarity $Sim_{\;d2,d1}$ must not be the same.

The comparison of decision tree_1 in Figure 21 with decision tree_2 in Figure 22 gives a similarity value of *0.9166* based on the above described measure. The upper structure of decision tree_2 is similar to decision tree_1 but decision tree_2 has a few more lower leaves. The decision tree_3 in Figure 23 is similar to decision tree_1 by a similarity of 0.375. Decision tree_4 in Figure 24 has no similarity at all compared to all other trees. The similarity value is zero.

Such a similarity measure can help an expert to understand the developed model and also help to compare two models that have been built based on two data set, where one contains *N* examples and the other one contains *N+L* samples.

The similarity values runs between *0* and *1*while zero is dissimilar and one is identical. That gives us a semantic meaning of the similarity. A value between *1* and *0.51* means more or less similar while *0.5* is neutral. A value between *0.49* and *0* is more or less dissimilar and indicates that something important has been changed in the data.

Note only the rules have been changed. In combination with the error rate the meaning might be rules have been changed but error rate is better. That means the model is better and vice versa.

There are other options for constructing the similarity measure but this is left for our future work.

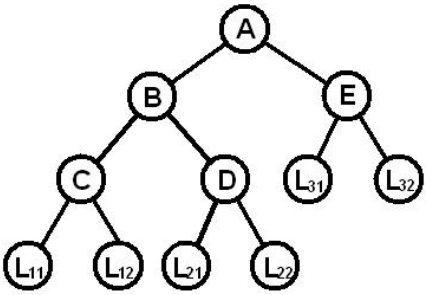


Fig. 21 Decision_Tree_1, $Sim_{d1,}d1=1$

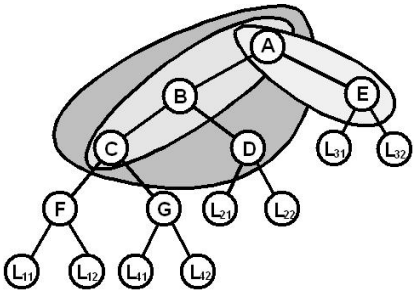


Fig. 22 Substructures of Decision Tree_1 to Decision Tree_2; $Sim_{d1,d2}=0.9166$

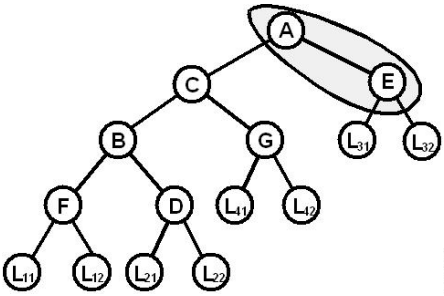


Fig. 23 Substructures of Decision Tree_1 to Decision Tree_3; $Sim_{d1,d3}=0.375$; $Sim_{d2,d3}=0.375$

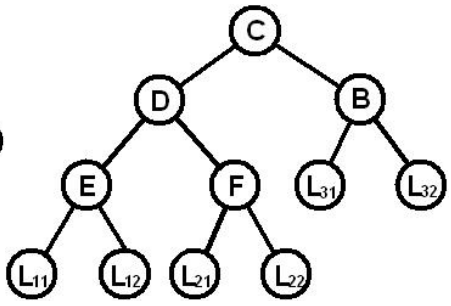


Fig. 24 Decision Tree_4 dissimilar to all other Decision Trees, $Sim_{d1,d4}=0$

9 Conclusions

In this chapter, we have described decision tree induction. We first explained the general methodology of decision tree induction and what are the advantages and disadvantages. Decision tree induction methods are easy to use. They only require a table of data for which the representation should be in an attribute value-based fashion. The tree induction process runs fully automatically. User interaction is not necessarily required. The fastness of the method allows quickly building a model that is preferable in many domains. Cross-validation is the method of choice to calculate the error rate based on the data set. The so calculated error rate comes close to the true error rate. Then we described the main structure of a decision tree and how an expert can read it in a common way. The overall algorithm of a decision tree induction method was given. For the main functions of the decision tree building process different methods have been explained for attribute selection, attribute discretization, and pruning. We described methods for attribute discretization that are standard methods and methods that we have developed.

Most of these methods described in this chapter are implemented in our Tool ***Decision Master***© (www.ibai-solutions.de). ***Decision Master*** is still one of the most flexible and reliable tools for decision tree induction and fits to the user needs. The examples given in this chapter have been calculated using the tool ***Decision Master***.

Many more decision tree induction algorithms have been developed over time. Most of them strongly depend on the underlying distribution in the data sample. The once that works on average good on all sample sets it up to our experience Quinlain´s *C4.5*. The other ones outperform *C4.5* on specific data sets but might give very worse results on other data sets.

It was explained how the explanations can be fit to the domain experts knowledge and what further can be done for feature selection in order to improve the quality of the model.

The quality of the model is assessed not only by the overall error rate rather than tis more specific error rates are necessary for a good evaluation of the model. We introduced a new quality criterion that can evaluate how much the structure of a tree differs from a former tree and how to interpret it.

Open problems in decision tree induction might be methods that can deal with imbalanced data sets, better visualization techniques of the properties in the data and of big decision tree models, strategies how to automatically deal with competing attributes, and more support on how to interpret decision trees.

References

1. Dougherty, J., Kohavi, R., Sahamin, M.: Supervised and Unsupervised Discretization of Continuous Features. In: 14th IJCAI Machine Learning, pp. 194–202 (1995)
2. Quinlan, J.R.: Induction of Decision Trees. Machine Learning 1, 81–106 (1998)

3. Kerber, R.: ChiMerge: Discretization of Numeric Attributes. In: AAAI 1992 Learning: Inductive, pp. 123–128 (1992)
4. Breiman, L., Friedman, J.H., Olshen, R.A.: Classification and Regression Trees. The Wadsworth Statistics/Probability Series, Belmont California (1984)
5. Quinlan, J.R.: Decision trees and multivalued attributes. In: Hayes, J.E., Michie, D., Richards, J. (eds.) Machine Intelligence 11. Oxford University Press (1988)
6. de Mantaras, R.L.: A distance-based attribute selection measure for decision tree induction. Machine Learning 6, 81–92 (1991)
7. Fayyad, U.M., Irani, K.B.: Multi-Interval Discretization of Continuous Valued Attributes for Classification Learning. In: 13th IJCAI Machine Learning, vol. 2, pp. 1022–1027. Morgan Kaufmann, Chambery (1993)
8. Perner, P., Trautzsch, S.: Multinterval Discretization for Decision Tree Learning. In: Amin, A., Pudil, P., Dori, D. (eds.) SPR 1998 and SSPR 1998. LNCS, vol. 1451, pp. 475–482. Springer, Heidelberg (1998)
9. Quinlan, J.R.: Simplifying decision trees. Machine Learning 27, 221–234 (1987)
10. Niblett, T., Bratko, I.: Construction decision trees in noisy domains. In: Bratko, I., Lavrac, N. (eds.) Progress in Machine Learning, pp. 67–78. Sigma Press, England (1987)
11. Philipow, E.: Handbuch der Elektrotechnik, Bd 2 Grundlagen der Informati-onstechnik, pp. 158–171. Technik Verlag, Berlin (1987)
12. Quinlan, J.R.: Decision trees and multivalued attributes. In: Hayes, J.E., Michie, D., Richards, J. (eds.) Machine Intelligence 11. Oxford University Press (1988)
13. Copersmith, D., Hong, S.J., Hosking, J.: Partitioning nominal attributes in decision trees. Journal of Data Mining and Knowledge Discovery 3(2), 100–200 (1999)
14. White, A.P., Lui, W.Z.: Bias in information-based measures in decision tree induction. Machine Learning 15, 321–329 (1994)
15. Kohonen, T.: Self-Organizing Maps. Springer (1995)
16. Wu, C., Landgrebe, D., Swain, P.: The decision tree approach to classification, School Elec. Eng., Purdue Univ., W. Lafayette, IN, Rep. RE-EE 75-17 (1975)
17. Perner, P., Belikova, T.B., Yashunskaya, N.I.: Knowledge Acquisition by Decision Tree Induction for Interpretation of Digital Images in Radiology. In: Perner, P., Rosenfeld, A., Wang, P. (eds.) SSPR 1996. LNCS, vol. 1121, pp. 208–219. Springer, Heidelberg (1996)
18. Kuusisto, S.: Application of the PMDL Principle to the Induction of Classification Trees. PhD-Thesis, Tampere Finland (1998)
19. Muggleton, S.: Duce - An Oracle-based Approach to Constructive Induction. In: Proceeding of the Tenth International Join Conference on Artificial Intelligence (IJCAI 1987), pp. 287–292 (1987)
20. Wu, B., Nevatia, R.: Improving Part based Object Detection by Unsupervised Online Boosting. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2007, pp. 1–8 (2007)
21. Whiteley, J.R., Davis, J.F.: A similarity-based approach to interpretation of sensor data using adaptive resonance theory. Computers & Chemical Engineering 18(7), 637–661 (1994)
22. Perner, P.: Prototype-Based Classification. Applied Intelligence 28(3), 238–246 (2008)
23. Perner, P.: Improving the Accuracy of Decision Tree Induction by Feature Pre-Selection. Applied Artificial Intelligence 15(8), 747–760
24. PernerZscherpelPerner, P., Zscherpel, U., Jacobsen, C.: A Comparision between Neural Networks and Decision Trees based on Data from Industrial Radiographic Testing. Pattern Recognition Letters 22, 47–54 (2001)

25. Georg, G., Séroussi, B., Bouaud, J.: Does GEM-Encoding Clinical Practice Guidelines Improve the Quality of Knowledge Bases? A Study with the Rule-Based Formalism. In: AMIA Annu. Symp. Proc. 2003, pp. 254–258 (2003)
26. Lee, S., Lee, S.H., Lee, K.C., Lee, M.H., Harashima, F.: Intelligent performance management of networks for advanced manufacturing systems. IEEE Transactions on Industrial Electronics 48(4), 731–741 (2001)
27. Bazijanec, B., Gausmann, O., Turowski, K.: Parsing Effort in a B2B Integration Scenario - An Industrial Case Study. In: Enterprise Interoperability II, Part IX, pp. 783–794. Springer (2007)
28. Seidelmann, G.: Using Heuristics to Speed Up Induction on Continuous-Valued Attributes. In: Brazdil, P.B. (ed.) ECML 1993. LNCS, vol. 667, pp. 390–395. Springer, Heidelberg (1993)

Sensory Data Gathering for Road Traffic Monitoring: Energy Efficiency, Reliability, and Fault Tolerance

Suchetana Chakraborty, Sandip Chakraborty, Sukumar Nandi, and Sushanta Karmakar

Department of Computer Science and Engineering,
Indian Institute of Technology, Guwahati, India 781039
{suchetana,c.sandip,sukumar,sushanatak}@iitg.ernet.in

Abstract. Vehicular traffic monitoring and control using through road sensor network is challenging due to a continuous data streaming over the resource constrained sensor devices. The delay sensitivity and reliability of the large volume of application data as well as the scarcity of sensor resources demand efficient designing of data collection protocol. In this Chapter, a novel tree-based data gathering scheme has been proposed, exploiting the strip-like structure of the road network. An efficient scheduling mechanism is implemented to assure both the coverage and the critical power savings of the sensor nodes. The network connectivity is guaranteed throughout by the proposed tree maintenance module that handles the dynamics of the network as a result of sensor node joining and leaving events. An application message controller has been designed that works cooperatively with the tree management module, and handles continuous streaming of the application data to ensure no loss or redundancy in data delivery. The performance of the proposed scheme is evaluated using the simulation results and compared with other approaches for large data collection in sensor network.

1 Introduction

Sensor networks have widespread applications in vehicular traffic monitoring along the highways to control the traffic signaling and vehicular traffic management [24]. In such networks, sensors are deployed along the road to sense the road condition or traffic load such that the data sensed from the network are accumulated at the road side gateways or sinks for monitoring and future analysis. This type of network applications require continuous data sensing and data streaming toward the gateways or sinks. A number of such sinks, connected to each other, are installed at road side with certain distance as given in Fig. 1. The square blocks denote the sinks and arrows represent the direction of data forwarding toward every sink. The area bounded by dotted rectangle is the area of interest for a particular sink and its corresponding dedicated sensors. Sensor nodes are power constrained devices limiting the performance, that makes the

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_4

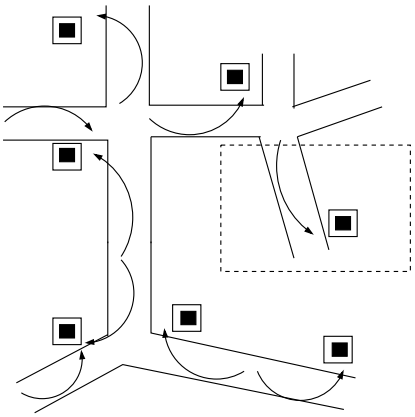


Fig. 1. Road sensor networks

continuous data streaming a challenging issue for design perspective. The traditional approaches for data collection in sensor network uses directional flooding, where the sensory data are streamed toward the sink. However, flooding-based approach is not scalable and reliable for a continuous data streaming over the low capacity devices. This is because the network gets overloaded with data packets, that increase the probability of packet loss, and consume an extra energy that makes the sensors more prone to sudden failure. Tree-based data gathering has many advantages over the data gathering through directional flooding. It offers an ordered delivery of application data with a minimum redundancy. Each node in a convergecast tree forwards the sensed data as well as the accumulated data from all its children to its parent node so that all the data are eventually delivered to the root or sink node. Thus, the collision-free limited communication saves the critical battery power at sensor node. Considering the strip-like physical distribution of the sensors for road traffic management, every set of sensors dedicated to a particular sink can form a Depth First Search (DFS) tree rooted at that sink. As internode communication consumes the maximum power [37], energy saving is crucial requirement for the resource-constrained sensors. Interference among neighbors, idle listening, and overhearing are other major sources of energy wastage. Past researches on sensor power management incorporated wakeup-based schedule for sensor nodes [18]. Based on slotted time interval, nodes switch between the sleep and wakeup state. To access sensory data at the sink or the gateway, it is required that the network remains connected at any point of time maintaining the sensing coverage [45].

There exists a number of chain-based, cluster-based, or tree-based solutions for data gathering in WSN [32]. As pointed out in [31], most of them do not consider coverage, connectivity, and fault tolerance during the data forwarding. Authors in [26] have proposed a virtual robust spanning tree-based protocol for data gathering maintaining the point coverage in WSN. Here, the sensing and relaying activities are distributed among nodes. Every node is assumed to know

the location of all the targets. The tree construction algorithm selects a tree edge based on the link weight and the hop-count distance. Authors in [25] have proposed an autonomous tree maintenance scheme for WSN with the assumption that all the nodes including the sink are homogeneous and can leave the network. The paper lacks in the theoretical complexity analysis and the coverage issues. Thus neighbor information is maintained at each node and the maintenance scheme is not local also. Another connected dominating set (CDS)-based topology control scheme has been proposed in [39] that focused mainly on the coverage issues. Here, nodes require the neighbor distance information. Additionally, no scheme for the topology maintenance and the application data handling has been provided. A comparative study between different graph-based topology control schemes and the CDS-based topology control technique has been reported in [22]. However, most of them require a 2-hop neighborhood information. For highly dense sensor network, a topology control scheme has been proposed by Iyengar *et al.* [12] assuring the connected-coverage and the low-coordination among sensors. Their wakeup-based scheme constructs the multiple node disjoint topologies to offer the fault tolerance with a minimum number of active nodes. The work assumed the position information of every node, and did not consider the changes in underlying topology. All the above-mentioned works on tree-based convergecast and topology control for general WSN are not applicable directly to the road sensor network environment where a continuous data streaming is essential. The limitations of the above works can be summarized as follows:

- Reliability should be ensured during the continuous data streaming over the resource constrained sensor devices. Duplicate data delivery and data flooding need to be avoided to assure an improved energy efficiency in the sensory devices. Further, the sensing coverage, the network connectivity and the fault-tolerance need to be assured simultaneously for an efficient sensing and the data collection. Most of the existing works do not ensure the reliability, coverage, connectivity and fault tolerance simultaneously [31], which are essential to be considered for the road sensor network. The existing works that ensure coverage during the data forwarding require either the position information or is centralized in nature.
- Most of the existing works as reported in [22] use the reactive path repairing technique, which is costly for delay sensitive road sensor networks.
- During the path establishment after a node fails, the reliability of application message delivery should be maintained. The existing data forwarding schemes do not consider the application message reliability during repairing time.

2 Literature Survey

WSN has emerged to become an integral part for the design and development of Intelligent Transport System (ITS) for road and traffic monitoring. Most of the works in the literature, based on road sensor networks have mainly focused on the

design aspects of low cost and effective sensors that can detect road conditions and moving vehicles. Different types of sensors have been designed for vehicular traffic monitoring, such as, optical remote sensing [27], air-bone sensors [28], laser scanner sensors [10], magnetic sensors [1], etc., out of which magnetic sensors have been proved to be the most reliable and cost-effective solution [6,43]. In [36], the authors have shown that magnetic impedance sensors provide better low-field sensitivity to detect the passing vehicles based on the Earth's magnetic flux. With the advances in the sensor hardware research for efficient road-surveillance, many algorithms have been designed for the development of ITS utilizing the advantages of magnetic sensors. The design objectives of these algorithms mainly include the detection of passing vehicles [19], the vehicle speed measurement [30], traffic information prediction [4,29], collision warning [33], and the traffic light control [35,3] etc.

Though the design and use of sensors for ITS have been widely studied in the literature, its networking aspects are still underexplored. As discussed earlier, sensor network for road-surveillance demands special design issues in terms of deployment parameters for the network architecture, topology control, sensor scheduling based on the connectivity and the coverage, and the design of data forwarding protocols based on road traffic characteristics. Two types of network architecture are explored in the literature - the vehicular sensor network [16,23,5,21,11,41], where the sensors are placed inside the vehicles, and the road sensor network [14,40,42,20,13], where the sensor nodes are placed along the roads or the pavements.

In a vehicular sensor network, sensors are placed inside the vehicles, and sensory data are forwarded either to the neighboring nodes or to the roadside sinks, for further processing and information extraction. The nodes in a vehicular sensor network are mobile in nature, that imposes several research challenges for the design of data collection protocols. In [16], the authors have proposed a coordination mechanism among the vehicular sensors to design a data collection protocol that aims in minimizing the network congestion. They have used a hybrid model where the mobile vehicular sensors forward data to the static roadside sensors. For congestion free data collection, their proposed scheme uses a collaboration mechanism among the static and the mobile sensors, based on the position information obtained from Geographical Positioning System (GPS). However, their scheme requires an absolute position information, and GPS is costly enough to use with every sensor node as it consumes a significant amount of power. Momen *et al.* [23] have proposed a random structure vehicular sensor system, where the network coverage is guaranteed based on the vehicular mobility model. In [21], the authors have developed a multi-hop data dissemination protocol for vehicular sensor network based on the absolute positioning information obtained from the GPS server. Though their protocol reduces the transmission delay, it suffers from extra power consumption required for the GPS functioning. Haddadou *et al.* [11] have proposed another data dissemination protocol for vehicular sensor network using a diffusion-based approach. Though the design of a vehicular sensor network have classified it as a cost-effective solution for vehicle monitoring,

it has some disadvantages for road surveillance. First, it is practically difficult to place the sensors on every vehicle, and the sensors may also be affected due to tampering. Next, in the context of network architecture, it is difficult to design a proactive mechanism for data collection from the mobile sensors. Further, mobile sensors may not guarantee the required connectivity and the sensing coverage always, and therefore, leaves the chances of false information prediction.

On the contrary, the road sensor network can provide a trustworthy architecture for road surveillance and vehicular traffic management. However, the design aspects of such a network are not being well-investigated in the existing literature. In [14], the authors have designed a passive localization algorithm for road sensor network. Xie *et al.* [40] have designed a topology management algorithm for road sensor network. The topology uses a linear placement strategy to reduce the load of the network traffic. However, their placement strategy does not guarantee the road coverage, and a vehicle may remain undetected. In [20], the authors have designed an optimization problem to reduce the communication cost in a road sensor network. Being a non-convex joint optimization problem, their solution is hard to implement in a distributed sensor network. Coleri *et al.* [7] have designed a traffic surveillance system based on road sensor network. They have used the *Power Efficient and Delay Aware Medium Access* (PEDAMACS) [8] protocol for the sensor scheduling to improve the network performance. In the PEDAMACS protocol, sensor nodes forward their one-hop neighbor information to the access points, and the access point determines the schedule based on the complete network information. In PEDAMACS, every node has three power levels $P_l > P_m > P_s$ corresponding to three transmission ranges $R_{cl} > R_{cm} > R_{cs}$. The largest transmission range R_{cl} is used by the base station to broadcast coordination packets to all sensor nodes. The smallest transmission range R_{cs} is used by sensor nodes to forward their data packets to the base station through multiple paths. Nodes transmit at medium transmission range to discover their latest topology. The PEDAMACS protocol works in four phases.

1. Topology learning phase: In this phase the topology learning coordination packets are forwarded from base station to sensors for synchronization. Thus, the construction of distributed tree rooted at the base station is performed by a flooding of tree construction packets in the network.
2. Topology collection phase: In this phase every node transmits 'local topology' packets, which include the list of node's parent, neighbors, and interferers, to its parent. This information is propagated to the base station.
3. Scheduling phase: In this phase the base station generates an interference free scheduling and the information is broadcast to all nodes in the network.
4. Adjustment phase: In this phase the updated topology information is propagated to the base station from all sensor nodes for rescheduling purpose.

This protocol incurs high control overhead in terms of control message transmission and delay involved in scheduling. There are few works in the literature that propose some design aspects of the road sensor network. Flathagen *et al.* have proposed an edge-betweenness community detection algorithm for a cluster

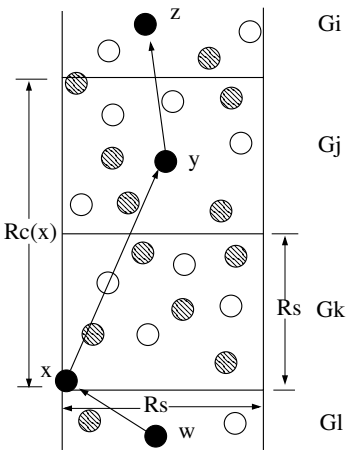


Fig. 2. Convergecast tree

determination to facilitate the in-network data aggregation in the road sensor network [9]. However, the required topology information exploiting the underlying routing protocol and centralized control increase the overhead. In [34], authors have investigated the problem of wakeup scheduling for the directional road sensor network, satisfying the connected-coverage criteria in the network. However, the problem of topology management as an effect of schedule maintenance or arbitrary node failure remained unaddressed.

This chapter presents a reliable and fault-tolerant continuous data streaming scheme maintaining the coverage throughout the road. To achieve the complete coverage even after node failure, the distribution of sensor nodes is virtually divided into the equal sized blocks, and the redundant nodes are placed in every block. It can be noted that the proposed scheme differs from the existing zone-based routing protocols used in mobile ad hoc networks, such as [17] and the references therein, in the sense that these protocols do not consider the coverage and connectivity issues in the sensor networks, and only concentrate on the data forwarding. The major contributions of this chapter are summarized as follows.

1. The proposed scheme aims to construct a DFS tree rooted at the sink node assuring both the coverage and connectivity into the network. The tree structure ensures a continuous data streaming over the sensor nodes with the minimum energy requirement. A tree management module is proposed to handle the dynamics of the network during the data streaming. The objective is to maintain the convergecast tree on the events of node leaving and joining in the tree either due to a state transition or due to an arbitrary node failure. The focus has been given to satisfy the connected-coverage criteria even after the repairing or recovery such that the minimum number of nodes remain in wakeup state at any point of time. The maintenance cost, both in terms of delay and communication, is aimed to be low.

2. The continuous streaming of application data are handled properly with the motivation of assuring no loss or redundancy in the data delivery. A data management module, that works cooperatively with the tree management module, has been designed for this purpose.
3. The effectiveness of the proposed scheme is evaluated using the simulation results, and compared with other naive approaches for data collection.

3 Convergecast Tree Management Scheme

The proposed tree management scheme is implemented by designing a sublayer within the network layer at every node. The sublayer consists of two modules, Tree Management Module (TMM) and Convergecast Controller (CC), that interact to each other through exchange of control signals. TMM is responsible for initial tree construction and maintenance of the tree where as CC takes care of the application messages. In distributed environment, the cooperative interaction among nodes, through control message exchange, enables the successful execution of the proposed scheme. System model and assumptions, description of TMM and CC along with the theoretical analysis have been provided in following subsections.

3.1 System Model and Assumptions

Magnetic sensors [1,15], that can detect the moving vehicles from disturbances in the Earth's magnetic field, are widely used for in-road deployment, due to high sensitivity, small size, and low-power consumption. Every sensor node is assumed to be static and equipotential in terms of battery power, memory limit, and processing capacity. Each sensor is assumed to have the data sensing range of radius R_s and the communication range of radius R_c such that $R_c = 2 \times R_s$ [38]. Nodes go to sleep periodically to save critical battery power. A node is called *active* when it is in wakeup state and actively participating in data sensing and forwarding. Again, a sensor node is called *inactive* when it is sleeping and saving the energy. In sleep mode, the transmitter does not send or receive any message. However a node can receive and respond to the triggered interrupt even it is sleeping. This chapter focuses on a part of the road sensor network consisting of a single sink and its set of dedicated sensors as shown by the bounded region in Fig. 1. It is assumed that the physical distribution of the sensors along the road is virtually divided into blocks of equal length R_s. The breadth of each block is typically $\leq R_s$. The set of nodes in every block forms a *group*. Nodes of a particular group are synchronized with the same schedule such that only one node stays awake at any time instance. However, the schedule of different groups are different. The assumption of keeping one node *active* at each block suffices to keep the road segment sensing-covered. The set of *active* and *inactive* nodes, for a particular group, together is called the set of *dedicated nodes* for that group. There exists a small overlapping region, called *timeout interval*, between two adjacent time slots of any schedule to accomplish the handover

activities for state transition. Every node is identified by a unique combination of nodeID and groupID. NodeIDs are computed randomly and assigned by the node itself temporarily based on consistent agreement in 1-hop neighborhood. Thus nodeIDs are unique only within 1-hop neighborhood. No position information or centralized control for node identification is required in this scheme. Abrupt power exhaustion, technical failure, or natural disaster may cause permanent failure of sensor nodes. This is why sensors are deployed in high redundancy to the environment of interest. In addition to the set of dedicated nodes, every block contains few redundant nodes for backup purpose. On failure of an *active* node, one redundant node from the corresponding block is awaken. The node is then added to the network through proper adjustment in the existing convergecast tree and schedule synchronization of the corresponding group.

3.2 Initialization

The sink node is assumed to be aware of the physical distribution of all the sensors. The sink node initiates the tree construction and assigns groupID to each node of the respective block during this phase. Once the groupID is set and the set of dedicated nodes are identified for a group, the schedule is computed locally for that particular group. The local schedule computation can be performed by running a distributed randomized algorithm at each dedicated node, similar to the one given as Algorithm 1 in [44]. This local computation of schedule also assures that global consistency and synchronization within a group. Once the *active* node for the next time slot is selected, remaining nodes of the dedicated set for that group go the sleep state. The sink node initiates the tree construction by 1-hop *Token* broadcast. All the *active* nodes from every block participate in tree construction by broadcasting the *Token* in 1-hop neighborhood in turn. As the *Token* is forwarded from the sink toward the nodes at the farthest distance, a DFS tree rooted at the sink, is constructed level by level maintaining the connectivity in the network. Every node in the tree sense data as well as forward it to the parent where all data are eventually forwarded to the root or sink node. A part of such convergecast tree has been shown in Fig. 2, where an arrow represents a tree edge and an dark shaded circle represents an *active* node. The light shaded circles represent the set of redundant nodes and empty circles denote the set of *inactive* nodes.

3.3 Tree Maintenance

Once the tree is constructed, the proposed TMM performs the maintenance activities on any changes in underlying topology. An active node u of group G_u might change the state from wakeup to sleep, where another node v of G_u would wake up to serve on behalf of the group. This events of leaving and joining of nodes in the tree due to state transition of the sensor nodes are called *graceful leaving* and *graceful joining*, respectively. Again, due to technical fault, disaster, or power exhaustion, a node may crash suddenly and thus leave the network. The parent and child of the leaving node as well as the newly added node must

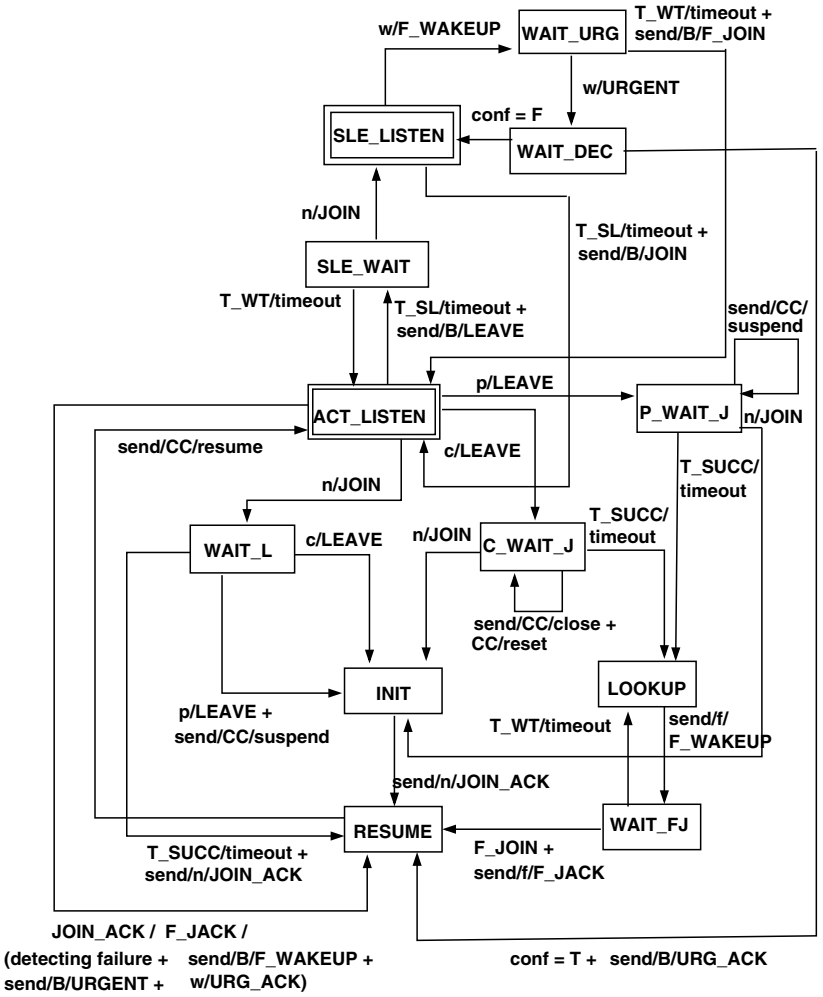


Fig. 3. State transition diagram for any node u

adapt to the changes in the tree such that no data is lost or delivered redundant. Based on these two cases of possible changes in underlying convergecast tree, the activities of each node have been modeled by state transition diagram (STD) as given in Fig.3. Initially, every active node remains in *ACT_LISTEN* state where as rest other nodes remain in *SLE_LISTEN* state. After the successful implementation of all necessary actions for tree maintenance, every node participating in maintenance eventually reaches the *RESUME* state. The tree maintenance activities have been described in view of the STD considering two different cases of node leaving. The control messages that trigger a transition between two states in STD, are assumed to be 1-hop.

Tree Maintenance (On Graceful Leaving and Joining of Nodes). Let *T_SL* denotes the timer that defines the sleep time for every node in a particular group. On timeout of *T_SL*, the active node $u \in G_k$ changes state from *ACT_LISTEN* to *SLE_WAIT* by broadcasting a *LEAVE* message. Similarly, another node $v \in G_k$ changes state from *SLE_LISTEN* to *ACT_LISTEN* by broadcasting a *JOIN* message on *T_SL* timeout. From *ACT_LISTEN* state, a node may switch to different states depending on the receipt of control messages to perform maintenance activities.

- ***ACT_LISTEN***
  - On receiving a *JOIN* message from the new node n, node u in *ACT_LISTEN* state goes to *WAIT_L* state and waits for a *LEAVE* message from either the parent or child node.
  - On receiving a *LEAVE* message at state *ACT_LISTEN*, node u goes to either state *C_WAIT_J* (if received from the child node c) or state *P_WAIT_J* (if received from the parent node p) and waits for a *JOIN* message.
  - After sleep timeout *T_SL*, a node goes to *SLE_WAIT* state and broadcast a *LEAVE* message.
- ***SLE_WAIT***
  - If a node receives a *JOIN* message within *T_WT* timeout, it goes to *SLE_LISTEN* state and changes its mode from active to sleep.
  - If a node does not receive a *JOIN* message within *T_WT* timeout, that indicates a possible node crash which will be discussed later.
- ***WAIT_L***
  - If the *LEAVE* message is received from the parent node p, node u goes to *INIT* state after sending a *suspend* signal to CC (to stop data forwarding from buffer).
  - If the *LEAVE* message is received from a child node c, node u goes to *INIT* state.
  - If no *LEAVE* message is received within *T_SUCC* timer timeout, it is assumed that the sending node has crashed. u safely broadcasts *JOIN_ACK* message and goes to *RESUME* state to set its parent and child accordingly.
- ***C_WAIT_J***
  - From state *C_WAIT_J*, node u sends a *close* signal to CC and waits for receiving *reset* signal from CC to remove the leaving node from child set.
  - On receiving *JOIN* message from the new node n, u goes to *INIT* state.
  - If no *JOIN* is received within *T_SUCC* timeout, node u goes to state *LOOKUP*, where it performs special maintenance activities.
- ***P_WAIT_J***
  - From state *P_WAIT_J*, node u sends a *suspend* signal to CC to stop data forwarding from buffer.
  - On receiving *JOIN* message from new node n, u reaches *INIT* state.
  - If no *JOIN* is received within *T_SUCC* timeout, node u goes to state *LOOKUP*, where it performs special maintenance activities.

– ***INIT***
  - As node u has received both the *LEAVE* and *JOIN* messages, it sends a *JOIN_ACK* message to the newly added node n and reaches to *RESUME* state to set the parent or child variable accordingly.

– **LOOKUP**
  - A node reaches to *LOOKUP* state if no *JOIN* message has been received within *T_SUCC* timeout after receiving a *LEAVE* message. This may happen if the intended recipient is outside the communication range of the new node, as shown in Fig. 4. In that case, to continue the uninterrupted data forwarding and maintain the tree connectivity, extra nodes from the set of *inactive* nodes are awaken forcefully by triggering interrupt. Thus for a particular group, under this kind of situation, more than one nodes may remain in *active* state. The challenge is to keep the count of *active* nodes for any group as low as possible. Node u at *LOOKUP* state, sends an interrupt *F_WAKEUP* to forcefully wakeup the next node to wakeup in G_u and goes to *WAIT_FJ* state.

– ***SLE_LISTEN***
  - A node remains in this state at sleep mode. There can be two actions on which a node goes from sleep mode to active mode, and changes its state.
  - After sleep timeout *T_SL*, a node goes to wake-up mode (normal scenario), broadcast *JOIN* message and goes to *ACT_LISTEN* state.
  - On receiving *F_WAKEUP* from a node w, the node goes to wakeup mode (forced wakeup scenario due to special maintenance activities), transfers to *WAIT_URG* state and waits *T_WT* time to make sure that it should perform special maintenance activities.

– ***WAIT_URG***
  - Node u at state *WAIT_URG* goes to *ACT_LISTEN* state after *T_WT* timeout by broadcasting a *F_JOIN* message.

– ***WAIT_FJ***
  - On receiving a *F_JOIN* message from node f, node at state *WAIT_FJ* sends a *F_JACK* message to node f and goes to *RESUME* state to set its parent or child accordingly.

– ***ACT_LISTEN*** (Extended)
  - A node that has changed the state from *WAIT_URG* to *ACT_LISTEN* by broadcasting a *F_JOIN* message, waits at *ACT_LISTEN* state for a *F_JACK* message. On receiving a *F_JACK* message, it goes to *RESUME* state and sets its parent or child accordingly.

– ***RESUME***
  - On reaching state *RESUME*, finally the node sends a *resume* signal to CC to start data forwarding from buffer. The node returns back to *ACT_LISTEN* state after successfully changing parent (or child) node and resuming back data forwarding activities.

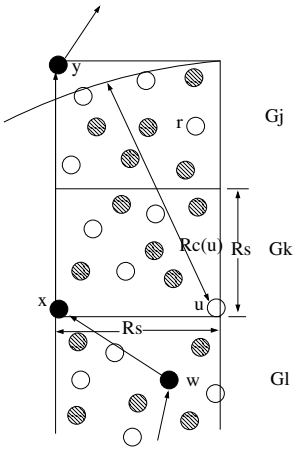


Fig. 4. Node y is outside of $R_c(u)$

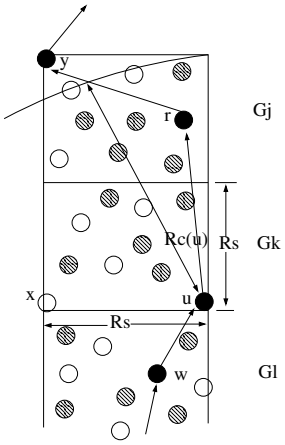


Fig. 5. The repaired tree

The maintenance scheme for handling the special case has been described pictorially in Fig. 4 and Fig. 5. Let node $x \in G_k$ remains *active* for the time slot t_m and goes to sleep during slot t_n as in Fig. 4. Another node, say $u \in G_k$, becomes *active* in slot t_n. τ_{mn} denotes the *timeout interval* between slot m and n. Let, $w \in G_l$ and $y \in G_j$ are *active* nodes for the corresponding blocks according to Fig. 4. If $y \in G_j$ is outside the range of $R_c(u)$, it does not receive *Join* message from u as in Fig. 4. On expiry of *T_SUCC*, y sends a *F_WAKEUP* signal as an interrupt to a node $r \in G_j$ where r is the node to become *active* for the time slot o (m, n, o are consecutive time slots for the schedule of G_j). On receiving the *F_WAKEUP* from y, node r becomes *active* and broadcasts a *F_JOIN* message. Node $u \in G_k$ on receiving *F_JOIN* from r, sets $parent(u) \leftarrow r$ and sends back *F_JACK* as an acknowledgment. On receiving *F_JACK*, node r sets $parent(r) \leftarrow y$ and $Child(r) \leftarrow u$ and thus, both the nodes y and r remain *active* for the time slot n. The application data is forwarded through the path $(\boldsymbol{w, u, r, y})$ according to Fig. 5. At the end of τ_{no}, node y goes to sleep and node r continues for the time slot o.

Tree Maintenance (On Sudden Leaving and Joining of Nodes). The activities of TMM for tree maintenance are different in case of node crash. Every active node is assumed to detect the node crash in neighborhood by lower layer fault detection technique. Let node $x \in G_k$ crashes suddenly at some time t in slot t_m. The maintenance scheme can be discussed in two cases based on the status of x at the time of failure. *Case 1 : The node fails in active mode*

From the STD given in Fig. 3, a node at *ACT_LISTEN* state, which detects the failure of a neighboring node, initiates the maintenance activities by broadcasting a *F_WAKEUP* interrupt signal followed by an *URGENT* message. On receiving *F_WAKEUP* interrupt signal at *SLE_LISTEN* state, a node in sleep

mode wakes up and goes to *WAIT_URG* state as discussed earlier. It can be noted that during tree maintenance for the special case (when two active nodes are not in the communication range of each other) the *F_WAKEUP* interrupt signal is used only to forcefully wakeup the next node in G_u according to the schedule (which is known to every other nodes in the group). However, for node crash, the *F_WAKEUP* interrupt signal is broadcast to wakeup all the nodes in the group to select the optimal node based on their *level of confidence* ($conf$). The *level of confidence* at each node is calculated based on the residual energy and the distance metric from both the parent and child of the crashed node (can be calculated from the signal strength and free space path loss model). The node with maximum residual energy and within the communication range of both the specified nodes is the best suitable for elected to be *active* node and thus, sets $conf$ to True. If multiple nodes satisfy the conditions, the one with the minimum nodeID is selected as the active one. The recovery actions for this case are as follows.

- ***WAIT_URG*** (Extended)
  - On receiving an *URGENT* message, every node u in G_k, goes to *WAIT_DEC* state to decide its *conf*.
- ***WAIT_DEC***
  - If $conf$ is set to True, the node broadcasts an *URG_ACK* message and reaches the *RESUME* state to set its parent and child accordingly.
  - If $conf$ is set to false, the node goes back to *SLE_LISTEN* state.
- ***ACT_LISTEN*** (Extended)
  - On receiving an *URGENT_ACK* message, a node that has sent an *URGENT* message, goes to *RESUME* state to set its parent and child accordingly.

Case 2 : The node fails in sleep mode

Let a node $x \in G_k$ should wake up in time slot t_n following normal schedule. It may happen that due to some nonrecoverable fault, x failed to wake up on time. The failure detection of node x is delayed until x is expected to participates in maintenance activities at time t. From the given STD in Fig. 3, a leaving node that has already sent a *LEAVE* message waits for a *JOIN* message from a new node at state *SLE_WAIT*. If u does not receive any *JOIN* message within *T_WT* timeout, it assumes the new node to be failed while sleeping and continues the active state by moving back into *ACT_LISTEN*.

Again, during the special maintenance activities, at *LOOKUP* state, a node sends a *F_WAKEUP* message to the node to be active in next slot and waits for an acknowledgment at state *WAIT_FJ*. If the next node to be active fails to acknowledge, it is assumed to be failed and the node u returns back to the *LOOKUP* state to reinitiate the process considering another sleeping node after *T_WT* timeout. Once a new node becomes *active* the tree is repaired locally following the similar steps as mentioned in the previous case. Following theorems show the correctness of the proposed scheme. Let $\mathbb{V}_T$ denotes the set of active nodes that are part of the convergecast tree at any instance of time. Also $d(x, y)$ be the eucleadian distance between any two nodes x and y.

Lemma 1. *Let a node $v \in G_j$ had broadcast a LEAVE message at time t_p. The message will eventually be received by both the nodes $u \in G_i$ by time t_q and $w \in G_k$ by time t_r, provided neither of the nodes u and w had failed between time t_p and t_r. G_i, G_j and G_k are three consecutive groups and u and w be the active nodes for the respective groups.*

Proof. At time t_p, node $v \in G_j$ broadcasts a *LEAVE* message. Therefore, as both the nodes $u \in G_i$ and $w \in G_k$ are active at time t_p, from the proposed scheme, $u, v, w \in \mathbb{V}_T$ at time t_p. Hence, at time t_p, $u = parent(v)$ and $w \in Child(v)$, which implies $d(u, v) < R_c$ and $d(v, w) < R_c$. If u is active at time $t_q > t_p$, it successfully receives the *LEAVE* message sent by node u by time t_q. Similarly, if w is active at time $t_r > t_p$, it successfully receives the *LEAVE* message sent by node u by time t_r. □

Lemma 2. *Let node $u \in G_i$ and $v \in G_j$ are two nodes such that G_i and G_j are consecutive to each other. The maximum distance between u and v is $\sqrt{5}R_s$.*

Proof. Each block is of dimension $R_s \times R_s$ and $R_c = 2R_s$, from assumption. Considering G_i and G_j are consecutive, two nodes u and v of the respective groups can be positioned at the end points of the diagonal, δ of the rectangle $2R_s \times R_s$, in the worst case. Hence, the maximum distance between u and v is the length of the diagonal δ and that is $\sqrt{5}R_s$. □

Lemma 3. *Let a node $u \in G_i$ had broadcast a JOIN message at time t_p. The message may not be received by another node $v \in G_j$, at any time greater than t_p where G_i and G_j are two consecutive groups and v be the active node for the corresponding group.*

Proof. From the assumption, for every group, at least one node is in active state and participates in convergecast tree. From Lemma 2, in the worst case, $d(u, v) = \sqrt{5}R_s > R_c$. Thus in the worst case, the *JOIN* message sent by node $u \in G_i$ will never be received by node $v \in G_j$, where G_i and G_j are consecutive. □

Lemma 4. *Let $v \in G_j$ and $w \in G_k$ be two active nodes such that the groups G_j and G_k are consecutive to each other. Now if $d(v, w) > R_c$ then there must exist at least one node u, in G_j (or in G_k), such that u becomes active to make the path $\{v, u, w\}$ a part of the tree.*

Proof. From Lemma 2, the maximum distance between two nodes $v \in G_j$ and $w \in G_k$ is $\sqrt{5}R_s$. As $\sqrt{5}R_s > R_c$, v and w will not be connected. Let v and w be positioned at the end point of the diagonal δ of the rectangle $\mathbb{R} = 2R_s \times R_s$, considering G_j and G_k consecutive and $\mathbb{R} = B_j \bigcup B_k$, where B_j and B_k are the blocks for the groups G_j and G_k, respectively. Let u be any node positioned at any point within the circular area of radius R_c and centered at w and u be within $\mathbb{R}$. Therefore, $d(u, w) < R_c$ and u and w are connected. Now, to maintain the connectivity such that $\{v, u, w\}$ be a path of the tree, $d(u, v) < R_c$ must be true. In the worst case, if u be positioned on the diagonal δ, to maintain

the path $\{v, u, w\}$, the minimum distance between u and w or u and v must be $(\sqrt{5}R_s - 2R_s)$. If $(\sqrt{5}R_s - 2R_s) \leq d(u, w) \leq R_c$, then $d(u, v) < R_c$ is true and vice- versa. Hence, there must exist at least one node u either in G_j or G_k to maintain the connectivity of the tree via path $\{v, u, w\}$. □

Theorem 1. *In the worst case, due to local maintenance minimum one and maximum two nodes in any group may remain in active state at a time.*

Proof. From assumption, initially one node from every group remains active to maintain both the connectivity and coverage. *Case 1 (Maintenance from graceful leaving) :* During the maintenance under normal state transition, from Lemma 3, the parent or child node w of a leaving node $u \in G_i$ may not receive the *JOIN* message from the new node $v \in G_i$. Thus, in the worst case, if $d(w, v) > R_c$, from Lemma 4, there exists at least one node z, either in G_i or in G_k such that $w \in G_k$, active to maintain the connectivity. If $z \in G_i$, then the group G_i has two active nodes, else the group G_k has two active nodes at that point of time. The objective is to keep minimum number of nodes active at any point of time and from Lemma 4, two nodes active in a group suffices to handle this situation.

Case 2 (Maintenance from node crash) : Again, during the maintenance from node crash, if more than one nodes are found to satisfy the conditions for being elected as active, the *conf* is set to True at one node only based on the value of nodeID. Thus, in this case, only one node remains active for the corresponding group. Hence proved. □

Theorem 2. *On leaving of node $v \in G_j$, either sudden or graceful, the parent and child of v is connected by either one or two nodes to maintain the constant connectivity such that at least one node remain active in G_j even after repairing.*

Proof. Let node $v \in G_j$ leaves by broadcasting a *LEAVE* message at time t_p. Let $u \in G_i$ be the parent of v and $w \in G_k$ be the child of node v such that G_i, G_j, G_k are consecutive. Now, let node $z \in G_j$ broadcasts a *JOIN* message such that both the nodes u and w receive it. Hence, u, z, w path is established where parent and child of the leaving node is connected by single node and one node remains active at every group. Now, let either of the nodes u and w does not receive the *JOIN* message according to Lemma 3. If u does not receive it, then another node $y \in G_i$ becomes active to establish the path $\{u, y, z, w\}$ according to Theorem 1. If w does not receive it, then another node $y \in G_k$ becomes active to establish the path $\{u, z, y, w\}$ according to Theorem 1. Hence, either u, y, z, w or u, z, y, w path is established where parent and child of the leaving node is connected by two nodes and at least one node remains active at every group. □

While tree maintenance is one issue, performed efficiently by TMM, the Quality of Service is another issue that is assured by the Convergecast Controller (CC), described in the following subsection.

3.4 Convergecast Controller

CC works cooperatively with the TMM and interacts with the TMM by exchanging few control signals, *start, close, suspend, resume*, and *reset*. The main

objective of the CC is to control the flow of convergecast data to and from the buffer at each node such that no data is lost or delivered redundant to the sink. Actions performed on receiving each such control signal are stated as follows.

- On receiving *start* from TMM, CC starts forwarding data from Buffer on wakeup of a node for the first time.
- On receiving *close* from TMM, CC forwards all the buffered data to the parent. On receiving last message from the child, CC sends a *reset* signal to TMM notifying the last message sent from the child has received.
- On receiving *reset* from CC, TMM resets its child set.
- On receiving *suspend* from TMM, CC stops data forwarding from the Buffer temporarily.
- On receiving *resume* from TMM, CC resumes data forwarding from the buffer.

Theorem 3. *Let G_i and G_j are two consecutive groups. Convergecast message upto sequence number S_m, sent from an active node $v \in G_j$ via its parent node $u \in G_i$, has been received by the sink node at time t_p. Then all the messages from sequence number S_n onwards, where $S_n = S_m + 1$, sent from node v via the new parent $w \in G_i$, will be received by the sink at time $t_q > t_p$, where node v does not crash.*

Proof. Let at time $t_m < t_p$, node $v \in G_j$ receives the *LEAVE* message from its parent node $u \in G_i$. According to the STD given in Fig. 3, at this state *P_WAIT_J*, node v sends a *suspend* signal to CC. CC at node v stops data forwarding from buffer. The sequence number of the last message sent from CC at node v to its parent node u be S_m. Now, node v changes state from *INIT* to *RESUME* after sending a *JOIN_ACK* message to the new parent node $z \in G_i$. As the path $\{v, z\}$ is established, node v sends a *resume* signal to CC to start data fowarding and goes to *ACT_LISTEN* state. Thus, all data stored in buffer at node v starting from sequence number $S_n = S_m + 1$ are forwarded to the new parent node z. No data is lost at node v.

Again, the leaving node u forwards all the received data from its child node v upto sequence number S_m to its parent node w before going to sleep. Node v does not send any message with sequence number greater than S_m to node u. Hence no message is lost at node u.

Let at time $t_m < t_n < t_p$, node w, receives the *LEAVE* message from its child node $u \in G_i$. From STD given in Fig. 3, st this state *C_WAIT_J*, node v sends a *close* signal to its CC. On receiving the last message with sequence number S_m from the child node u, CC sends back a *reset* signal to TMM of node w such that node w can safely remove node u from its child set. At this point, node w has forwarded the last message, with sequence number S_m, received from node u toward the sink. Thus no message is lost at node w. Let the message with sequence number S_m be received at the sink at time t_p. Let the path $\{v, z, w\}$ establishment time be Δ. As node, v has forwarded the messages starting from sequence number S_n to node z and z will eventually forward this to its parent,

node w. The messages starting from sequence number S_n will be received by the sink node via node w at t_q, where $t_q \geq t_p + \Delta$. Hence proved. □

The proposed scheme assures the correct delivery of all application data with no loss or redundancy, even during the local maintenance under normal state transition. However, if a node crashes before forwarding all its buffered data to its parent, the loss of those buffered data at the crashed node can not be avoided.

4 Simulation Result

The proposed scheme has been simulated with the message-passing interface of NS2 [2]. The simulation scenario has been designed by considering a strip-like topology with 25 nodes including the sink. For every sensor, R_c has been considered to be 250 unit and R_s be half of that. There exists five blocks (corresponding groups G_1 to G_5), each with the dimension of $R_s \times R_s$. The sensors are distributed according to the proposed scheme maintaining the redundancy. The MAC layer protocol has been considered to be SMAC with 1 Mbps data rate. Transport layer protocol is UDP and Application data traffic is assumed to be CBR with data generation rate of 12 Kbps. The first state transition for G_1 occurs at 1^{st} sec from the simulation starts. The schedule interval between two consecutive blocks is 10 ms where as the slot time is 30 ms for every schedule in every block. Each sensor is assumed to have 5 Joule of energy in store initially. Considering micaZ sensor mote, the per node energy dissipation is assumed to be $0.053W$ for receiving, $0.045W$ for transmitting and $0.0027W$ for sleeping. The proposed scheme has been compared with the diffusion-based convergecast and the chain-based convergecast under same topology. For diffusion-based convergecast, all the 25 nodes are active all the time. For chain-based convergecast five nodes, each from different block, that participate in chain construction are active all the time. For the proposed scheme, only the set of designated nodes participate in data forwarding periodically. The result presented in Fig. 6 shows that the proposed scheme outperforms the other two schemes. The slope for the plotted line denoting the average residual energy for the proposed scheme is almost the half of the diffusion- and chain-based solution. This implies that the proposed scheme would help in increasing the network lifetime. Fig. 7 shows average energy dissipation with respect to number of groups. The contention among nodes increases as the number of groups increases. As a result, the average energy dissipation increases exponentially for diffusion and linearly for chain-based forwarding. However, for the proposed scheme, average energy dissipation is almost constant. The periodic sleep-wakeup schedule and redundancy-based failure handling reduces the contention among neighboring nodes.

The proposed scheme has also been simulated to observe the effect of tree management delay on the performance of convergecast and the amount of packets received at the sink. Average delay for an application message sent from a node to be received by the sink node is called average end-to-end delay Δ, and is plotted with respect to every group. Δ is maximum because of collision if the diffusion-based convergecast is used. On the other hand, the chain-based convergecast

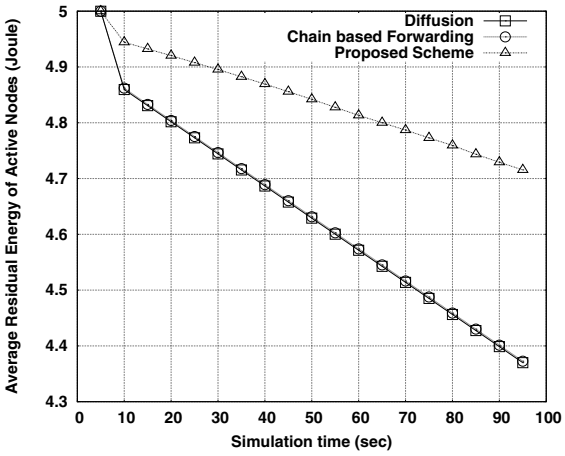


Fig. 6. Average residual energy of the network

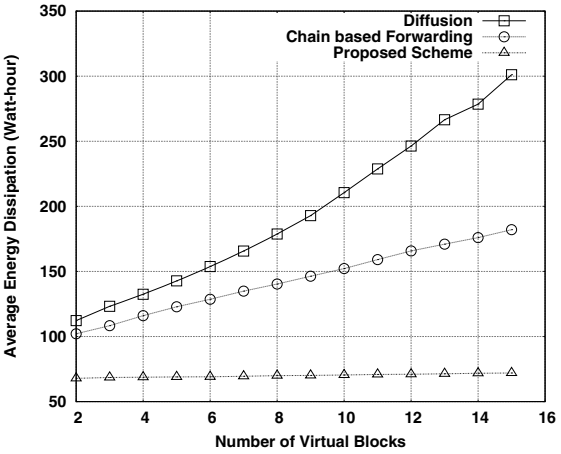


Fig. 7. Average energy dissipation

offers minimum Δ. Fig. 8 shows that offered Δ for the proposed scheme is almost similar to that of the chain-based solution even in presence of tree maintenance responsibility. Thus, the proposed scheme would serve good for delay sensitive road sensor network. It has also been noted that the % of loss and redundancy for the application messages received at the sink is 0. Due to local repairing and recovery technique the proposed scheme is scalable too. Fig. 9 shows the amount of packets received from individual groups. It can be observed from the figures that the proposed scheme outperforms diffusion- and chain-based forwarding.

Cumulative control overhead at time t_n can be defined as the fraction of total number of control messages sent to the total number of messages sent in

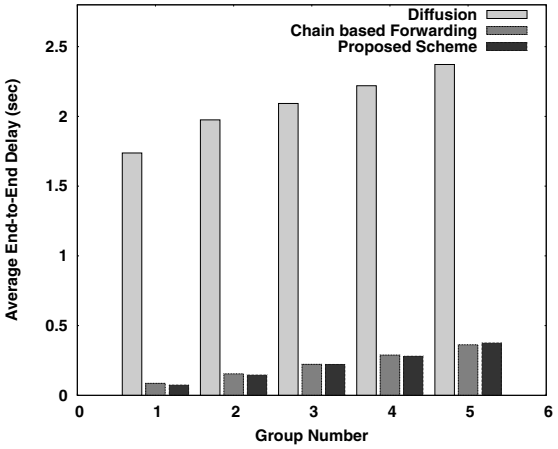


Fig. 8. Average end-to-end delay

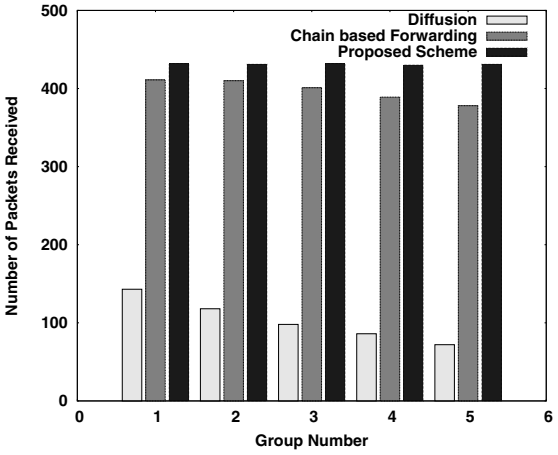


Fig. 9. Number of packets received at sink

the network during the time interval $[t_0, t_n]$, where t_0 denotes the time nodes started communication. From Fig. 10, it can be observed that the cumulative control overhead for the proposed scheme is only the 2% and that also remains stable with time. This proves that the maintenance overhead in terms of control message communication is nominal and constant for the proposed scheme. Fig. 11 shows the probability of simultaneous activation of two nodes in a single block. It can be seen from the figure that the probability is very less. For 16 numbers of consecutive blocks, on average only 2.4% of blocks will have two active nodes simultaneously. It can be noted that the above two metrics are not compared with diffusion- and chain-based forwarding, as they does not ensure coverage and connectivity during node failure.

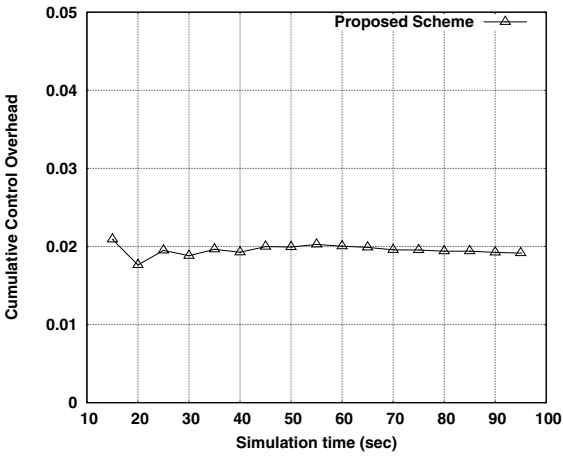


Fig. 10. Cumulative control message overhead

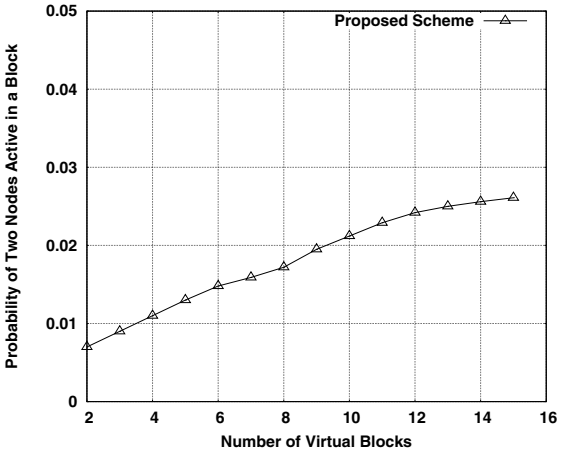


Fig. 11. Probability of two active nodes in a block

5 Conclusion and Future Directions of Research

Wireless sensor network is an effective technology for providing various intelligent services, like automated traffic monitoring and road signaling control, for intelligent transport systems. One of the major problems associated with this network is the continuous data streaming from the roadside sensors, which need to be delivered to the base station in an effective and timely manner. This chapter gives an overview of the works proposed in the literature for designing effective protocols for data gathering in a roadside sensor network. The discussion over the state-of-the-art works reveals that the existing works do not consider the effect of sensor scheduling as well as the arbitrary sensor failures during the data

gathering procedure. In this chapter a novel tree-based data gathering scheme has been proposed for road side sensor networks. The network coverage has been assured by distributing the sensor nodes in virtual blocks and selecting one active node from each block periodically to construct the data gathering tree. As the nodes join and leave the network, the connectivity is maintained throughout by designing an efficient tree maintenance module that assures correct delivery of data. The effectiveness of the proposed scheme has been justified using simulation results. The data gathering scheme proposed in this chapter assumes a simple sleep-wakeup scheduling of the sensors. As the sensor scheduling has significant impact over the data streaming, and the performance of the data gathering scheme depends on the scheduling mechanism, the proposed scheme in this chapter can be extended to design an integrated scheduling and data forwarding scheme for road-side sensor networks. The future research in this direction can explore several other issues, like assuring the sensing and communication coverage during sensor scheduling for seamless data sensing, streaming, forwarding, and collection, as an objective to design a complete road traffic monitoring system.

Acknowledgment. The authors would like to thank TATA Consultancy Services (TCS), India for supporting this work through TCS Research Fellowship to the first two authors.

References

1. Magnetic sensors, honeywell, `http://www.magneticsensors.com/vehicle-detection-solutions.php`
2. Ns-2 network simulator, version 2.34, `http://www.isi.edu/nsnam/ns/`
3. Ceriotti, M., Corra, M., D'Orazio, L., Doriguzzi, R., Facchin, D., Guna, S., Jesi, G., Lo Cigno, R., Mottola, L., Murphy, A., Pescalli, M., Picco, G., Pregnolato, D., Torghele, C.: Is there light at the ends of the tunnel? wireless sensor networks for adaptive lighting in road tunnels. In: Proceedings of the 10th International Conference on Information Processing in Sensor Networks, pp. 187–198 (2011)
4. Chan, K.Y., Dillon, T.: On-road sensor configuration design for traffic flow prediction using fuzzy neural networks and taguchi method. IEEE Transactions on Instrumentation and Measurement 62(1), 50–59 (2013)
5. Chen, L.W., Peng, Y.H., Tseng, Y.C.: An infrastructure-less framework for preventing rear-end collisions by vehicular sensor networks. IEEE Communications Letters 15(3), 358–360 (2011)
6. Cheung, S.Y., Varaiya, P.: Traffic surveillance by wireless sensor networks: Final report. Tech. Rep. UCB-ITS-PRR-2007-4, University of California, Berkeley (2007)
7. Coleri, S., Cheung, S.Y., Varaiya, P.: Sensor networks for monitoring traffic. In: Proceedings of the Allerton Conference on Communication, Control and Computing (2004)

8. Ergen, S.C., Varaiya, P.: Pedamacs: Power efficient and delay aware medium access protocol for sensor networks. IEEE Transactions on Mobile Computing 5, 920–930 (2006)
9. Flathagen, J., Drugan, O., Engelstad, P., Kure, O.: Increasing the lifetime of roadside sensor networks using edge-betweenness clustering. In: Proc. of IEEE ICC, pp. 1–6 (2011)
10. Gallego, N., Mocholi, A., Menendez, M., Barrales, R.: Traffic monitoring: Improving road safety using a laser scanner sensor. In: Proceedings of the Electronics, Robotics and Automotive Mechanics Conference, pp. 281–286 (2009)
11. Haddadou, N., Rachedi, A., Ghamri-Doudane, Y.: Advanced diffusion of classified data in vehicular sensor networks. In: Proceedings of the 7th International Wireless Communications and Mobile Computing Conference, pp. 777–782 (2011)
12. Iyengar, R., Kar, K., Banerjee, S.: Low-coordination topologies for redundancy in sensor networks. In: Proc. of the 6th ACM MobiHoc, pp. 332–342 (2005)
13. Jeong, J., Guo, S., He, T., Du, D.: APL: Autonomous passive localization for wireless sensors deployed in road networks. In: Proceedings of the 27th IEEE Conference on Computer Communications, pp. 583–591 (2008)
14. Jeong, J., Guo, S., He, T., Du, D.: Autonomous passive localization algorithm for road sensor networks. IEEE Transactions on Computers 60(11), 1622–1637 (2011)
15. Knaian, A.N.: A wireless sensor network for smart roadbeds and intelligent transportation systems. Tech. rep., Massachusetts Institute of Technology, USA (2000)
16. Kong, F., Tan, J.: A collaboration-based hybrid vehicular sensor network architecture. In: Proceedings of the International Conference on Information and Automation, pp. 584–589 (2008)
17. Koyama, A., Honma, Y., Arai, J., Barolli, L.: An enhanced zone-based routing protocol for mobile ad-hoc networks based on route reliability. In: Proceedings of the 20th IEEE AINA, pp. 61–68 (2006)
18. Kumar, S., Chauhan, S.: A survey on scheduling algorithms for wireless sensor networks. International Journal of Computer Applications 20(5), 7–13 (2011)
19. Gu Lee, B., Han Kim, J.: Algorithm for finding the moving direction of a vehicle using magnetic sensor. In: Proceedings of the IEEE Symposium on Computational Intelligence in Control and Automation, pp. 74–79 (2011)
20. Li, W., Chan, E., Hamdi, M., Lu, S., Chen, D.: Communication cost minimization in wireless sensor and actor networks for road surveillance. IEEE Transactions on Vehicular Technology 60(2), 618–631 (2011)
21. Lim, K.W., Jung, W.S., Ko, Y.B.: Multi-hop data dissemination with replicas in vehicular sensor networks. In: proceedings of the IEEE Vehicular Technology Conference, pp. 3062–3066 (2008)
22. Manolopoulos, Y., Katsaros, D., Papadimitriou, A.: Topology control algorithms for wireless sensor networks: a critical survey. In: Proc. of the 11th CompSysTech, pp. 1–10 (2010)
23. Momen, A., Azmi, P., Bazazan, F., Hassani, A.: Optimised random structure vehicular sensor network. IET Intelligent Transport Systems 5(1), 90–99 (2011)
24. Ng, E.H., Tan, S.L., Guzman, J.: Road traffic monitoring using a wireless vehicle sensor network. In: Proceedings on IEEE International Symposium on Intelligent Signal Processing and Communications Systems, pp. 1–4 (2009)

25. Onodera, K., Miyazaki, T.: An autonomous algorithm for construction of energy-conscious communication tree in wireless sensor networks. In: Proc. of the 22nd AINAW, pp. 898–903 (2008)
26. Ostovari, P., Dehghan, M., Wu, J.: Connected point coverage in wireless sensor networks using robust spanning trees. In: Proc. of the 31st ICDCSW, pp. 287–293 (2011)
27. Palubinskas, G., Kurz, F., Reinartz, P.: Detection of traffic congestion in optical remote sensing imagery. In: Proceedings of the IEEE International Geoscience and Remote Sensing Symposium, vol. 2, pp. 426–429 (2008)
28. Pantavungkour, S., Shibasaki, R.: Three line scanner, modern airborne sensor and algorithm of vehicle detection along mega-city street. In: Proceedings of the 2nd GRSS/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas, pp. 263–267 (2003)
29. Pascale, A., Nicoli, M., Deflorio, F., Dalla Chiara, B., Spagnolini, U.: Wireless sensor networks for traffic management and road safety. IET Intelligent Transport Systems 6(1), 67–77 (2012)
30. Pelczar, C., Sung, K., Kim, J., Jang, B.: Vehicle speed measurement using wireless sensor nodes. In: Proceedings of the IEEE International Conference on Vehicular Electronics and Safety, pp. 195–198 (2008)
31. Rothery, S., Hu, W., Corke, P.: An empirical study of data collection protocols for wireless sensor networks. In: Proceedings of the ACM REALWSN, pp. 16–20 (2008)
32. Santi, P.: Topology control in wireless ad hoc and sensor networks. ACM Comput. Surv. 37(2), 164–194 (2005)
33. Sung, K., Yoo, J.J., Kim, D.: Collision warning system on a curved road using wireless sensor networks. In: Proceedings of the IEEE 66th Vehicular Technology Conference, pp. 1942–1946 (2007)
34. Tang, J., Zhu, B., Zhang, L., Hincapie, R.: Wakeup scheduling in roadside directional sensor networks. In: Proc. of IEEE GLOBECOM, pp. 1–6 (2011)
35. Tubaishat, M., Qi, Q., Shang, Y., Shi, H.: Wireless sensor-based traffic light control. In: Proceedings of the 5th IEEE Consumer Communications and Networking Conference, pp. 702–706 (2008)
36. Uchiyama, T., Mohri, K., Itho, H., Nakashima, K., Ohuchi, J., Sudo, Y.: Car traffic monitoring system using MI sensor built-in disk set on the road. IEEE Transactions on Magnetics 36(5), 3670–3672 (2000)
37. Wang, Q., Hempstead, M., Yang, W.: A realistic power consumption model for wireless sensor network devices. In: Proc. of 3rd Annual IEEE SECON, vol. 1, pp. 286–295 (2006)
38. Wang, X., Xing, G., Zhang, Y., Lu, C., Pless, R., Gill, C.: Integrated coverage and connectivity configuration in wireless sensor networks. In: Proc. of the 1st SenSys, pp. 28–39 (2003)
39. Wightman, P., Labrador, M.: A3Cov: A new topology construction protocol for connected area coverage in WSN. In: Proc. of IEEE WCNC, pp. 522–527 (2011)
40. Xie, W., Zhang, X., Chen, H.: Wireless sensor network topology used for road traffic. In: Proceedings of the IET Conference on Wireless, Mobile and Sensor Networks, pp. 285–288 (2007)
41. Yu, X., Liu, Y., Zhu, Y., Feng, W., Zhang, L., Rashvand, H., Li, V.O.K.: Efficient sampling and compressive sensing for urban monitoring vehicular sensor networks. IET Wireless Sensor Systems 2(3), 214–221 (2012)

42. Zeng, Y., Xiang, K., Li, D.: Applying behavior recognition in road detection using vehicle sensor networks. In: Proceedings of the International Conference on Computing, Networking and Communications, pp. 751–755 (2012)
43. Zhang, L., Wang, R., Cui, L.: Real-time traffic monitoring with magnetic sensor networks. Journal of Information Science and Engineering 27, 1473–1486 (2011)
44. Zhou, G., Huang, C., Yan, T., He, T., Stankovic, J.A., Abdelzaher, T.F.: MMSN: Multi-frequency media access control for wireless sensor networks. In: Proc. of the 25th IEEE INFOCOM, pp. 1–13 (2006)
45. Zhu, C., Zheng, C., Shu, L., Han, G.: A survey on coverage and connectivity issues in wireless sensor networks. J. Netw. Comput. Appl. 35, 619–632 (2012)

Data Aggregation and Forwarding Route Control for Efficient Data Gathering in Dense Mobile Wireless Sensor Networks

Kazuya Matsuo, Keisuke Goto, Akimitsu Kanzaki, Takahiro Hara, and Shojiro Nishio

Osaka University, 1-5 Yamadaoka, Suita, Osaka, Japan
{matsuo.kazuya,goto.keisuke,kanzaki,hara,nishio}@ist.osaka-u.ac.jp

Abstract. This chapter presents a data gathering method considering geographical distribution of data values for reducing traffic in dense mobile wireless sensor networks. First, we present our previous method (DGUMA) which is a data gathering method that efficiently gathers sensor data using mobile agents in dense mobile wireless sensor networks. Second, we introduce an extended method of DGUMA, named DGUMA/DA (DGUMA with Data Aggregation), that exploits geographical distribution of data values in order to further reduce traffic. Finally, we analyze DGUMA/DA and confirm the effectiveness of the method through some simulation experiments.

Keywords: mobile agent, data gathering, data aggregation, forwarding route control.

1 Introduction

Recently, *participatory sensing*, where sensor data are gathered from portable sensor devices such as smart phones, has attracted much attention [2, 9, 14, 15]. In participatory sensing, it is general that sensor data are uploaded to the Internet through some infrastructures such as 3G and LTE networks. It is undesirable for participatory sensing to generate a large amount of traffic that may exhaust the limited channel bandwidth shared by a wide variety of applications. For this reason, *MWSNs (Mobile Wireless Sensor Networks)*, which are constructed by mobile sensor nodes held by ordinary people without any infrastructure [17], have recently attracted much attention as a way of realizing participatory sensing. In a MWSN, sensor readings are gathered to a sink through multi-hop wireless communication [5, 20].

In MWSNs constructed by mobile sensor nodes held by ordinary people, the number of sensor nodes is generally very large. For example, there are generally more than 10,000 people in the daytime in some stations in major cities such as Tokyo. When assuming that all of them hold sensor devices with Wi-Fi interface whose communication range is about 100[m], a sensor node can directly communicate with about 100 sensor nodes. In such an environment, an arbitrary geographical point in the sensing area can be sensed by many sensor nodes when an application monitors the geographical distribution of temperature in the station. We call this networks *dense MWSNs*.

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_5

On the other hand, most of MWSN applications require a certain geographical granularity of sensing in a specific area (e.g., sensor data of every 100[m]×100[m] square in a 1,000[m]×1,000[m] flatland) at every sensing time. In such a situation, if a sink gathers sensor data from all sensor nodes in the entire area, the network bandwidth and the battery of sensor nodes are unnecessarily wasted. Thus, it is desirable to efficiently gather sensor data from the minimum number of sensor nodes which are necessary to guarantee the geographical granularity required by the application. For this aim, as a data gathering method which efficiently gathers sensor data in dense MWSNs, *DGUMA (Data Gathering method Using Mobile Agents)* has been proposed [6]. DGUMA uses mobile agent, which is an application software that autonomously operates on a sensor node and moves among sensor nodes. Mobile agents are generated by the sink and allocated on sensor nodes located near the sensing points, which are determined from the requirement on geographical granularity. For gathering sensor data, DGUMA constructs a static tree-shaped logical network whose nodes are mobile agents (sensing points). Every time when the sensing time comes, sensor nodes where agents locate perform sensing and send the sensor data to the sink according to the tree-shaped network. By doing so, DGUMA can reduce the traffic for gathering sensor data since mobile agents control transmissions of sensor data.

Here, sensor readings on environmental information such as sound and temperature tend to have a same or similar value at adjacent sensing points. However, DGUMA does not consider such a characteristic of sensor readings, and gathers sensor readings acquired by all agents even when there are ones with the same value. If we can aggregate such sensor readings with the same value, further reduction of traffic for gathering sensor data can be expected [1, 10, 11, 13]. In addition, it is general that the geographical distribution of data values changes over time. In such a case, it is effective to dynamically construct communication routes (tree-shaped network in DGUMA) so that many sensor readings with the same value are aggregated. However, since DGUMA constructs a static tree-shaped network for gathering sensor data, effective data aggregation cannot be achieved in some geographical distribution of data values.

In this chapter, we present an extended method of DGUMA, named *DGUMA/DA (DGUMA with Data Aggregation)*, that considers the geographical distribution of data values in dense MWSNs. In DGUMA/DA, each mobile agent aggregates multiple readings with the same value in order to reduce the traffic for gathering sensor data. Moreover, at every sensing time, each mobile agent searches its adjacent agents which have the same reading during gathering sensor data by changing their direction of forwarding sensor data from lengthwise (up or down) to crosswise (right or left) or vice versa. When an adjacent agent which has the same reading is found, the mobile agent fixes its direction of forwarding sensor data so that the adjacent agent can continuously aggregate the readings. In addition, when the reading of the adjacent agent on the fixed direction becomes different, the mobile agent releases its direction of forwarding sensor data. This mechanism increases the chances of aggregating multiple readings with the same value. Using these two mechanisms (i.e., data aggregation and forwarding route control), DGUMA/DA further reduces the traffic for gathering sensor data to the sink.

Furthermore, we confirm the effectiveness of DGUMA/DA by comparing with DGUMA through a theoretical analysis and some simulation experiments.

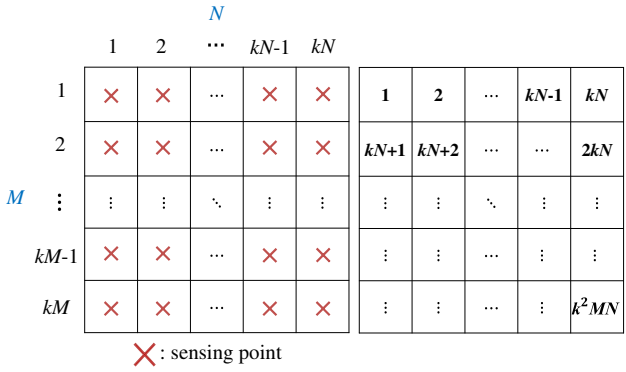


Fig. 1. A sensing area, sensing points and gridIDs

The remainder of this chapter is organized as follows. In Section 2, we describe assumptions in this chapter. In Section 3, we introduce related work. In Section 4, we describe our previous method (DGUMA). In Section 5, we explain the details of DGUMA/DA, which is an extended method of DGUMA. In Section 6, we discuss the performance gain of DGUMA/DA. In Section 7, we show the results of the simulation experiments. Finally, in Section 8, we summarize this chapter.

2 Assumptions

We assume dense MWSNs constructed by mobile sensor nodes which are held by ordinary people and equipped with a radio communication facility. These sensor nodes periodically observe (sense) a physical phenomenon (e.g., sound, temperature, and light), and communicate with each other using multi-hop radio communication. According to the requirement from an application, the sink periodically monitors the sensing area while guaranteeing the geographical granularity of sensing. More specifically, the sink gathers sensor readings from sensor nodes located near the sensing points which are determined from the requirement of the geographical granularity at the timing of sensing. We call the interval of data gathering the *sensing cycle*.

2.1 System Environment

The sensing area is assumed to be a two-dimensional plane whose horizontal to vertical ratio is $M : N$ (M and N are positive integers). The application specifies its requirement of the geographical granularity of sensing as an integer of $k^2 \cdot M \cdot N$ ($k = 1, 2, \cdots$). Then, the sink divides the sensing area into $k \cdot M \times k \cdot N$ lattice-shaped subareas (*grid*) and determines the center point of each grid as a sensing point, which is the target of data gathering. The sink assigns the *gridID* $\{1, 2, \cdots, k^2MN\}$ to each grid from left-upper grid to right-lower grid in order (see Fig. 1). Since no infrastructure for communication is available in the sensing area, the sink gathers sensor data by using a *MANET*

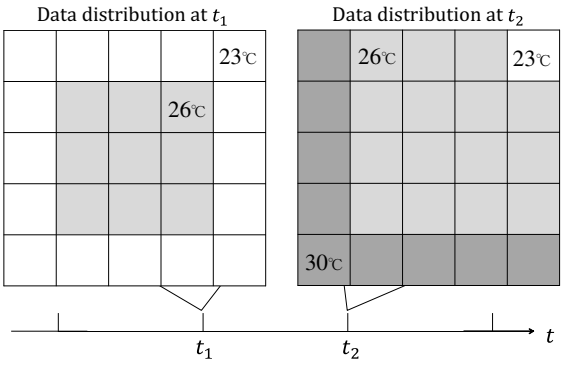


Fig. 2. An example of distribution of data values

(Mobile Ad Hoc Network) constructed by sensor nodes. The communication range of each sensor node is a circle with a radius of r. Each sensor node is equipped with a positioning device such as GPS, and communicates with other sensor nodes using *geo-routing*, which is a multi-hop radio communication based on their positions (the details are described in the next section). Each sensor node freely moves in the sensing area, while the sink is stationary. Since we assume that an enormous number of mobile nodes densely exist in the sensing area, there are many sensor nodes for each geographical point that can sense (cover) the point in the entire sensing area. Here, we assume that a sensing point is covered when at least one sensor node exists inside of a circle inscribed in the grid. We define this area the *valid area*, and the sensor reading sensed by the sensor nodes located in the valid area is defined as the *valid data*.

When assuming a physical phenomenon such as temperature in an urban area, the values of sensor readings observed at geographically close points tend to become the same with a high probability. In addition, the geographical distribution of data values changes as time passes. Fig. 2 shows an example of dynamic change in the geographical distribution of data values assuming temperature. In this figure, a number in a grid denotes the temperature observed in the corresponding grid. The same colored grids show that the same data values are observed. As shown in this example, the same data value tends to be observed at adjacent grids.

2.2 Geo-Routing

Sensor nodes adopt a geo-routing protocol based on that proposed in [7] to transport a message to the destination specified as a position (not a node). In this protocol, nodes perform a transmission process using the information on positions of the transmitter and the destination specified in the packet header. Specifically, the transmitter records the coordinates of the destination and itself into the packet header of the message, and broadcasts the message to its neighboring nodes. Each node which received this message judges whether it locates within the *forwarding area*. The forwarding area is determined based on the positions of the transmitter, the destination and the communication

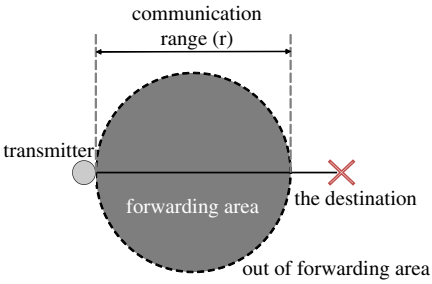


Fig. 3. Forwarding area

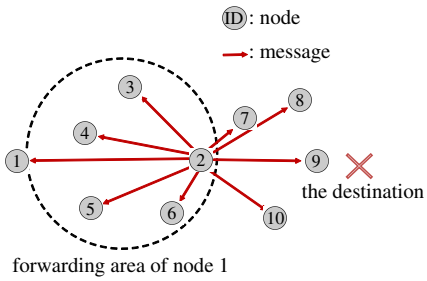


Fig. 4. An example of message forwarding in geo-routing

range, so that any node in the forwarding area is closer to the destination than the transmitter and can communicate directly to all nodes in the area (see Fig. 3). A node within the forwarding area sets the waiting time, and then it forwards the message after the waiting time elapses. The waiting time is set shorter as the distance between the node and the destination gets shorter. Each node within the forwarding area cancels its transmission process when it detects the message forwarded by another node. For example, in Fig. 4, nodes {2, . . ., 6} receive a message from node 1, and set the waiting time. Since node 2 is the closest node to the destination in the forwarding area of node 1, it first sends the message. On receiving this message, all nodes in the forwarding area of node 1 (i.e., nodes {3, . . . , 6}) cancel their transmission process. By repeating this procedure, the message is forwarded to nodes which are closer to the destination. If the transmitter node exists within half of the communication range ($r/2$) from the destination, each node which received the message sends an ACK to the transmitter node after the waiting time elapses instead of forwarding the message. As a result, the nearest node from the destination (which has first sent the ACK) can find that it is the nearest one because all nodes within $r/2$ from the destination can detect the ACK sent by the node and cancel sending it. If the transmitter node did not receive an ACK from any node, it also can find that the node itself is the nearest node.

3 Related Work

3.1 Location-Based Data Management in Dense MANETs

In [8], the authors proposed a data management method which realizes efficient access to location-based data in dense MANETs. In this method, nodes exchange data so that the data is always held by a node within the range of $r/2$ from the position corresponding to the data. By doing so, data access can be realized by using geo-routing. This method is similar to our method in the way that data (or agent) is held near a certain position. However, it is different from ours which aims data gathering to guarantee the geographical granularity of sensing specified from an application.

3.2 Data Gathering Utilizing Correlation of Data in Wireless Sensor Networks

In [16], the authors proposed a traffic reduction method utilizing temporal correlation of data in wireless sensor networks. In this method, each sensor node stores a sensor reading at last sensing time and compares a new sensor reading and the previous one at every sensing time. If the difference between the sensor readings is smaller than the threshold predetermined by the sink, the node does not send its sensor reading to the sink at the sensing time. Thus, the traffic for data gathering is reduced by utilizing temporal correlation of data. It is different from ours which reduces traffic by using the geographical distribution of data values. However, it is similar to ours in the way that the node does not send its sensor reading if the sensor reading corresponds to another one.

In [19, 21–24], the authors proposed data gathering methods which construct an overlay network to effectively aggregate sensor data using the spatial correlation of data in wireless sensor networks. In [19], sensor data are gathered using a routing tree, which is a combination of the minimum spanning tree and the shortest path tree. Raw data is sent according to the former tree and aggregated data is gathered according to the latter tree. In [21], an energy efficient routing tree is constructed using game theory that considers transmission energy, the effect of wireless interference, and the opportunity for aggregating correlated sensor data. In [22], clusters are constructed based on the compression ratio, which indicates the ratio that a node compresses the data received from its neighbors using spatial correlation. In [23], a routing tree is dynamically constructed. In this routing tree, each node sends its holding packet to its neighbors which hold more packets and exist closer to the sink. This is because a sensor node can have more opportunities to aggregate sensor data when holding more packets. In [24], assuming a circular sensing area, a data gathering method using mobile sensor nodes is proposed. This method divides the area into polar grids. In each grid, a node is chosen in each grid to aggregate data observed in the grid and to join a routing tree. These methods are similar to ours in the way that sensor data are gathered using spatial correlation. However, these methods do not consider the change in the geographical distribution of data values.

In [18], assuming that sensor nodes which have spatial correlated data tend to exist close to each other, and continuous queries which specify a condition on a sensor reading to be gathered, a dynamic route construction method for the queries is proposed. This method detects sensor nodes which satisfy the condition, and constructs some clusters consisting of sensor nodes which exist close to each other. For gathering sensor data, a minimum spanning tree is constructed by those clusters. When sensor nodes which satisfy the condition are changed by the change in the distribution of data value, the sink computes a new minimum spanning tree and informs it to all sensor nodes. This method is similar to ours in the way that routing tree is constructed dynamically according to the distribution of data value. However, this method is conducted in a centralized manner.

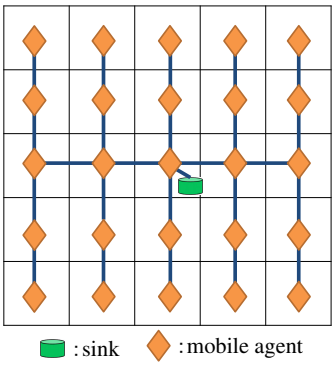


Table 1. Forwarding directions of agent data

Location of the sensor node that previously forwarded the agent data	Forwarding directions of the agent data
-(sink)	up, down, right, left
grid next to right	up, down, left
grid next to left	up, down, right
grid next to down	up
grid next to up	down

Fig. 5. A forwarding tree in DGUMA

4 DGUMA: Our Previous Method

In this section, we briefly describe our previous method, DGUMA [6], which is a sensor data gathering method using mobile agents.

4.1 Mobile Agent

A mobile agent is an application software which autonomously operates on a sensor node and moves between sensor nodes. A sensor node boots a mobile agent by referring to the agent data, which consists of the information on geographical granularity of sensing, the sensing cycle, and the position of the sink. The role of mobile agents is to transmit sensor data to the sink at every sensing time. Mobile agents are deployed on sensor nodes so that they can guarantee the requirement from the application regarding the geographical granularity of sensing according to the procedure described in Section 4.2.

4.2 Deployment of Mobile Agents

In DGUMA, the sink sends the agent data, which consists of information on the geographical granularity of sensing, the sensing cycle, and the position of the sink, along the static tree-shaped network created based on the geographical relationships among sensing points (see Fig. 5) according to the following procedure.

First, the sink generates the agent data and sends it to the sensing point in the grid where the sink itself locates using the geo-routing protocol described in Section 2.2. When the sensor node located at the closest position of the sensing point receives the agent data, it boots a mobile agent. As the initial operation, the mobile agent retransmits the agent data to the sensing points in the adjacent grids existing in the directions shown in Table 1. This retransmission of the agent data is repeated until mobile agents in grids on top and bottom edges of the sensing area are booted.

By doing so, a tree-shaped network whose root is the sink and nodes are mobile agents is constructed. We call this network the *forwarding tree*.

4.3 Movement of Mobile Agent

In DGUMA, if a sensor node on which a mobile agent operates moves away from the sensing point, the mobile agent moves from the current sensor node to another sensor node which locates closest to the corresponding sensing point. Specifically, a mobile agent starts moving when the distance between the sensing point and itself becomes longer than the threshold. This threshold is a system parameter which is set as a constant smaller than $r/2$ and the radius of the valid area for sensing, which can guarantee that a sensor node on which a mobile agent operates can communicate with all sensor nodes located near (within $r/2$) from the sensing point and can sense the data at the sensing point. In order to move to the sensor node located closest to the sensing point, the mobile agent issues a message containing the agent data, and broadcasts it to neighboring nodes within $r/2$ from the sensing point. As in Section 2.2, the sensor node located closest to the sensing point first sends an ACK and boots a mobile agent. Other sensor nodes cancel to send the same ACK because they can detect this ACK. Also, the original mobile agent stops its operation when detecting the ACK.

4.4 Transmission of Sensor Data

Mobile agents deployed at (near) sensing points send the sensor data held by the sensor nodes on which these agents operate to the sink at every sensing time. Here, a sensor data consists of a sensor reading and a gridID.

First, mobile agents located in grids of the top and bottom edges in the sensing area start to send their sensor data to their parents in the forwarding tree. In doing so, the geo-routing protocol is used. Each mobile agent can receive the sensor data from its children in the forwarding tree, because it keeps its own position within $r/2$ from the sensing point according to the procedure described before. When mobile agents except for that located in the grid where the sink exists receive the sensor data from all their children, they pack the received sensor data and their own sensor readings in a packet as their own sensor data, and forward the sensor data to their parents. This procedure is repeated until the mobile agent located in the grid where the sink exists receives sensor data from all its children. Finally, the mobile agent located in the grid where the sink exists packs all the received sensor data and its own sensor reading in a packet, and forwards it to the sink.

Fig. 6 shows an example of the above procedure. First, mobile agents at grids $\{1, \ldots, 6\}$ send a packet with their own sensor data (i.e., their own sensor readings and gridIDs) to their parents. On receiving the packet from the mobile agent at grid 1, the mobile agent at grid 7 sends the packet after adding its sensor data. Mobile agent at grid 12 performs in the same way as that at grid 7. The mobile agent at grid 8 receives multiple packets from grids 2 and 7, and packs sensor data included in these packets and its sensor data in a packet. Mobile agents at grids 9 and 11 perform in the same way as that at grid 8. Also, the agent at grid 10 receives multiple packets from grids $\{4, 9, 11\}$, packs them, and sends the packet with the aggregated sensor data to the sink.

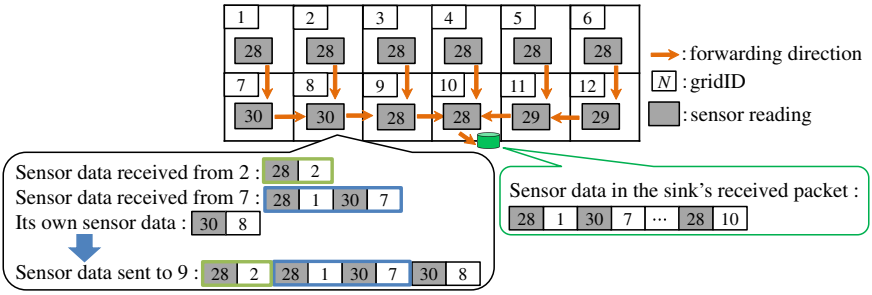


Fig. 6. An example of transmission of sensor data in DGUMA

5 DGUMA/DA: The Extended Method

As discussed in Section 1, DGUMA does not consider geographical distribution of data values, and gathers sensor readings acquired by all agents even when there are ones with the same value. In addition, DGUMA constructs a static forwarding tree for gathering sensor data. Considering dynamic change in the geographical distribution of data values, it is expected that traffic can be further reduced if we control the topology of the forwarding tree according to the data distribution.

In DGUMA/DA, each mobile agent aggregates multiple readings with the same value in order to reduce the traffic for gathering sensor data. Moreover, it dynamically constructs the forwarding tree so that more sensor readings can be aggregated when gathering sensor data.

5.1 Outline

Similar to DGUMA, DGUMA/DA deploys the agent data to $k^2 \cdot M \cdot N$ sensing points. Each mobile agent sets up *timer*, which will be described in Section 5.2. When the timer expires, the mobile agent sends sensor data to the sink. In this process, each mobile agent aggregates multiple readings with the same value as described in Section 5.3. In addition, DGUMA/DA dynamically constructs the forwarding tree according to the procedure described in Section 5.4.

When the sink receives the sensor data, it restores aggregated sensor readings, according to the procedure described in Section 5.5.

5.2 Timer Setting

In DGUMA, each mobile agent sends sensor data after receiving those from all its children. On the other hand, DGUMA/DA dynamically changes the forwarding tree. This means that the set of children for each mobile agent dynamically changes. Thus, each mobile agent cannot recognize whether its child(ren) exist(s) or not. In order to gather sensor data to the sink even in that situation, DGUMA/DA sets the timer for each mobile agent. Each mobile agent starts to send its sensor data to its parent when its timer has expired at each sensing time.

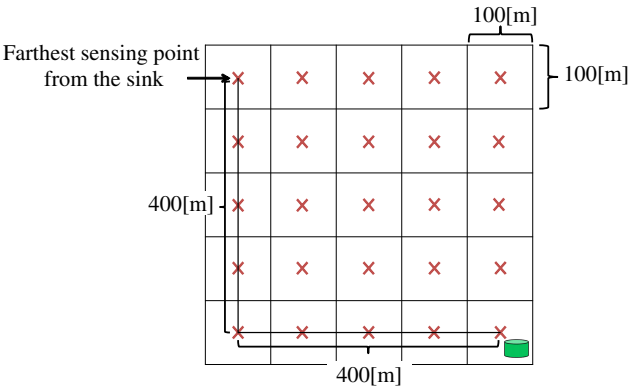


Fig. 7. An example of relationship of timer and the location of the sink

The timer, T, is determined by the following equation:

$$T = \frac{Dist_{max} - (|x_{sink} - x| + |y_{sink} - y|)}{Grain} \cdot (t_a + rand_{max}) + rand. \quad (1)$$

Here, t_a is the time required to send data to the adjacent agent, (x_{sink}, y_{sink}) is the coordinate of the sink, (x, y) is the coordinate of the sensing point, *Grain* is the distance between adjacent sensing points, and $Dist_{max}$ is the sum of the distances in x- and y-direction between the sensing point in the grid where the sink exists and that of the farthest sensing point from the sink. For example in Fig. 7, $Dist_{max}$ is 800[m] because the farthest sensing point from the sink is the top left one. *rand* is a random number within the range [0, $rand_{max}$] to avoid packet collision. By setting the timer for each mobile agent according to Eq.(1), mobile agents send sensor data in descending order of distance from the sink on the forwarding tree without any knowledge about their children.

5.3 Transmission of Sensor Data

Similar to DGUMA, each mobile agent sends a packet which contains multiple sensor data. However, unlike DGUMA, multiple gridIDs can be attached to a sensor reading. Mobile agents aggregate sensor readings using this packet structure.

At each sensing time, each mobile agent whose timer has expired puts its own sensor reading and gridID into a packet, and sends the packet to its parent. Here, the parent of the mobile agent is determined according to the forwarding route control described in Section 5.4. When a mobile agent received packets from other mobile agents before its timer expires, it aggregates its own sensor data and those in the received packets (details are described below), puts the aggregated sensor data into a packet, and sends the packet to its parent (or the sink). Here, since the timer of a mobile agent is set so as to expire after those of its descendants in the forwarding tree, every mobile agent can receive sensor data from all its children before its timer expires.

Each mobile agent aggregates sensor data according to the following procedure:

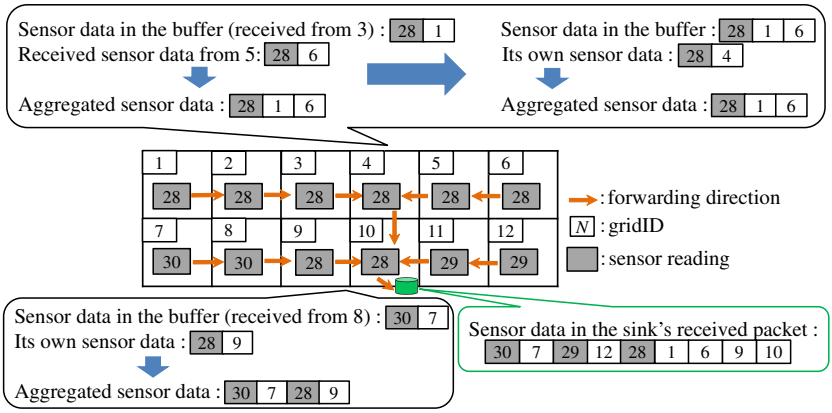


Fig. 8. An example of aggregating data values

(1) When a mobile agent has received multiple packets, it first copies sensor data included in the packet first received to its buffer.

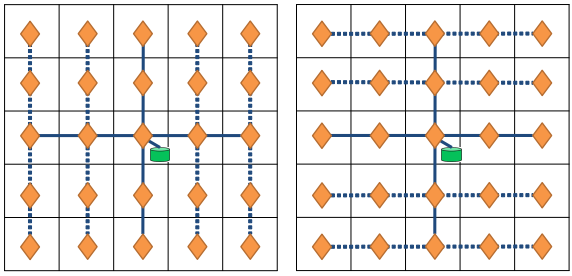
(2) It checks whether there is a sensor reading in another received packet, which is identical with that in the buffer. If so, the mobile agent adds only the gridIDs of the sensor reading after the gridIDs of the corresponding reading in the buffer. After that, the mobile agent adds other sensor data, whose sensor reading is not identical with any of those in the buffer, to the end of buffer.

(3) When its timer expires, the mobile agent adds its own sensor data to the buffer in the same way. If its sensor reading is identical with one of them in the buffer, it adds only its gridID after the gridIDs of the corresponding reading in the buffer and moves the corresponding sensor data to the end of the buffer. After that, the agent creates a new packet that contains all sensor data in the buffer, and sends the packet to its parent.

Here, when the mobile agent has received only one packet, or when there is only one sensor data in the buffer, it checks whether its own sensor reading and that at the end of the buffer is identical. If so, the mobile agent adds neither its own sensor reading nor its gridID to the buffer.

According to the above procedure, each mobile agent can aggregate sensor readings by adding only its gridID when there is a sensor reading identical with its own reading in the received packet. In addition, when the sensor readings become identical between adjacent sensing points on the forwarding tree, more traffic can be reduced since nothing is added to the packet. Note that all sensor readings can be restored at the sink from the aggregated sensor data. The details are presented in Section 5.5.

Fig. 8 shows an example of the above procedure. First, mobile agents at grids {1, 6, 7, 12} send a packet with its own sensor data (i.e., their own sensor readings and gridIDs) to their parents. On receiving the packet from the mobile agent at grid 1, the mobile agent at grid 2 copies the received sensor data to its buffer. After the expiration of its timer, the agent at grid 2 sends the packet without adding its sensor data (its own



(a) Lengthwise tree (b) Crosswise tree

Fig. 9. Two fundamental trees in DGUMA/DA

sensor reading and gridID) because its own sensor reading is identical with that at the end of the buffer. Mobile agents at grids {3, 5, 8, 11} perform in the same way as that at grid 2. On the other hand, after copying sensor data received from the agent at grid 8 to its buffer, the mobile agent at grid 9 adds its sensor data to the end of the buffer because there is no sensor data with the same sensor reading as its own reading (i.e., 28). Then, the mobile agent at grid 4 receives multiple packets from grids 3 and 5, and aggregates sensor data included in these packets according to the procedure described in step (2). After the aggregation, the buffer contains only one sensor data whose sensor reading is identical with its own reading (i.e., 28). Thus, the mobile agent sends the packet without adding its sensor data. Also, the agent at grid 10 receives multiple packets from grids {4, 9, 11}, aggregates them, and sends the packet with the aggregated sensor data to the sink.

5.4 Forwarding Route Control

DGUMA/DA dynamically constructs the forwarding tree based on two fundamental trees, the *lengthwise tree* and the *crosswise tree* (see Fig. 9) in order to aggregate more sensor readings. Here, the lengthwise tree has the same topology as the forwarding tree for deploying mobile agents. DGUMA/DA switches between the two fundamental trees at every sensing time, and searches routes on which sensor readings have the same value. By doing so, DGUMA/DA can construct the forwarding tree by which more sensor readings are aggregated. Here, mobile agents at the grids which are on the same column or row of the grid where the sink locates (solid lines in Fig. 9) do not change their forwarding directions.

The detailed procedure is as follows:

(1) At every sensing time, each mobile agent changes its forwarding direction, and sends a packet following the procedure in Section 5.3. Note that the lengthwise tree is used at the first sensing time.

(2) When each mobile agent receives a packet, the mobile agent checks whether the sensor reading at the end of the received packet is identical with its own reading. If

so, and the path from the child to itself is not fixed, it sends a *route fix message* to the child. On the other hand, if its own sensor reading is not identical with that of the end of the received packet, and when the route from the child to itself is fixed, the mobile agent sends a *fixed-route release message* to the child.

(3) When a mobile agent receives a route fix message from its current parent, it fixes the route from itself to the parent. After that, it does not change its forwarding direction at the subsequent sensing times.

(4) When a mobile agent receives a fixed-route release message, it stops fixing the route. After that, it restarts to change its forwarding direction at the subsequent sensing times.

By doing so, DGUMA/DA can construct the forwarding tree which aggregates more sensor readings even when the geographical distribution of data values dynamically changes.

Fig. 10 shows the detailed procedure of the forwarding route control. In this figure, grids with the same color indicate the sensor readings with the identical value. In Fig. 10(a), at the first sensing time, each mobile agent sends a packet using the lengthwise tree according to the procedure described in Section 5.3. In this phase, the agent at grid 6 sends the route fix message to its child, the agent at grid 1, because its sensor reading is identical with that of grid 6. On receiving this message, the agent at grid 1 fixes the route to the agent at grid 6. The agents in the other grids perform in the same way. As a result, the red-colored lengthwise routes as shown in Fig. 10(b) are fixed in this phase. At the second sensing time, each mobile agent without receiving route fix message switches its forwarding direction from lengthwise to crosswise, and sends a packet using the forwarding tree in Fig. 10(c). In this phase, the agent at grid 7 sends the route fix message to the agent at grid 6 because its sensor reading is identical with that of grid 7. The agent at grid 6 fixes the route to the agent at grid 7. After performing this procedure at other grids, the red-colored crosswise routes as shown in Fig. 10(d) are also fixed.

5.5 Restoring Sensor Readings at the Sink

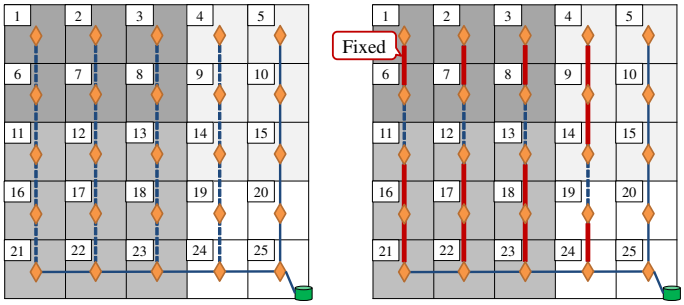
In DGUMA/DA, some sensor readings can be excluded from the packet. Thus, the sink needs to restore these excluded sensor readings.

In order to do this, the sink maintains the *tree information*, which stores the information on the topology of the forwarding tree. Note that the topology in the tree information is initially set as the lengthwise tree. The sink updates the tree information while restoring excluded sensor readings by referring to sensor data in the received (aggregated) packet. The detailed procedure is as follows:

(1) The sink extracts sensor readings with gridIDs from sensor data in the received packet and assigns them to the corresponding grids.

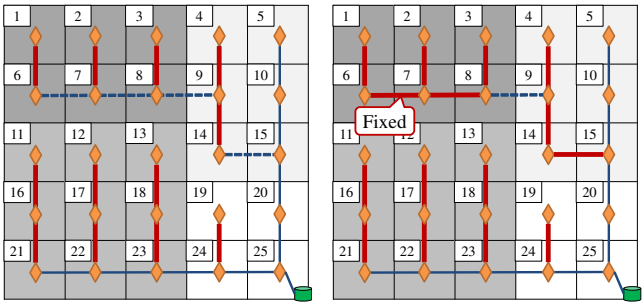
(2) For each grid without sensor reading, the sink assigns the sensor reading of the grid which is the child of the corresponding grid in the current tree information. This step is repeated until sensor readings are assigned to all grids.

(3) The sink recognizes that the routes have been fixed between grids where step (2) is applied. Thus, the sink fixes these routes in the tree information. For other routes, the sink switches the directions from lengthwise to crosswise or vise versa.



(a) Forwarding tree at the 1st sensing time.

(b) Route fixing after the 1st sensing time.



(c) Forwarding tree at the 2nd sensing time.

(d) Route fixing after the 2nd sensing time.

Fig. 10. An example of forwarding route control

(4) If the sensor readings become different between grids on a fixed route, the sink recognizes that the fixed route has been released. Thus, the sink stops fixing the corresponding route in the tree information, and switches the forwarding direction at the next sensing time.

According to the above procedure, the sink can recognize the topology of the forwarding tree and restore sensor readings at all grids.

Fig. 11 shows an example of the above procedure when the sink received sensor data in the environment shown in Fig. 8. First, the sink extracts sensor readings at grids {1, 6, 7, 9, 10, 12} from the received (aggregated) packet. Next, the sensor reading at grid 2 is restored by referring to its current child (i.e., grid 1). The sensor reading at grid 3 is also restored by referring to that at its current child (i.e., grid 2). In addition, The sink fixes the routes between these grids (i.e., from 1 to 2, and from 2 to 3), where the restoring process is applied. By applying this procedure for other grids {4, 5, 8, 11}, the sink can restore all the sensor readings. Note that, routes on the same column or row of the grid where the sink locates (i.e., from 7 to 10, from 12 to 10, and from 4 to 10) are not fixed.

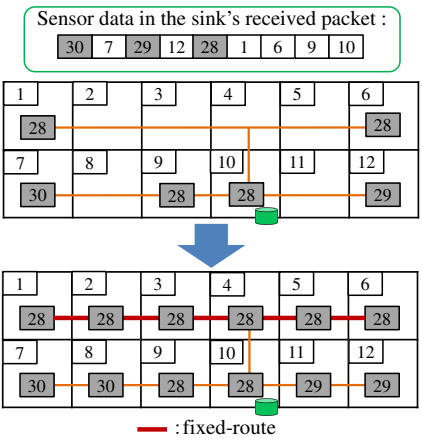
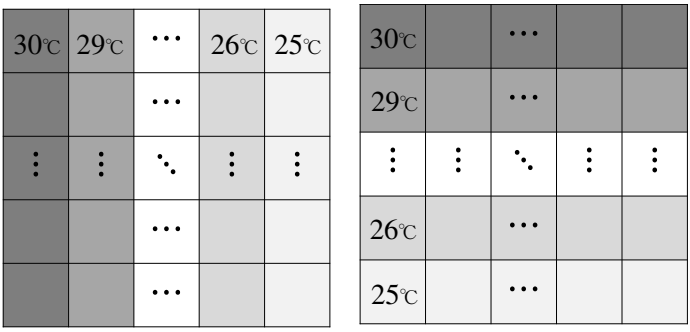


Fig. 11. An example of restoring sensor readings



(a) Lengthwise distribution (b) Crosswise distribution

Fig. 12. Geographical distribution of data values

6 Discussion

DGUMA/DA can reduce traffic for data gathering by data aggregation and forwarding route control. On the other hand, the forwarding route control generates the overhead since it needs to send route fix and fixed-route release messages. In this section, we discuss the performance gain (traffic reduction) and the overhead by the forwarding route control. Here, it is difficult to cover all situations of distribution of data values. In order to simplify the discussion, we assume a situation in which the distribution changes from Fig. 12(a) (*lengthwise distribution*) to Fig. 12(b) (*crosswise distribution*). In the lengthwise distribution, sensor readings can be aggregated efficiently without forwarding route control (only using the lengthwise tree). On the other hand, in the crosswise distribution, all routes between grids need to change (the forwarding tree needs to change to the crosswise tree) in order to efficiently aggregate sensor readings.

Table 2. The variables in this section

meaning of variable	variable
The size of packet header	s_{header}[B]
The size of sensor reading	s_{data}[B]
The size of gridID	s_{ID}[B]
The size of ACK	s_A[B]
The average distance of 1-hop transmission	l[m]
The average of the number of hops between the adjacent agents	h

In other words, the performance gain and the overhead generated by the forwarding route control become the largest in this situation.

For the discussion, we assume a D[m]×D[m] flatland as the sensing area. The sink divides the area into G lattice-shaped grids whose size is $D/\sqrt{G}$[m] × $D/\sqrt{G}$[m]. In addition, variables in Table 2 are used in this section. h in Table 2 is expressed by the following equation:

$$h = \begin{cases} 1 & (\frac{D}{\sqrt{G} \cdot l} \leq 1). \\ \frac{D}{\sqrt{G} \cdot l} & (Otherwise). \end{cases} \tag{2}$$

We assume that the mobile agent located in the grid where the sink exists can communicate directly to the sink. The sink locates at the grid of nth row and mth column.

Assuming the above situation, we calculate the following theoretical values:

- The overhead which is generated by changing the forwarding tree from Fig. 13(b) to Fig. 13(c).
- The traffic which is generated by data gathering in Fig. 13(a).
- The traffic which is generated by data gathering in Fig. 13(c).
- The traffic which is generated by data gathering in Fig. 13(b).

6.1 Overhead Generated by the Forwarding Route Control

We assume that all lengthwise routes are fixed in Fig. 13(b). In this case, all the fixed routes are released in the data gathering process with the lengthwise tree. The number of released routes is $(\sqrt{G}-1)^2$ because routes on nth row and those on mth column are not fixed. Considering that the size of a fixed-route release message is equal to s_{header}[B], and that the message is sent using the geo-routing protocol (shown in Fig 14), the traffic generated when releasing a route becomes $(s_{header} \cdot h + s_A)$[B]. Thus, the total traffic for releasing fixed routes, $S_{release}$, is expressed by the following equation:

$$S_{release} = (\sqrt{G} - 1)^2 (s_{header} \cdot h + s_A)\,[\mathrm{B}]. \tag{3}$$

At the next sensing time, the forwarding route becomes the crosswise tree. When gathering data with the crosswise tree, all routes are fixed except for those on nth row or mth column. Thus, the number of fixed routes is $(\sqrt{G} - 1)^2$. Considering that the size

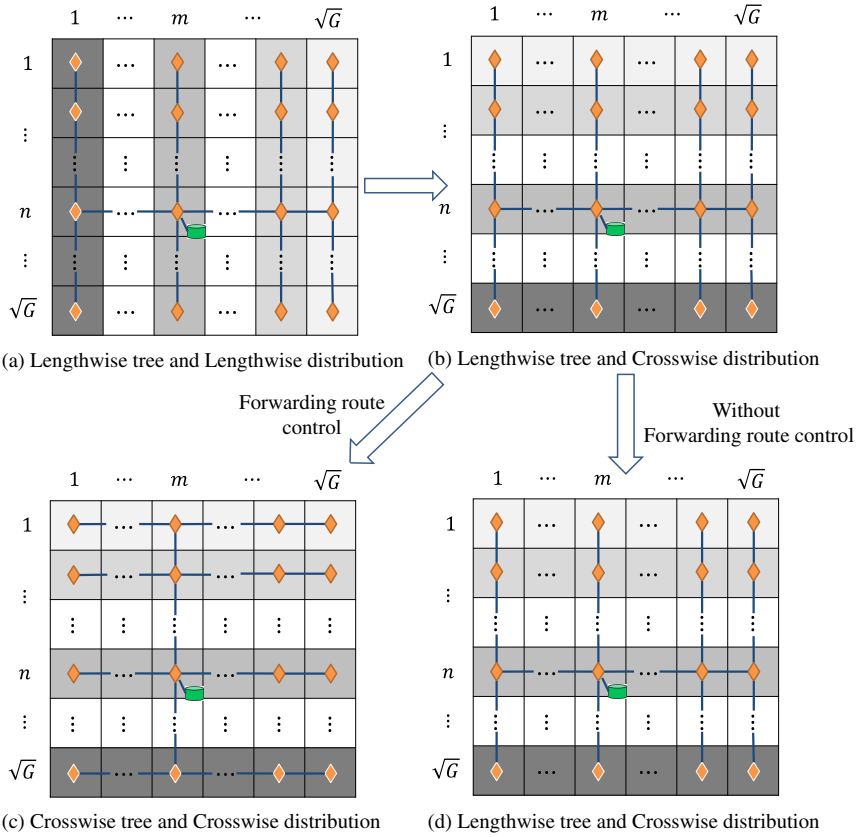


Fig. 13. Forwarding tree and distribution of data values

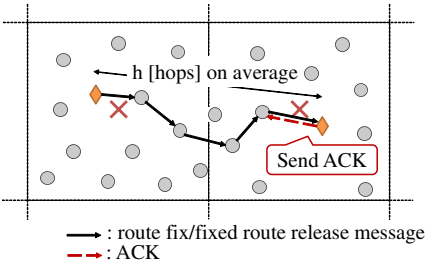


Fig. 14. An example of sending a message using the geo-routing

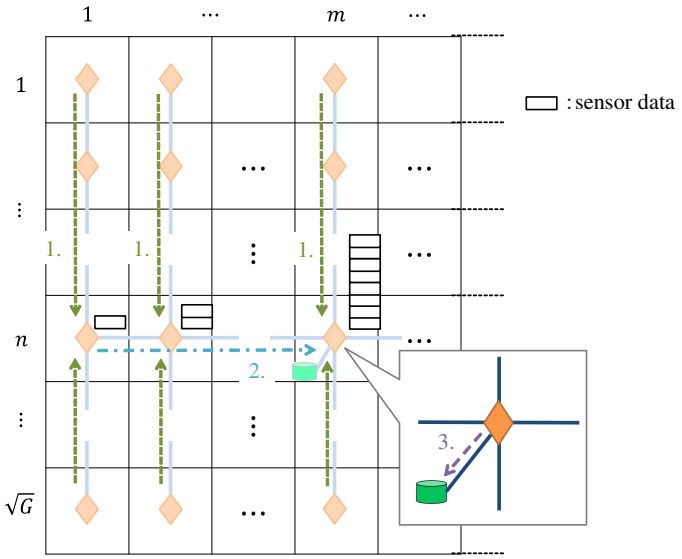


Fig. 15. An example of data gathering using lengthwise tree in the lengthwise distribution

of a route fix message is also equal to s_{header}[B], and that these messages are sent using the geo-routing protocol, the total traffic for fixing routes, S_{fix}, is expressed by the following equation:

$$S_{fix} = (\sqrt{G} - 1)^2(s_{header} \cdot h + s_A)\,[\mathrm{B}]. \tag{4}$$

As a result, the total overhead generated by the forwarding route control, $S_{overhead}$, is derived by the following equation:

$$S_{overhead} = S_{release} + S_{fix} = 2(\sqrt{G} - 1)^2(s_{header} \cdot h + s_A)\,[\mathrm{B}]. \tag{5}$$

6.2 Traffic for Data Gathering Using Lengthwise Tree in the Lengthwise Distribution

In this case, the data gathering tree can be treated as *optimized*. In order to derive the theoretical value of the traffic for data gathering, S_{opt}, we separately derive traffic in the following steps (see Fig. 15):

1. Lengthwise packet transmission (at every column).
2. Crosswise packet transmission (at nth row).
3. Packet transmission from the mobile agent to the sink at the grid where the sink locates.

STEP1. Lengthwise Packet Transmission: At first, a mobile agent located in a grid of the top or bottom edge in the sensing area starts to send its packet to its parent in the

forwarding tree. The size of this packet is $(s_{header} + s_{data} + s_{ID})$[B]. So, the traffic for delivering this packet to the parent becomes $(h \cdot (s_{header} + s_{data} + s_{ID}) + s_A)$[B].

The parent sends the packet to its parent without adding its sensor reading nor gridID because its own sensor reading is identical with that included in the received packet. Thus, the traffic for delivering this packet to the next parent becomes the same, that is, $(h \cdot (s_{header} + s_{data} + s_{ID}) + s_A)$[B].

This packet is delivered in the same way until a mobile agent at the grid on n-th row receives it. Since the number of mobile agents which send the packet is $(\sqrt{G} - 1)$ in a column, the total traffic generated at this column, $S_{opt,col}$, is expressed by the following equation:

$$S_{opt,col} = ((\sqrt{G} - 1)\{(s_{header} + s_{data} + s_{ID})h + s_A\})\,[\mathrm{B}].$$

Considering that there are $\sqrt{G}$ columns in the sensing area, the total traffic generated by mobile agents for this phase, $S_{opt,step1}$, is expressed by the following equation:

$$S_{opt,step1} = \sqrt{G} \cdot S_{opt,col} = (\sqrt{G}(\sqrt{G} - 1)\{(s_{header} + s_{data} + s_{ID})h + s_A\})\,[\mathrm{B}].$$

STEP2. Crosswise Packet Transmission (At *n*th Row): A mobile agent at the left end or the right end grid of nth row receives two packets from its children (at upper and lower grids). The agent sends the packet to its parent without adding its sensor reading nor gridID because its own sensor reading is identical with that included in the packet. Thus, the size of packet transmitted by the agent becomes $(s_{header} + s_{data} + 2s_{ID})$[B], and the traffic for delivering this packet to its parents becomes $(h \cdot (s_{header} + s_{data} + 2s_{ID}) + s_A)$[B].

On the other hand, mobile agents except for those at the left- and right-end grids receive multiple packets with different sensor readings, and aggregate them. Let us focus on the parent of the agent at the left end grid (shown in Fig. 16). This agent receives three packets from its children at the upper, the lower, and the left grids. Here, since sensor readings in the packets received from upper and lower grids are identical, the size of sensor data in the buffer is $(s_{data} + 2s_{ID})$[B] after aggregating these packets. On the other hand, since the sensor reading in the packet from the left grid is different from that in the buffer, sensor data whose size is $(s_{data} + 2s_{ID})$[B] is added to the buffer. In addition, the mobile agent adds only its own gridID after the gridIDs with the sensor reading which is identical with its own reading. As a result, the agent creates a packet whose size is $(s_{header} + 2(s_{data} + 2s_{ID}) + s_{ID})$[B]. Thus, the size of a new created packet increases by $((s_{data} + 2s_{ID}) + s_{ID})$[B] at the agent at each grid on nth row. Therefore, the size of packet transmitted by the agent at the kth grid from the left end becomes $(s_{header} + k(s_{data} + 2s_{ID}) + (k - 1)s_{ID})$[B], and the traffic for delivering this packet to its parent becomes $(h \cdot \{s_{header} + k(s_{data} + 2s_{ID}) + (k - 1)s_{ID}\} + s_A)$[B].

There are $(m - 1)$ grids from the left end grid to the mth grid. Thus, the total traffic generated at these grids, $S_{opt,step2(left)}$, is expressed by the following equation:

$$S_{opt,step2(left)} = (h \cdot \left\{(m-1)s_{header} + \sum_{k=1}^{m-1} k(s_{data} + 2s_{ID}) + \sum_{k=1}^{m-1}(k-1)s_{ID}\right\} + (m-1)s_A)\,[\mathrm{B}].$$

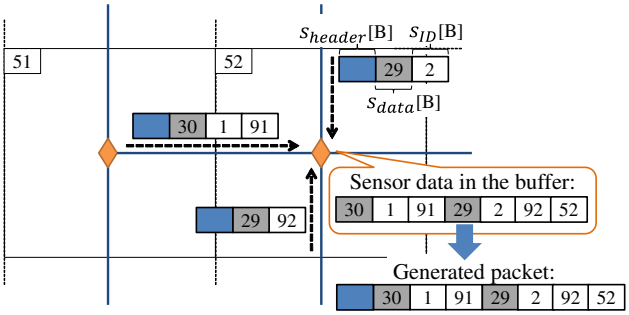


Fig. 16. An example of aggregation by the parent of the agent at the left-end grid in the lengthwise distribution (G=100, n=5, m=5)

In the same way, the total traffic generated at ($\sqrt{G} - m$) grids between the right end and mth grids, $S_{opt,step2(right)}$, is expressed by the following equation:

$$S_{opt,step2(right)} = (h \cdot \left\{ (\sqrt{G} - m)s_{header} + \sum_{k=1}^{\sqrt{G}-m} k(s_{data} + 2s_{ID}) + \sum_{k=1}^{\sqrt{G}-m} (k-1)s_{ID} \right\} + (\sqrt{G} - m)s_A)\,[\mathrm{B}].$$

STEP3. Packet Transmission from the Mobile Agent to the Sink at the Grid Where the Sink Locates: As shown in Fig. 17, the mobile agent at the grid where the sink locates records different $\sqrt{G}$ readings, 2 $\sqrt{G}$ gridIDs of the top and bottom grids in the sensing area, and ($\sqrt{G} - 2$) gridIDs of grids in nth row except for those of the left- and right-end grids, in the buffer. Thus, the size of the packet transmitted by this agent becomes ($s_{header} + \sqrt{G} \cdot s_{data} + (3\sqrt{G} - 2)s_{ID}$)[B]. Since we assume that this agent can directly communicate with the sink, the traffic generated at this grid, $S_{opt,step3}$, is expressed by the following equation:

$$S_{opt,step3} = (s_{header} + \sqrt{G} \cdot s_{data} + (3\sqrt{G} - 2)s_{ID} + s_A)\,[\mathrm{B}].$$

Consequently, the total traffic, S_{opt}, is expressed by the following equation:

$$\begin{aligned} S_{opt} &= S_{opt,step1} + S_{opt,step2(left)} + S_{opt,step2(right)} + S_{opt,step3} \\ &= h(G - \sqrt{G})(s_{data} + s_{ID}) + h(\sum_{k=1}^{m-1} k + \sum_{k=1}^{\sqrt{G}-m} k)(s_{data} + 2s_{ID}) \\ &+ h(\sum_{k=1}^{m-1}(k-1) + \sum_{k=1}^{\sqrt{G}-m}(k-1))s_{ID} + h(G-1)s_{header} \\ &+ s_{header} + \sqrt{G} \cdot s_{data} + (3\sqrt{G} - 2)s_{ID} + G \cdot s_A\,[\mathrm{B}]. \end{aligned} \quad (6)$$

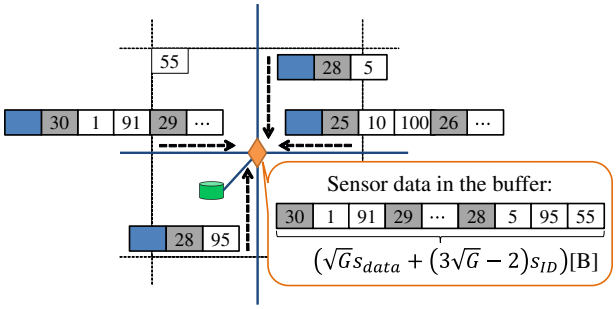


Fig. 17. An example of aggregation at the grid where the sink locates in the lengthwise distribution (G=100, n=5, m=5)

6.3 Traffic for Data Gathering Using Crosswise Tree in the Crosswise Distribution

In this case, the relation between the forwarding tree and the distribution of data values is the same as that in Section 6.2. Thus, the total traffic becomes S_{opt}, which is expressed by Eq.(6).

6.4 Traffic Generated by Data Gathering Using Lengthwise Tree in the Crosswise Distribution

In the same way as described in Section 6.2, in order to derive the theoretical value of the traffic for data gathering in this situation, S_{no_opt}, we separately derive traffic in the following steps (see Fig. 18):

1. Lengthwise packet transmission (at every column).
2. Crosswise packet transmission (at nth row).
3. Packet transmission from the mobile agent to the sink at the grid where the sink locates.

STEP1. Lengthwise Packet Transmission: At first, a mobile agent located in a grid of the top or bottom edge in the sensing area starts to send its packet to its parent in the forwarding tree. Since the size of this packet is $(s_{header} + s_{data} + s_{ID})$[B], the traffic for delivering this packet to the parent becomes $(h \cdot (s_{header} + s_{data} + s_{ID}) + s_A)$[B].

The parent adds its own sensor reading and gridID to the end of the received packet because its own sensor reading is different from that included in the received packet. Thus, the size of a new created packet increases by $(s_{data} + s_{ID})$[B] at every agent. Therefore, the size of packet transmitted by the agent at the kth grid from the grid of the top or bottom edge becomes $(s_{header} + k(s_{data} + s_{ID}))$[B], and the traffic for delivering this packet to its parent becomes $(h \cdot \{s_{header} + k(s_{data} + s_{ID})\} + s_A)$[B].

There are $(n - 1)$ grids from the top edge to the n-th row on a column. Thus, the total traffic generated at these grids becomes $(h \cdot \left\{(n-1)s_{header} + \sum_{k=1}^{n-1} k(s_{data} + s_{ID})\right\} +$

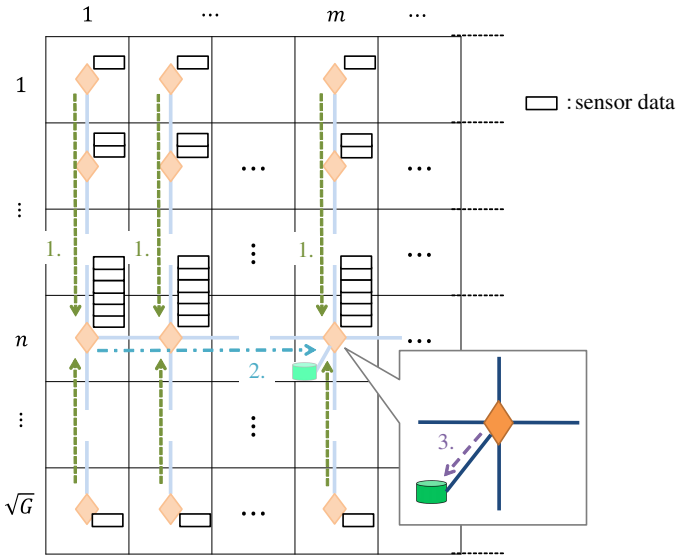


Fig. 18. An example of data gathering using lengthwise tree in the crosswise distribution

$(n-1)s_A)$[B]. In the same way, the total traffic generated at $(\sqrt{G}-n)$ grids between the bottom edge and nth row becomes $(h\cdot\left\{(\sqrt{G}-n)s_{header}+\sum_{k=1}^{\sqrt{G}-n}k(s_{data}+s_{ID})\right\}+(\sqrt{G}-n)s_A)$[B]. Therefore, the total traffic generated at a column, $S_{no\_opt,col}$, is expressed by the following equation:

$$S_{no\_opt,col}=(h\cdot\left\{(\sqrt{G}-1)s_{header}+(\sum_{k=1}^{\sqrt{G}-n}k+\sum_{k=1}^{n-1}k)(s_{data}+s_{ID})\right\}+(\sqrt{G}-1)s_A)\,[\mathrm{B}].$$

Considering that there are $\sqrt{G}$ columns in the sensing area, the total traffic generated by mobile agents in this phase, $S_{no\_opt,step1}$, is expressed by the following equation:

$$\begin{aligned}S_{no\_opt,step1} &= \sqrt{G}\cdot S_{no\_opt,col}\\ &= (\sqrt{G}\cdot h\left\{(\sqrt{G}-1)s_{header}+(\sum_{k=1}^{\sqrt{G}-n}k+\sum_{k=1}^{n-1}k)(s_{data}+s_{ID})\right\}\\ &+ \sqrt{G}(\sqrt{G}-1)s_A)\,[\mathrm{B}].\end{aligned}$$

At nth row, every agent receives the packet from the upper grid (the size is $(s_{header}+(n-1)(s_{data}+s_{ID}))$[B]) and lower grid (the size is $(s_{header}+(\sqrt{G}-n)(s_{data}+s_{ID}))$[B]). Thus, each agent stores sensor data whose size is $((\sqrt{G}-1)(s_{data}+s_{ID}))$[B] in its buffer.

STEP2. Crosswise Packet Transmission (At *n*th Row): A mobile agent at the left-end or the right-end grid of nth row first adds its own sensor reading and gridID to its

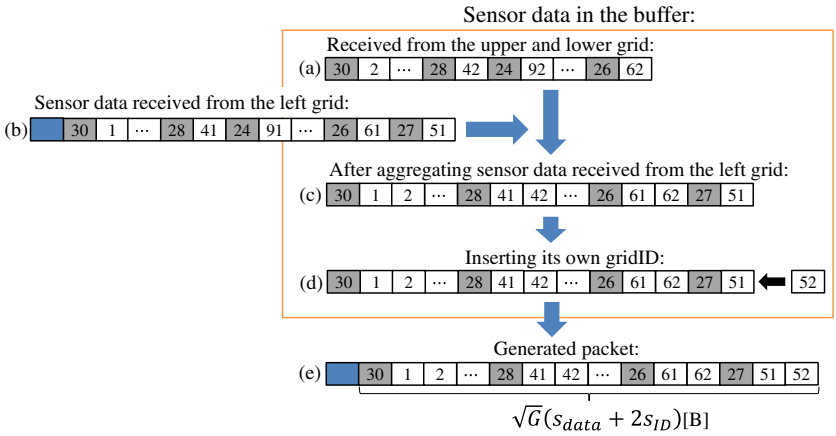


Fig. 19. An example of aggregation by the parent of the agent at the left-end grid in the crosswise distribution (G=100, n=5, m=5)

buffer because its own reading is identical to none of sensor readings in the buffer. Thus, the size of packet transmitted by the agent becomes $(s_{header} + \sqrt{G}(s_{data} + s_{ID}))$[B], and the traffic for delivering the packet to its parent becomes $(h \cdot \{s_{header} + \sqrt{G}(s_{data} + s_{ID})\} + s_A)$[B].

Every other agent on nth row also holds sensor data whose size is $((\sqrt{G} - 1)(s_{data} + s_{ID}))$[B] in the buffer. On receiving a packet from the child at nth row (from the left or right grid), an agent first merges the sensor data included in the packet with those in its buffer. Let us focus on the parent of the agent at the left-end grid (in Fig. 19). First, every sensor reading included in the packet from the left-end grid (i.e., {30, . . ., 28, 24, . . ., 26}, (b) in Fig. 19) except for the last one (i.e., 27) is included in the buffer ((a) in Fig. 19). Thus, according to step (2) in Section 5.3, only the gridIDs in the packet are added to the buffer ((c) in Fig. 19), and the size of sensor data in the buffer becomes $(\sqrt{G} \cdot s_{data} + (2\sqrt{G} - 1)s_{ID})$[B]. Second, according to step (3) in Section 5.3, the mobile agent adds only its own gridID after the gridIDs with the sensor reading which is identical with its own reading ((d) in Fig. 19). As a result, the agent creates a packet whose size is $(s_{header} + \sqrt{G} \cdot s_{data} + 2\sqrt{G} \cdot s_{ID})$[B] ((e) in Fig. 19). Thus, the size of a new created packet increases by $\sqrt{G} \cdot s_{ID}$[B] at the agent at each grid on nth row. Therefore, the size of packet transmitted by the agent at the kth grid from the left-end becomes $(s_{header} + \sqrt{G} \cdot s_{data} + k\sqrt{G} \cdot s_{ID})$[B], and the traffic for delivering this packet to its parent becomes $(h \cdot (s_{header} + \sqrt{G} \cdot s_{data} + k\sqrt{G} \cdot s_{ID}) + s_A)$[B]. Since there are $(m - 1)$ grids from the left-end grid to the mth grid, the total traffic generated at these grids, $S_{no\_opt,step2(left)}$, is expressed by the following equation:

$$S_{no\_opt,step2(left)} = (h \cdot \left\{(m-1)(s_{header} + \sqrt{G} \cdot s_{data}) + \sum_{k=1}^{m-1} k\sqrt{G} \cdot s_{ID}\right\} + (m-1)s_A)\ [\mathrm{B}].$$

In the same way, the total traffic generated at ($\sqrt{G} - m$) grids between the right end and mth grids, $S_{no\_opt,step2(right)}$, is expressed by the following equation:

$$S_{no\_opt,step2(right)} = (h \cdot \left\{ (\sqrt{G} - m)(s_{header} + \sqrt{G} \cdot s_{data}) + \sum_{k=1}^{\sqrt{G}-m} k \sqrt{G} \cdot s_{ID} \right\} + (\sqrt{G} - m) s_A) \, [\mathrm{B}].$$

STEP3. Packet Transmission from the Mobile Agent to the Sink at the Grid Where the Sink Locates: The mobile agent at the grid where the sink locates records different $\sqrt{G}$ readings, G gridIDs. Thus, the size of the packet transmitted by this agent becomes $(s_{header} + \sqrt{G} \cdot s_{data} + G \cdot s_{ID})$[B]. Thus, the traffic generated at this grid, $S_{no\_opt,step3}$, is expressed by the following equation:

$$S_{no\_opt,step3} = (s_{header} + \sqrt{G} \cdot s_{data} + G \cdot s_{ID} + s_A) \, [\mathrm{B}].$$

Consequently, the total traffic, S_{no_opt}, is expressed by the following equation:

$$\begin{aligned} S_{no\_opt} &= S_{no\_opt,step1} + S_{no\_opt,step2(left)} + S_{no\_opt,step2(right)} + S_{no\_opt,step3} \\ &= h \sqrt{G} \left\{ (\sum_{k=1}^{\sqrt{G}-n} k + \sum_{k=1}^{n-1} k)(s_{data} + s_{ID}) + (\sum_{k=1}^{\sqrt{G}-m} k + \sum_{k=1}^{m-1} k) s_{ID} + (\sqrt{G} - 1) s_{data} \right\} \\ &+ h(G-1) s_{header} + s_{header} + \sqrt{G} \cdot s_{data} + G \cdot s_{ID} + G \cdot s_A \, [\mathrm{B}] \end{aligned} \tag{7}$$

6.5 The Relation between the Performance Gain and the Overhead Generated by the Forwarding Route Control

In the situation discussed in this section, the performance gain by the forwarding route control becomes $(S_{no\_opt} - S_{opt})$[B], while the overhead $S_{overhead}$[B] is needed to reconstruct the forwarding tree. Assuming that $s_{header} = 21$[B], $s_{data} = 2$[B], $s_{ID} = 1$[B], $s_A = 5$[B], $D = 1,000$[m], $G = 100$, $l = r = 100$[m], $m = n = 5$, the performance gain and the overhead respectively, become 0.866[B] and 4.212[B]. This indicates that the overhead becomes larger.

However, when the distribution of data values does not change during successive T sensing times after changing from lengthwise to crosswise distributions, the performance gain becomes larger as T increases. We derive the minimum number of sensing times, T, in order for the performance gain to be larger than the overhead.

First, when the forwarding route control is not implemented, the total traffic for T times of data gathering becomes $T \cdot S_{no\_opt}$[B]. On the other hand, in DGUMA/DA, the total traffic for T times of data gathering becomes $T \cdot S_{opt}$[B]. Thus, in order for the performance gain to be larger than the overhead, T must be satisfied the following condition:

$$\begin{aligned} T \cdot S_{no\_opt} - T \cdot S_{opt} &> S_{overhead}. \\ T &> \frac{S_{overhead}}{S_{no\_opt} - S_{opt}}. \end{aligned} \tag{8}$$

Assuming the case described above, T must be larger than 4.863. This indicates that, in the situation discussed in this section, the performance gain becomes larger than the overhead generated by the forwarding route control when the distribution of data values does not change in successive five sensing times.

7 Simulation Experiments

In this section, we show the results of simulation experiments for validating the discussion in Section 6, and evaluating performance of DGUMA/DA. For the simulations, we used the network simulator, Scenargie 1.5[1].

7.1 Simulation Model

There are 2,000 mobile sensor nodes and a sink in a two-dimensional area of 1,000[m]×1,000[m] ($D = 1,000$). The sink is fixed of the point of (400[m], 400[m]) from the left and the bottom edges of the sensing area. Each sensor node moves according to the random waypoint mobility model with a home area [3]. Specifically, a grid is assigned to each node so that the number of nodes assigned to each grid becomes nearly equal. Each node initially located at its assigned grid, and randomly selects its destination in its assigned grid with the probability of 90%, or in the entire sensing area with the probability of 10%. After determining its destination, the node moves there at a constant speed uniformly determined within the range of [0.5, 1][m/sec]. After arriving at the destination, it stops there for 60[sec] before determining the next destination. The sink and sensor nodes communicate with IEEE 802.11p whose transmission rate is 3[Mbps] and communication range r is about 100[m] (the average distance of 1-hop transmission, l, is set to 100[m]). Each sensor node continuously senses the area. The sink divides the area into G ($10^2 \leq G \leq 15^2$) lattice-shaped grids whose size is $1,000/\sqrt{G}$[m] $\times$ $1,000/\sqrt{G}$[m], and sets the center point of each grid as a sensing point. The sink deploys a mobile agent at each sensing point when the simulation starts. The size of the agent data is set as 60[B], assuming that each sensor node has the source code of the mobile agent in advance. The sensing cycle is set as 30[sec]. Moreover, a mobile agent moves to the sensor node located closest to the sensing point when the distance between the sensing point and itself becomes longer than 47[m], which is set as an appropriate value according to our preliminary experiments, or the mobile agent is out of the valid sensing area. As the geographical distribution of data values, we used two different situations, lengthwise distribution as shown in Fig. 12(a) and the crosswise distribution as shown in Fig. 12(b). Table 3 shows the size of each message at the application layer. The size of a sensor reading, s_{data}, is set as 2[B], and that of a gridID, s_{ID}, is 1[B]. In Table 3, i denotes the number of sensor readings and j denotes the number of gridIDs.

[1] Scenargie1.5 Base Simulator revision 8217, Space-Time Engineer, https://www.spacetime-eng.com/

Table 3. Message size

Roll	Message name	Size[B]
Deploying a mobile agent	Deployment message	s_{header} + 60 = 21 + 60 = 81
Sending sensor data	Data message	$s_{header} + s_{data} \cdot i + s_{ID} \cdot j$ = 21 + 2·i + 1·j
Moving a mobile agent	Movement message	s_{header} + 60 = 81
Fixing a route	Route fix message	s_{header} = 21
Releasing a fixed route	Fixed-route release message	s_{header} = 21
ACK	ACK message	s_A =5

Table 4. m and n in a certain G

G	10^2	11^2	12^2	13^2	14^2	15^2
m	5	5	5	6	6	6
n	5	5	5	6	6	6

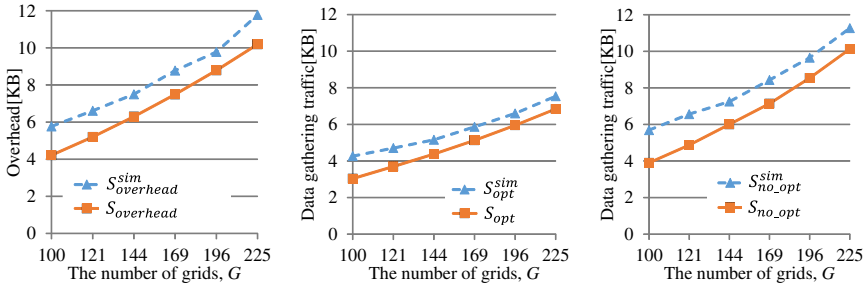
7.2 Validation of the Discussion in Section 6

First, in order to validate the discussion in Section 6, we simulated an environment in Section 6, and measured traffics for the forwarding route control and data gathering in DGUMA/DA. Specifically, we simulated the following three sensing times and measured traffics in each sensing time:

1. At the first sensing time, the data gathering is performed using the lengthwise tree in the lengthwise distribution. In this sensing time, the forwarding routes are fixed.
2. At the second sensing time, the distribution of data values changes from the lengthwise to the crosswise. In this sensing time, fixed routes are released. Let the traffics for data gathering and for releasing fixed routes at this sensing time be $S^{sim}_{no\_opt}$ and $S^{sim}_{release}$, respectively. These values respectively correspond to the theoretical values, S_{no_opt} and $S_{release}$.
3. At the third sensing time, the data gathering is performed using the crosswise tree in the crosswise distribution. In this sensing time, the forwarding routes are fixed. Let the traffics for data gathering and fixing routes at this sensing time be S^{sim}_{opt} and S^{sim}_{fix}, respectively. These values respectively correspond to the theoretical values, S_{opt} and S_{fix}.

Note that the theoretical values ($S_{overhead}$, S_{opt} and S_{no_opt}) are calculated from Eqs.(5), (6) and (7). Here, m and n (i.e., the coordinates of the grid where the sink exists) are respectively set according to the number of grids, G, as shown in Table 4. In the experiment, we have simulated the above situation 100 times, and derives the average of S^{sim}_{opt}, $S^{sim}_{no\_opt}$ and $S^{sim}_{overhead}$ (= $S^{sim}_{fix} + S^{sim}_{release}$).

Fig. 20 shows the experimental results and the theoretical values. The horizontal axis of all graphs is the number of grids, G. The vertical axis respectively indicates $S^{sim}_{overhead}$ and $S_{overhead}$ in Fig. 20(a), S^{sim}_{opt} and S_{opt} in Fig. 20(b), and $S^{sim}_{no\_opt}$ and S_{no_opt} in Fig. 20(c).



(a) $S^{sim}_{overhead}$ and $S_{overhead}$. (b) S^{sim}_{opt} and S_{opt}. (c) $S^{sim}_{no\_opt}$ and S_{no_opt}.

Fig. 20. Comparison of experimental results and theoretical values

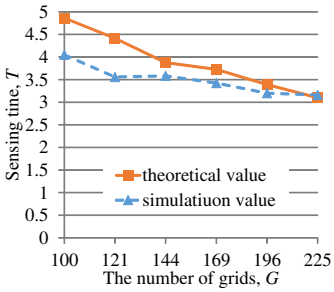


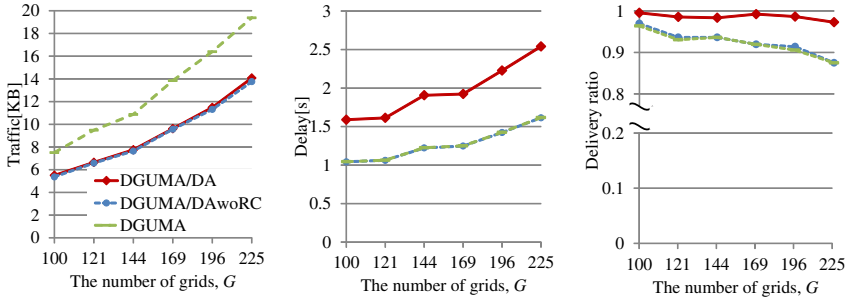
Fig. 21. The minimum number of sensing times in order for the performance gain to be larger than the overhead

From these results, we can see that the theoretical values show similar tendency as the experimental results. Here, the experimental results become larger than the theoretical values especially when G is small. This is mainly because the average of the number of hops in the simulation experiment becomes different from h calculated by Eq.(2).

In addition, we derived the minimum number of sensing times, T, in order for the performance gain to be larger than the overhead using the results in Fig. 20. Fig. 21 shows the result. The horizontal axis of this graph is G. We can see that the difference between the experimental results and the theoretical values become smaller as G increases. This is because the average of the number of hops in the simulation experiment becomes close to h as G increases.

7.3 Performance Evaluation of DGUMA/DA

Second, in order to verify the efficiency of DGUMA/DA, we evaluated the performances of DGUMA/DA and some other methods. For comparison, we evaluated the performances of DGUMA and DGUMA/DA without route control (DGUMA/DAwoRC for short), which only aggregates sensor data according to the procedure in Section 5.3



(a) Traffic. (b) Delay. (c) Delivery ratio.

Fig. 22. Effects of the number of grids (lengthwise distribution)

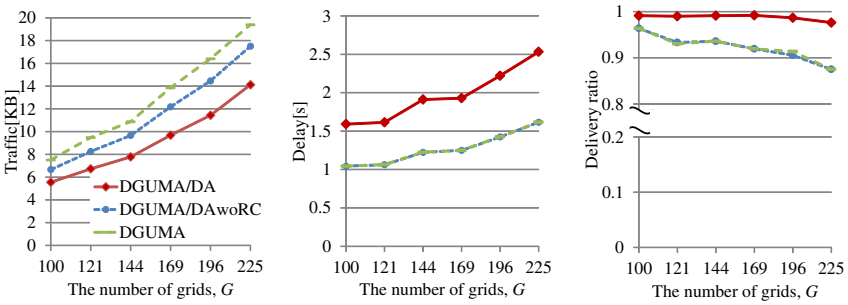
using the fixed (initial) forwarding tree. The simulation time is 3,600[sec] and we evaluated the following three criteria:

- **Traffic:** The traffic is defined as the average of the summation of the size of all packets sent by the sink and all sensor nodes between two consecutive sensing times.
- **Delay:** The delay is defined as the average elapsed time from the start of each sensing time to the time that the sink successfully receives sensor data.
- **Delivery ratio:** The delivery ratio is defined as the ratio of the number of sensor readings which the sink correctly restored to that observed in all grids.

We examined the effects of G, the number of grids. Figs. 22 and 23 show the simulation results. The horizontal axis of all graphs is the number of grids, G.

Lengthwise Distribution: Fig. 22(a) shows the traffic. From this result, we can see that the traffics in all methods increase as G increases. This is because the number of sensor data increases as G increases. We can also see that DGUMA/DA and DGUMA/DAwoRC can gather sensor data with less traffic than DGUMA. This is because DGUMA gathers all sensor data without aggregating them. The traffic in DGUMA/DA is almost same as that in DGUMA/DAwoRC in the lengthwise distribution. This is because the topology of the forwarding tree incidentally becomes suitable for data aggregation even in DGUMA/DAwoRC. Here, the traffic in DGUMA/DA is slightly larger than that in DGUMA/DAwoRC. This is because DGUMA/DA has to send messages to fix routes for data aggregation.

Fig. 22(b) shows the delay. From this result, we can see that the delays in all methods increase as G increases. This is obvious because the number of sensor data increases as G increases. We can also see that the delay in DGUMA/DA becomes longer than those in other methods. This is because mobile agents in DGUMA/DA have to wait until their timers expire before sending a packet, while they can send their packet immediately after receiving packets from all their child agents in other methods.



(a) Traffic. (b) Delay. (c) Delivery ratio.

Fig. 23. Effects of the number of grids (crosswise distribution)

Fig. 22(c) shows the delivery ratio. From this result, we can see that the delivery ratio in DGUMA/DA is higher than those in other methods. In DGUMA and DGUMA/DAwoRC, mobile agents cannot send their packet until receiving packets from all their children. Thus, no packet is sent to the sink once a packet collision occurs. On the other hand, thanks for introducing the timer, mobile agents in DGUMA/DA can send their packet even when packet collisions occur at their descendant. As G increases, the delivery ratio in all methods become lower. This is because a chance of packet collisions becomes larger due to the increase of the number of sensor data. Among the three methods, the delivery ratio in DGUMA/DA keeps high since mobile agents with the different distances from the sink sends their packet at different timings according to their timers. However, DGUMA/DA cannot completely eliminate packet collisions. This is because mobile agents which have almost the same distance to the sink send their packets at almost the same time when *rand* in Eq.(1) becomes very close between these agents.

Crosswise Distribution: Fig. 23(a) shows the traffic. From this result, we can see that the traffic in DGUMA/DA is still small even in the crosswise distribution, while the traffic in DGUMA/DAwoRC becomes much larger. This is because, in the crosswise distribution, less sensor data are aggregated on the forwarding tree with the initial (lengthwise) tree. On the other hand, DGUMA/DA appropriately changes the topology of the forwarding tree according to the geographical distribution of sensor data. Thus, more sensor data can be aggregated. As G increases, the difference in traffic increases between methods. This is because the number of sensor data increases as G increases.

Figs. 23(b) and 23(c), respectively, show the delay and the delivery ratio. These results are almost the same as in Figs. 22(b) and 22(c). This is because the differences in delay and delivery ratio between DGUMA/DA and other methods are caused not only by the difference of forwarding route, but also by the introduction of timer.

8 Conclusion

In this chapter, we have presented DGUMA/DA, which is a data gathering method considering geographical distribution of data values in dense MWSNs. DGUMA/DA

can reduce traffic for gathering sensor data by aggregating the same sensor readings and dynamically constructing the forwarding tree for data aggregation. The results of the simulation experiments show that DGUMA/DA can gather sensor data with high delivery ratio and small traffic.

In this chapter, we assume that DGUMA/DA gathers sensor readings which have only one attribute. However, in a real environment, it is possible that each sensor reading has multiple attributes (e.g., temperature and light intensity). Therefore, it is necessary to extend DGUMA/DA in order to efficiently gather sensor readings with multiple attributes. In addition, DGUMA and DGUMA/DA do not consider erroneous or missing data. Thus, it is necessary to extend DGUMA in order to handle them.

Acknowledgments. This research is partially supported by the Grant-in-Aid for Scientific Research (S)(21220002), (B)(24300037) of MEXT, and for Young Scientists (B)(23700078) of JSPS, Japan.

References

1. Ali, A., Khelil, A., Szczytowski, P., Suri, N.: An adaptive and composite spatio-temporal data compression approach for wireless sensor networks. In: Proc. Int. Conf. on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2011), pp. 67–76 (2011)
2. Burke, J., Estrin, D., Hansen, M., Parker, A., Ramanathan, N., Reddy, S., Srivastava, M.B.: Participatory sensing. Proc. Int. Workshop on World-Sensor-Web (WSW) at Embedded Networked Sensor Systems (Sensys) (2006)
3. Camp, T., Belong, J., Davies, V.: A survey of mobility models for ad hoc network research. Wireless Communications and Mobile Computing 2(5), 483–502 (2002)
4. Campbell, A.T., College, D., Eisenman, S.B., Lane, N.D., Miluzzo, E., Peterson, R.A., Lu, H., Zheng, X., Musolesi, M., Fodor, K., Ahn, G.S.: The rise of peple-centric sensing. IEEE Internet Computing 12(4), 12–21 (2008)
5. Di Francesco, M., Das, S.K., Anastasi, G.: Data collection in wireless sensor networks with mobile elements: a survey. ACM Transactions on Sensor Networks 8(1), 1–34 (2011)
6. Goto, K., Sasaki, Y., Hara, T., Nishio, S.: Data gathering using mobile agents in dense mobile wireless sensor networks. In: Proc. Int. Conf. on Advances in Mobile Computing and Multimedia (MoMM), pp. 58–65 (2011)
7. Heissenbüttel, M., Braun, T., Bernoulli, T., Wälchli, M.: BLR: beacon-less routing algorithm for mobile ad hoc networks. Computer Communications 27(11), 1076–1086 (2004)
8. Landsiedel, O., Götz, S., Wehrle, K.: Towards scalable mobility in distributed hash tables. In: Proc. Int. Conf. on Peer-to-Peer Computing (P2P), pp. 203–209 (2006)
9. Lane, N.D., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T., Campbell, A.T.: A survey of mobile phone sensing. IEEE Communications Magazine 48(9), 140–150 (2010)
10. Luo, C., Wu, F., Sun, J., Chen, C.: Compressive data gathering for large-scale wireless sensor networks. In: Proc. Int. Conf. on Mobile Computing and Networking (MobiCom), pp. 145–156 (2009)
11. Luo, H., Liu, Y., Das, S.K.: Routing correlated data in wireless sensor networks: a survey. IEEE Network 21(6), 40–47 (2007)
12. Matsuo, K., Goto, K., Kanzaki, A., Hara, T., Nishio, S.: Data gathering considering geographical distribution of data values in dense mobile wireless sensor networks. In: Proc. Int. Conf. on Advanced Information Networking and Applications (AINA), pp. 445–452 (2013)

13. Pattem, S., Krishnamachari, B., Govindan, R.: The impact of spatial correlation on routing with compression in wireless sensor networks. ACM Trans. Sensor Networks 4(4), 28–35 (2008)
14. Reddy, S., Samanta, V., Burke, J., Estrin, D., Hansen, M., Strivastava, M.: Examining micropayments for participatory sensing data collections. In: Proc. Int. Conf. on Ubiquitous Computing (UBICOMP), pp. 33–36 (2010)
15. Reddy, S., Samanta, V., Burke, J., Estrin, D., Hansen, M., Strivastava, M.: Mobisense-mobile network services for coordinated participatory sensing. In: Proc. Int. Symposium on Autonomous Decentralized Systems (ISADS), pp. 231–236 (2009)
16. Sharaf, M.A., Beaver, J., Labrinidis, A., Chrysanthis, P.K.: TiNA: a scheme for temporal coherency-aware in-network aggregation. In: Proc. Int. Workshop on Data Engineering for Wireless and Mobile Access (MobiDE), pp. 66–79 (2003)
17. Shi, J., Zhang, R., Liu, Y., Zhang, Y.: Prisense: privacy-preserving data aggregation in people-centric urban sensing systems. In: Proc. Int. Conf. on Computer Communications (INFOCOM), pp. 758–766 (2010)
18. Umer, M., Kulik, L., Tanin, E.: Optimizing query processing using selectivity-awareness in wireless sensor networks. Computers, Environment and Urban Systems 33(2), 79–89 (2009)
19. Weng, H., Chen, Y., Wu, E., Chen, G.: Correlated data gathering with double trees in wireless sensor networks. IEEE Sensors Journal 12(5), 1147–1156 (2012)
20. Yick, J., Mukherjee, B., Ghosal, D.: Wireless sensor network survey. Computer Networks 52(12), 2292–2330 (2008)
21. Zeydan, E., Kivanc, D., Comaniciu, C., Tureli, U.: Energy-efficient routing for correlated data in wireless sensor networks. Ad Hoc Networks 10(6), 962–975 (2012)
22. Zhang, C.: Cluster-based routing algorithms using spatial data correlation for wireless sensor networks. Journal of Communications 5(3), 232–238 (2010)
23. Zhang, J., Wu, Q., Ren, F., He, T., Lin, C.: Effective data aggregation supported by dynamic routing in wireless sensor networks. In: Proc. Int. Conf. on Communications (ICC), pp. 1–6 (2010)
24. Zhu, Y., Vedantham, R., Park, S.J., Sivakumar, R.: A scalable correlation aware aggregation strategy for wireless sensor networks. Information Fusion 9(3), 354–369 (2008)

P2P Data Replication: Techniques and Applications

Evjola Spaho[1], Admir Barolli[2], Fatos Xhafa[3], and Leonard Barolli[1]

[1] Fukuoka Institute of Technology,
3-30-1 Wajiro-Higashi, Higashi-Ku, Fukuoka 811-0295, Japan
evjolaspaho@hotmail.com, barolli@fit.ac.jp
[2] Hosei University,
3-7-2, Kajino-cho, Koganei-shi, Tokyo 184-8584, Japan
admir.barolli@gmail.com
[3] Technical University of Catalonia,
C/Jordi Girona 1-3, 08034 Barcelona, Spain
fatos@lsi.upc.edu

Abstract. Peer-to-Peer (P2P) computing systems offer many advantages of decentralized distributed systems but suffer from availability and reliability. In order to increase availability and reliability, data replication techniques are considered commonplace in P2P computing systems. Replication can be seen as a family of techniques. Full documents or just chunks can be replicated. Since the same data can be found at multiple peers, availability is assured in case of peer failure. Consistency is a challenge in replication systems that allow dynamic updates of replicas. Fundamental to any of them is the degree of replication (full vs. partial), as well as the source of the updates and the way updates are propagated in the system. Due to the various characteristics of distributed systems as well as system's and application's requirements, a variety of data replication techniques have been proposed in the distributed computing field. One important distributed computing paradigm is that of P2P systems, which distinguish for their large scale and unreliable nature. In this chapter we study some data replication techniques and requirements for different P2P applications. We identify several contexts and use cases where data replication can greatly support collaboration. This chapter will also discuss existing optimistic replication solutions and P2P replication strategies and analyze their advantages and disadvantages. We also propose and evaluate the performance of a fuzzy-based system for finding the best replication factor in a P2P network.

Keywords: P2P Systems, Data replication, Replication techniques, Data availability, P2P applications.

1 Introduction

Peer-to-peer (P2P) systems have become highly popular in recent times due to their great potential to scale and the lack of a central point of failure. Thus, P2P

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_6

architectures will be important for future distributed systems and applications. In such systems, the computational burden of the system can be distributed to peer nodes of the system. Therefore, in decentralized systems users themselves become actors by sharing, contributing, and controlling the resources of the system. This characteristic makes P2P systems very interesting for the development of decentralized applications [1, 2].

Important features of such applications include the security, capability to be self-organized, decentralized, scalable, and sustainable [3–6]. P2P computing systems offer many advantages of decentralized distributed systems but suffer from availability and reliability. In order to increase availability and reliability, data replication techniques are considered commonplace in distributed computing systems [7–9]. Initial research work and development in P2P systems considered data replication techniques as means to ensure availability of static information (typically files) in P2P systems under highly dynamic nature of computing nodes in P2P systems. For instance, in P2P systems for music file sharing, the replication allows to find the desired file at several peers as well as to enable a faster download. In many P2P systems the files or documents are considered static or, if the change, new versions are uploaded at different peers. In a broader sense, however, the documents could change over time and thus, the issues of availability, consistency and scalability arise in P2P systems. Consider for instance, a group of peers that collaborate together in a project. They share documents among them, and, in order to increase availability, they decide to replicate the documents. Because documents can be changed by peers, for instance different peers can edit the same document, thus changes should be propagated and made to the replicas to ensure consistency. Moreover, the consistency should be addressed under the dynamics of the P2P systems, which implies that some updates could take place later (as a peer might be off at the time when document changes occurred). In this chapter we discuss different data replication techniques in P2P and different context and uses of data replication.

This chapter is organized as follows. In Section 2, we briefly describe the main characteristics of P2P systems. In Section 3, we explain different P2P data replication techniques. Section 4 describes replication requirements and solutions for different applications. In Section 5, we show our proposed fuzzy-based system for finding the best replication factor in a P2P network and give some simulation results. In Section 6, we give a brief discussion and analysis of replication in P2P. Section 7 concludes this chapter.

2 P2P Systems

A P2P system [10] is a self-organizing system of equal and autonomous entities, which aims for the shared usage of distributed resources in networked environment avoiding central services. A peer is an entity in the system, usually an application running on a device, or the user of such an application. All peers should be of equivalent importance to the system, no single peer should be critical to the functionality of the system.

In a P2P network, peers communicate directly with each other to exchange information. One particular example of this information exchange, that has been rather successful and has attracted considerable attention in the last years, is file sharing. These kind of systems are typically made up of millions of dynamic peers involved in the process of sharing and collaboration without relying in central authorities. P2P systems are characterized by being extremely decentralized and self-organized. These properties are essential in collaborative environments. The popularity and inherent features of these systems have motivated new research lines in the application of distributed P2P computing.

P2P applications such as distributed search applications, file sharing systems, distributed storage system and group ware have been proposed and developed [11], [12], [13], [14], [15].

Some of the essential features of P2P systems are:

- The peers should have autonomy and be able to decide services they wish to offer to other peers.
- Peers should be assumed to have temporary network addresses. They should be recognized and reachable even if their network address has changed.
- A peer can join and leave the system at its own disposal.

P2P systems have the following benefits.

- **Use of the previously unused resources:** On home and office computers, processing cycles are wasted constantly while the computer is on but underutilized/idle (generally overnight and during non-business hours). The disk storage is typically underutilized, as these computers are used mostly for simple nonintensive tasks. The P2P-based application can make use of these resources, thereby increasing utilization of an already paid resource.
- **Potential to scale:** The resources of the server or server-cluster limit the capabilities of the client-server system. As the number of clients increases, it becomes difficult to keep up with demand and maintain the performance and service at the required level. By distributing demand and load on the shared resources, the bottlenecks can be eliminated and a more reliable system achieved.
- **Self-organization:** P2P systems build and organize themselves. Each peer dynamically discovers other peers and builds the network. They organize according to their preferences and current conditions within the peer group. If a popular peer is overloaded and poor performance occurs, consumer peers can switch to another provider, effectively re-balancing load, and changing the network topology.

P2P computing systems offer many advantages of decentralized distributed systems but suffer from availability and reliability. In order to increase availability and reliability, data replication techniques are considered commonplace in P2P computing systems.

3 Data Replication and Update Management in P2P Systems

Initial research work and development in P2P systems considered data replication techniques as a means to ensure availability of static information (typically files) in P2P systems under highly dynamic nature of computing nodes in P2P systems. However, considering only static information and data might be insufficient for certain types of applications in which documents generated along application lifecycle can change over time. The need is then to efficiently replicate dynamic documents and data.

Data replication aims at increasing availability, reliability, and performance of data accesses by storing data redundantly [16–19]. A copy of a replicated data object is called a replica. Replication ensures that all replicas of one data object are automatically updated when one of its replicas is modified. Replication involves conflicting goals with respect to guaranteeing consistency, availability, and performance. Data replication techniques have been extensively used in distributed systems as an important effective mechanism for storage and access to distributed data. Data replication is the process of creating copies of data resources in a network. Data replication is not just copying data at multiple locations as it has to solve several issues. Data replication can improve the performance of a distributed system in several ways:

- **High availability, reliability, and fault tolerance:** Data replication means storing copies of the same data at multiple peers, thus improving availability and scalability. Full documents (or just chunks) can be replicated. Since the same data can be found at multiple peers, availability is assured in case of peer failure. Moreover, the throughput of the system is not affected in case of a scale-out as the operations with the same data are distributed across multiple peers.
- **Scalability:** Service capacity increased due to server load can be decreased. Response time and QoS requirements can be greatly improved.
- **Performance:** Increased performance due to data access.
- **"Fail Safe" infrastructures:** Replication is a choice for today's critical IT systems as replication is key to reducing the time for service recovery.

However, consistency is a challenge in replication systems that allow dynamic updates of replicas.

3.1 Data Replication Update Management

In [30], replica control mechanisms are classified using three criteria (see also [20], [21], [22]): where updates take place (single-master vs. multi-master), when updates are propagated to all replicas (synchronous vs. asynchronous) and how replicas are distributed over the network (full vs. partial replication) as shown in Fig. 1.

3.1.1 Where Updates Take Place: Single-Master and Multi-master

In the Single-master approach, there is only a single primary copy for each replicated object. The single-master allows only one site to have full control over the replica (read and write rights) while the other sites can only have a read right over the replica. This model is also known as the master-slave approach due to the interaction of the master node with the other nodes (slaves) storing the replica. Advantage of this model is the centralization of the updates at a single copy, simplifying the concurrency control. The disadvantage of this model is a single point of failure that can limit the data availability. The single-master approach is depicted in Fig. 2.

The updates can be propagated through *push mode* or *pull mode*. In *push mode*, it is the master that initiates the propagation of the updates, while in the case of *pull mode*, the slave queries the master for existing updates.

In the Multi-master approach, multiple sites hold primary copy of the same object. All these copies can concurrently updated. Multiple sites can modify their saved replicas. This approach is more flexible than single-master because in the case of one master failure, other masters can manage the replicas. The multi-master approach is presented in Fig. 3.

3.1.2 When Updates Are Propagated: Full Replication and Partial Replication

There are two basic approaches for replica placement: full replication and partial replication. Full replication takes place when each participating site stores a copy of every shared object. Every site should have the same memory capacities in order to replace any other site in case of failure. Figure 4 shows how two objects A and B respectively are replicated over three sites.

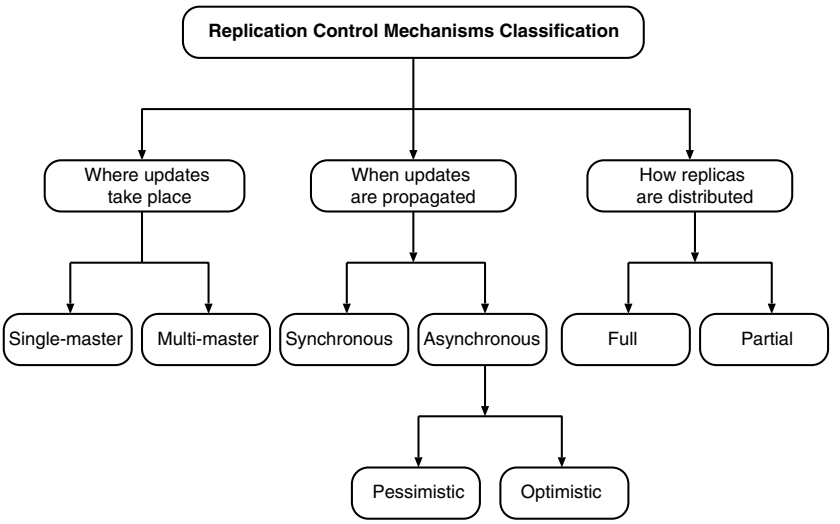


Fig. 1. Replica control mechanisms classification

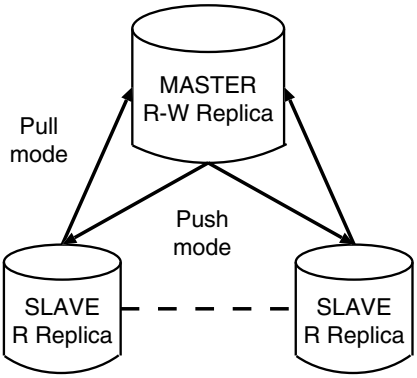


Fig. 2. Single-master replication

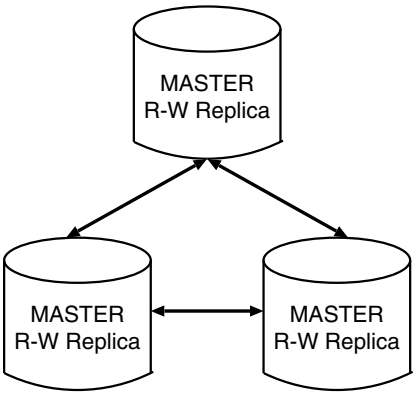


Fig. 3. Multi-master replication

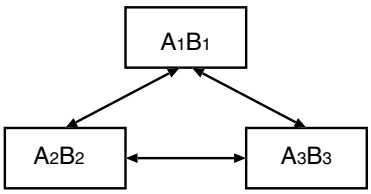


Fig. 4. Full replication with two objects A and B

In partial replication, each site holds a copy of a subset of shared objects so the sites can take different replica objects (see Fig. 5). This approach requires less storage space because updates are propagated only toward the affected sites. But this approach limits load balance possibilities as certain sites are not able to execute a particular type transaction [23]. In partial replication, it is important to

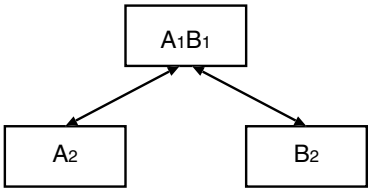


Fig. 5. Partial replication with two objects A and B

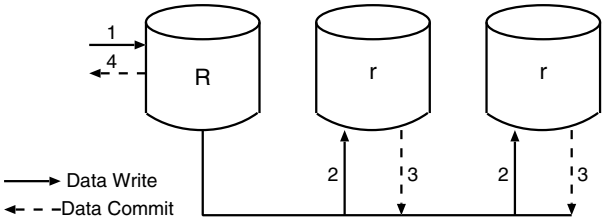


Fig. 6. Synchronous replication

find the right replication factor. Careful planning should be done when deciding which documents to replicate and at which peers.

3.1.3 How Replicas Are Distributed: Synchronous and Asynchronous Replication

Replication can be performed in an eager (synchronous) or lazy way (asynchronous) [30]. In the case of eager replication, when one replica is modified by a transaction, the other replicas of the concerned data object are updated within the original database transaction, as opposed to lazy replication where only the originally accessed replica is updated within the original transaction, while the other replicas are updated in separate transactions. The node that initiates the transaction propagates the update operations within the context of the transaction to all the other replicas before committing the transaction (see Fig. 6).

Combinations of synchronous and asynchronous replication have also been studied [31, 32]. In Synchronous replication, the node that initiate the transaction (set of update operations) propagates the update operations within the context of the transaction to all the other replicas before committing the transaction. There are several algorithms and protocols to achieve this behavior [33, 34]. Synchronous propagation enforces mutual consistency among replicas. In [33] authors define this consistency criteria as one-copy-serializability. The main advantage of synchronous propagation is to avoid divergences among replicas. The drawback is that the transaction has to update all the replicas before committing.

The asynchronous approach does not change all replicas within the context of the transaction that initiates the updates. The transaction is first committed at

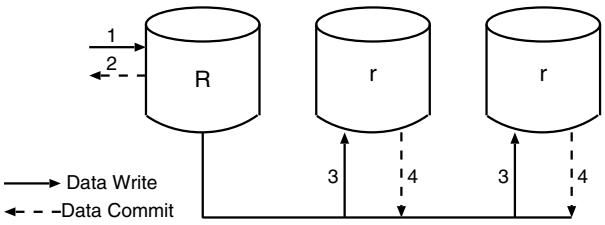


Fig. 7. Asynchronous replication

the local site and after that the updates are propagated to the remote sites as shown in Fig. 7. An advantage of asynchronous propagation is that the update does not block due to unavailable replicas, which improves data availability. The asynchronous replication technique can be classified as optimistic or non-optimistic in terms of conflicting updating [35, 36].

Pessimistic Approaches
These approaches combine lazy replication with one-copy-serializability. Each replicated data item is assigned a primary copy site and multiple secondary copy sites and only the primary copy can be modified. But because primary copies are distributes across the system, serializability cannot be always guaranteed [37]. In order to solve these problems, constraints on primary and secondary copy placements must be set. The problem is solved with the help of graph representation. The Data Placement Graph (DPG) is a graph where each node represents a site and there is a directed edge from Site i to Site j if there is at least one data item for which Site i is the primary site and Site j is the secondary site. The configurations a DPG can have for the system to be serializable is determined with the Global Serialization Graph (GSG). The GSG is obtained by taking the union of nodes and edges of the Local Serialization Graph (LSG) at each site. The LSG is a partial order over the operations of all transactions executed at that site. A DPG is serializable only if GSG is acyclic.

The above method can be enhanced so that it allows some cyclic configurations. In order to achieve this, the network must provide FIFO reliable multicast [35]. The time needed to multicast a message from one node to any other node is not greater than Max and the difference between any two local clocks is not higher than ε. Thus, how a site receives the propagated transaction in at most $Max+\varepsilon$ units of time, chronological, and total orderings can be assured without coordination among sites. The approach reaches the consistency level equivalent to one-copy-serializability for normal workloads and for bursty workloads it is quite close to it. The solution was extended to work in the context of partial replication too [23]. Pessimistic approaches have the disadvantage that two replicas might not be consistent for some time interval. That is why the criterion of consistency freshness is being used, which is defined as the distance between two replicas.

Optimistic Approaches
Optimistic approaches are used for sharing data efficiently in wide-area or mobile environments. The difference between optimistic and pessimistic replication is that the first does not use one-copy-serializability. Pessimistic replication use synchronization during replica propagation and block other users during an update. On the other hand, optimistic replication allows data to be accessed without using synchronization, based on the assumption that conflicts will occur only rarely, if at all. Update propagation is made in the background so that it is possible to exist divergences between replicas. Conflicting updates are reconciled later.

Optimistic approaches have powerful advantages over pessimistic approaches. They improve availability; applications do not block when the local or remote site is down. This type of replication also permits a dynamic configuration of the network, peers can join or leave the network without affecting update propagation. There are techniques that allow such a thing as epidemic replication that propagates operations reliably to all replicas. Unlike pessimistic algorithms, optimistic algorithms scale to a large number of replicas as there is little synchronization among sites. New replicas can be added on the fly without changing the existing sites, examples are FTP and Usenet mirroring. Last but not least, optimistic approaches provide quick feedback as the system applies the updates tentatively as soon as they are submitted [36].

These advantages come at a cost with system consistency. Optimistic replication encounters the challenges of diverging replicas and conflicts between concurrent updates. For these reasons, it is used only with systems in which conflicts are rare and that tolerate inconsistent data. An example are file systems in which conflicts do not happen often due to data partitioning and access arbitration that naturally happen between users.

3.2 Data Replication Techniques in P2P

Data replication techniques in unstructured P2P networks can be classified using two criteria: techniques related with site selection and techniques related with replica distribution (see Fig. 8). Ten different replication techniques that belong to these two groups are discussed in the following.

Owner replication, Path replication and Random replication techniques are evaluated in [24].

Owner replication replicates an object only at the requesting node. When a search is successful, the object is stored at the requester node only. The number of replicas will increase in proportion to the number of requests for the service. This technique uses non-active replication and replicates the item only on the node that requested it. In this technique, the number of replicas generated in the P2P network is limited to one at each data exchange, and so it takes a large amount of time to propagate replicas over the P2P network, thereby limiting the search performance for the requested data. Owner replication is used in systems such as Gnutella.

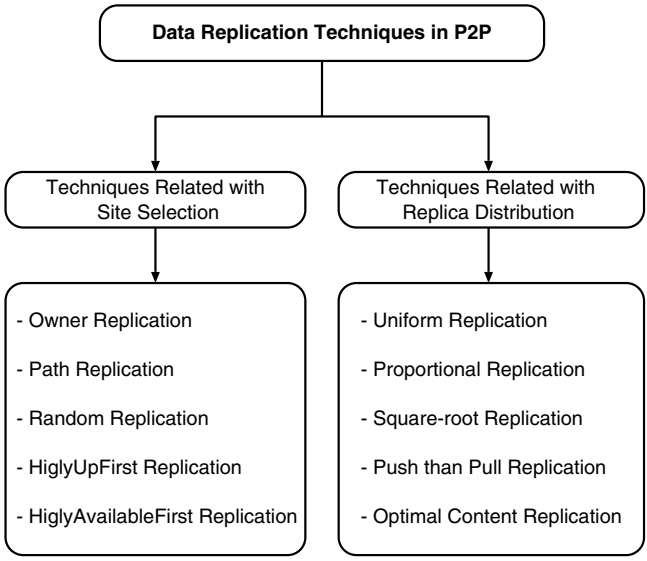


Fig. 8. Data replication techniques in P2P

Path replication is a technique that uses active replication and the requested item is replicated to all nodes of the path between source and destination nodes. In this replication, the peer with a high degree forwards much more data than the peer with a low degree, so that a large number of replications will occur at the peers with a high degree. Therefore, the storage load due to writing and/or reading can be concentrated on a few high-degree peers, which thus play an important role in the P2P system. If the system fails due to overload or some other reason, a large amount of time is needed to recover the system [25]. However, this scheme has been employed in many distributed systems because of its good search performance and ease of implementation. Path replication is used in systems such as Freenet [26].

Random replication distributes the replicas in a random order. In random forwarding n-walkers random walk, random replication is the most effective approach for achieving both smaller search delays and smaller deviations in searches. Random replication is harder to implement, but the performance difference between it and path replication highlights the topological impact of path replication.

In [27], authors use developed two heuristic algorithms (HighlyUpFirst and HighlyAvailableFirst) for solving the replica placement problem and improving quality of availability.

In the **HighlyUpFirst** replication the nodes with the highest uptime are put in the set that will receive the replica.

The **HighlyAvailableFirst** method fills the so-called replica set with the nodes that have a high availability. This technique deals only with availability and does not take into consideration the network overhead.

Uniform replication strategy replicates everything equally. The purpose of this technique is to reduce search traffic. The replicas are distributed uniformly through the network. For each data object, approximately the same number of replicas are created. While this controls the overhead of replication, replicas may be found in places where peers do not access the files.

In **Proportional replication**, the number of replicas is proportional to their popularity. This replication is used for reducing search traffic. If a data item is popular, it has more chances of finding the data close to the site where query was submitted, but it is difficult to find not very popular data items.

In **Square-root replication**, the number of replicas of a file is proportional to the square-root of query distribution. This technique reduces the number of hops needed for finding an object.

Pull-Then-Push replication [28] is based on the following idea: the creation of replicas is delegated to the inquiring node, not the providing node. The scheme consists of two phases. The pull phase refers to searching for a data item. After a successful search, the inquiring node enters a push phase, whereby it transmits the data item to other nodes in the network in order to force creation of replicas. This technique increases network overhead because all neighbors get a copy of replica.

Optimal content replication [29] is an adaptive, fully distributed technique that dynamically replicates content in a near-optimal manner. This replication is used to maximize hit probabilities in P2P communities, taking intermittent connectivity explicitly into account. The optimal object replication includes a logarithmic assignment rule, which provides a closed form optimal solution to the continuous approximation of the problem. This technique does not take into consideration the past performance of nodes for selecting a suitable location for replication and this leads to resource wastage.

4 Replication Requirements and Solutions for Different Applications

Replicating objects to multiple sites has several issues such as selection of objects for replication, the granularity of replicas, and choosing appropriate site for hosting new replica [42].

By storing the data at more than one site, if a data site fails, a system can operate using replicated data, thus, increasing availability and fault tolerance. At the same time, as the data are stored at multiple sites, the request can find the data close to the site where the request originated, thus increasing the performance of the system. But the benefits of replication, of course, do not come without overheads of creating, maintaining, and updating the replicas. If the application has read-only nature, replication can greatly improve the performance. But, if the application needs to process update requests, the benefits of replication can be neutralized to some extent by the overhead of maintaining consistency among multiple replicas. If an application requires rigorous consistency and has large numbers of update transactions, replication may diminish the performance as

a result of synchronization requirements. However, if the application involves read-only queries, performance can be enlarged [38].

4.1 Consistency and Limits to Replication

P2P systems can be used for a wide range of applications, including music and video sharing, wide-area file systems, archival file systems, software distribution. Two key properties of P2P applications that impact the use of replication are the size of the object that should be replicated and the time of replica delivery.

Replication should be transparent to the user, it has to achieve one logical view of the data. The fact that all users see the same data at any time is expressed in terms of consistency. Full consistency means that original data and its replicas are identical, while in partial consistency state there are differences or conflicts among original data and its replicas. One main issue is thus to achieve a satisfactory degree of consistency so that all users see the same data. The degree of consistency depends on many factors, but primarily it depends on whether the application or system can tolerate a partial consistency.

4.2 Context and Uses of Data Replication

Data replication arises in many contexts of distributed systems and applications.

Distributed Storage: One main context of replication is that of Distributed Database Management Systems (DBMS). With the emergence of large-scale distributed computing paradigms such as Cloud, Grid, P2P, Mobile Computing, etc., the data replication has become a commonplace approach, especially to ensure scalability to millions of users of such systems. In particular, data replication is used in Data Centers as part of Cloud Computing systems. In [39], authors propose and evaluate different replication methods for load balancing on distributed storages in P2P networks.

Disaster Management Scenarios: In these scenarios, ensuring anytime access to data is a must. The rescue teams need to collaborate and coordinate their actions and anytime access to data and services is fundamental to support teamwork because decision taking is time-sensitive and often urgent.

Business Applications: Data replication has attracted the attention of researchers and developers from businesses and business intelligence as a key technique to ensure business continuity, continuity-of-operations, real-time access to critical data as well as for purposes of handling big data for business analytics [40], [41]. In such context replication is seen as a choice to make data in a business environment operational.

Collaborative and Groupware Systems: One important requirement in collaborative and groupware systems is to support distributed teamwork, which often suffers from disruption. For example, supporting large user communities (e.g.,

from High Energy Physics community) during scientific collaborations projects. Replication is thus a means to ensure access to data anytime and thus support collaboration even in unreliable networking environments. This later feature is each time more important due to the mobility of the teamwork and use of mobile devices.

In P2P super-peer collaborative systems, peers are organized in peer-groups and collaborate together synchronously and/or asynchronously to accomplish a common project by sharing documents and data (contacts, calendar information, etc.), also known as P2P groupware systems. Such systems are attractive for several application contexts such as collaborative work in online teams in virtual campuses and small to medium corporates. These applications are especially interesting due to their low cost of deployment and maintenance compared to the rather high cost centralized groupware applications; also, they provide facilities to support opportunistic collaboration by giving full control to the users. P2P replication is particularly useful for P2P groupware systems and collaborative teamwork, by increasing the availability and access to all the information that the team manages, supporting thus a whole range of possibilities to accelerate, improve and make more productive teamwork [43].

5 A Fuzzy-Based System for Evaluating Data Replication Factor

5.1 Fuzzy Logic

Fuzzy Logic (FL) is the logic underlying modes of reasoning which are approximate rather then exact. The importance of FL derives from the fact that most modes of human reasoning and especially common sense reasoning are approximate in nature. FL uses linguistic variables to describe the control parameters. By using relatively simple linguistic expressions it is possible to describe and grasp very complex problems. A very important property of the linguistic variables is the capability of describing imprecise parameters.

The concept of a fuzzy set deals with the representation of classes whose boundaries are not determined. It uses a characteristic function, taking values usually in the interval [0, 1]. The fuzzy sets are used for representing linguistic labels. This can be viewed as expressing an uncertainty about the clear-cut meaning of the label. But the important point is that the valuation set is supposed to be common to the various linguistic labels that are involved in the given problem.

The fuzzy set theory uses the membership function to encode a preference among the possible interpretations of the corresponding label. A fuzzy set can be defined by examplification, ranking elements according to their typicality with respect to the concept underlying the fuzzy set [44].

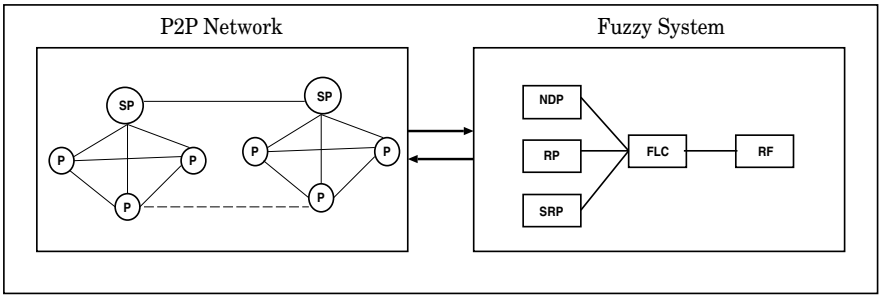


Fig. 9. Fuzzy-based system for evaluating replication factor

5.2 Proposed System

This section presents the architecture of a fuzzy-based system for evaluating replication factor in a P2P network. The structure of our system is shown in Fig. 9.

In our proposed system, we considered P2P systems with super-peer (SP) architecture. A peer-group has multiple peers that can be geographically far away from one another but have a common goal. For example, they could work together accomplishing a certain job. Job consists of several tasks that are to be completed by peers in the group. One of the most important tasks is the replication of the documents among peers of the group. Each peer can directly communicate with any other peer in the group.

In this work we considered peer-groups with the same number of peers. The peer-group has a central manager which is the super-peer. The super-peer assigns tasks to the peers in the group and keeps track of accomplishing the job. The super-peer facilitates the communication with other peers in other peer-groups. The advantage of having a super-peer is that job submissions and data queries arrive faster to the destination. The super-peer makes the connection between the peers in the group and the other peers and super peers from the network. If a part or the document in a peer changes, other peers that have the replica of this document make the changes.

During replication it is important to find a right replication factor. The replication factor is considered the total number of replicated documents over the total number of documents which means the sum of the replicas and original documents in all peers. Careful planning should be done when deciding which documents to replicate and at which peers.

Our fuzzy-based system uses three input parameters which are read from P2P network: Number of Documents per Peer *(NDP)*, Replication Percentage *(RP)*, and Scale of Replication per Peer *(SRP)*. The output parameter is Replication Factor *(RF)*.

The membership functions for our system are shown in Fig. 10. In Table 1, we show the Fuzzy Rule Base (FRB) of our proposed system, which consists of 27 rules.

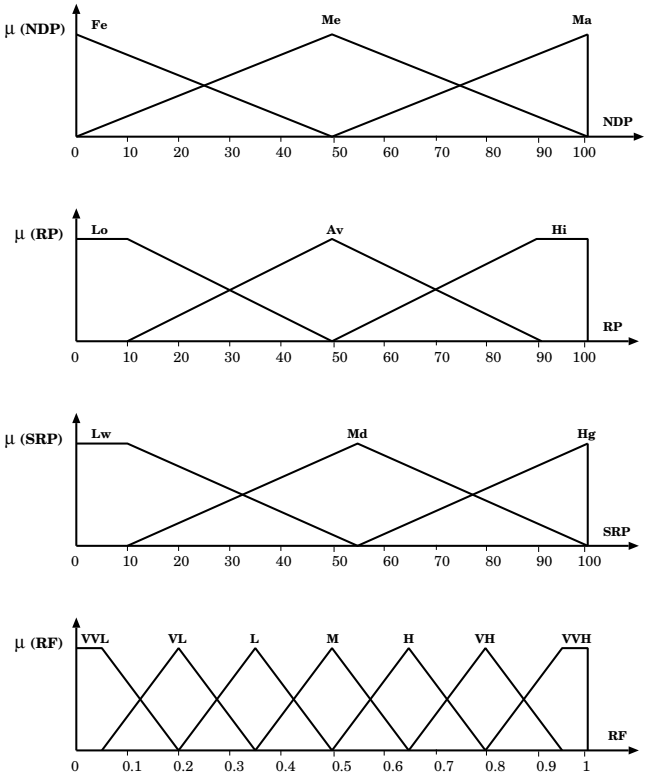


Fig. 10. Membership functions

The term sets of *NDP*, *RP*, and *SRP* are defined respectively as:

$$\begin{aligned}\mu(NDP) &= \{Few,\ Medium,\ Many\}\\ &= \{Fe,\ Me,\ Ma\};\\ \mu(RP) &= \{Low,\ Average,\ High\}\\ &= \{Lo,\ Av.\ Hi\};\\ \mu(SRP) &= \{Low,\ Medium,\ High\}\\ &= \{Lw,\ Md,\ Hg\}.\end{aligned}$$

and the term set for the output (RF) is defined as:

$$\begin{aligned}\mu(RF) = \{&Very\ Very\ Low,\ Very\ Low,\ Low,\ Middle,\ High,\\ &Very\ High,\ Very\ Very\ High\}\\ = \{&VVL,\ VL,\ L,\ M,\ H,\ VH,\ VVH\}.\end{aligned}$$

Table 1. FRB

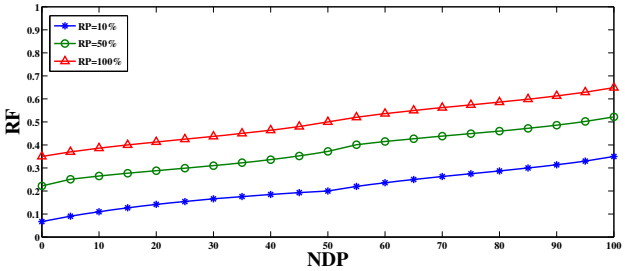
Rules	NDP	RP	SRP	RF
0	Fe	Lo	Lw	VVL
1	Fe	Lo	Md	VL
2	Fe	Lo	Hg	L
3	Fe	Av	Lw	VL
4	Fe	Av	Md	L
5	Fe	Av	Hg	M
6	Fe	Hi	Lw	L
7	Fe	Hi	Md	M
8	Fe	Hi	Hg	H
9	Me	Lo	Lw	VL
10	Me	Lo	Md	L
11	Me	Lo	Hg	M
12	Me	Av	Lw	L
13	Me	Av	Md	M
14	Me	Av	Hg	H
15	Me	Hi	Lw	M
16	Me	Hi	Md	H
17	Me	Hi	Hg	VH
18	Ma	Lo	Lw	L
19	Ma	Lo	Md	M
20	Ma	Lo	Hg	H
21	Ma	Av	Lw	M
22	Ma	Av	Md	H
23	Ma	Av	Hg	VH
24	Ma	Hi	Lw	H
25	Ma	Hi	Md	VH
26	Ma	Hi	Hg	VVH

5.3 Simulation Results

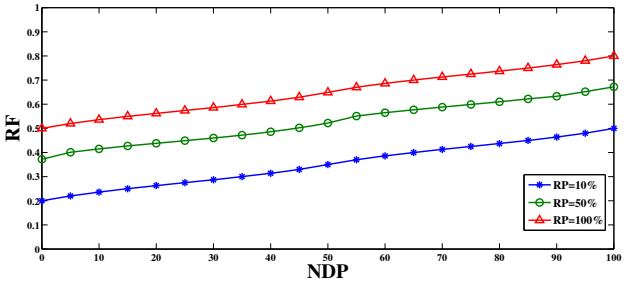
We evaluate the proposed system by simulations. The simulations are carried out using MATLAB. In Fig 11(a), we show the relation between RF and NDP, RP, SRP. In this case the SRP is considered 10%. From the figure we can see that for RP=10% with the increase of the NDP the RF increases. Also, when the RP is increased the replication factor is increased.

In Fig. 11(b) are shown the simulation results for SRP=60%. As can be seen, the replication factor increases with the increase of SRP parameter.

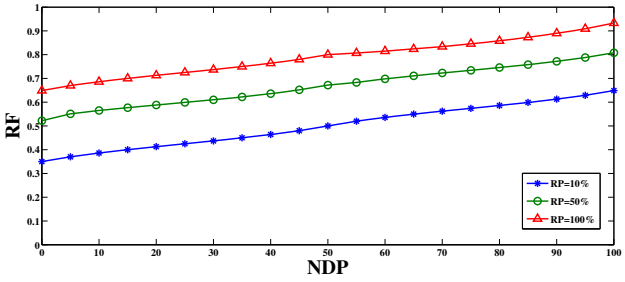
In Fig 11(c), the value of the SRP is considered 1. We can see that, with the increase of NDP and RP, the PR is increased. From the simulation results we conclude that the RF increase proportionally with increases of NDP, RP, and SRP parameters.



(a) Replication factor for SRP=10%.



(b) Replication factor for SRP=60%.



(c) Replication factor for SRP=100%.

Fig. 11. Simulation results

6 Discussion and Analysis

In P2P systems, the quality of the network will decrease if a very popular file is stored at only one available node, because of the high-network traffic rate that is caused by the amount of downloading peers. To solve this problem, replicas of popular files or objects have to be stored elsewhere in the network. If replication is implemented well, the availability, reliability, and scalability of the P2P system will increase.

6.1 Setting Up a Replication Plan

Implementing data replication requires setting up a replication plan and to answer some key questions to ensure desired properties of the system.

1. **What to replicate?** This is to identify the kind of data to replicate.
   - Full objects, fine-grained objects, chunks/blocks can be replicated.
   - Data could be documents, files, meta-data, multimedia, user profiles, events, messages, etc.
   - Data could be static or dynamic over time.
   - Replication could be meant for access purposes only or for disaster recovery as well.
   - Evaluate the homogeneity/heterogeneity degree of the data (heterogeneous vs. homogeneous data).
   - Evaluate the degree of structuring of the data (structured vs. unstructured data).

   Distributed resources should be replicated according to the resource popularities in order to maximize the probability that requests from peers will be satisfied (i.e., hit rate of requests). This is particularly true for popular resources that might have a limited number of replicas in a P2P network when first introduced, and many requests from peers can cause network congestions and slow download speed.
2. **Where to replicate?** This question has to do with the underlying computing environment where replication will take place.
   - Evaluate the heterogeneity degree of the computing environment (heterogeneous vs. homogeneous computing environment).
   - Evaluate how much storage capacity, performance, and reliability, type of storage available at replicated sites.
   - Evaluate the cost of the replication in the underlying infrastructure.

   The replicated copies should be placed in close proximity to peers who are likely to request the resource. This allows peers to be able to search and find desired resources, and reduces delays occurring during search and downloading. Also, P2P architecture is required to adapt replicas into various variations, the replication strategy should use the properties of peers and their surrounding usage environment attributes to determine which peers should be selected to perform adaptive replications and where the resulting replicas should be stored.
3. **How to replicate?** This should address the needs of:
   - How much data has to be online (to be replicated)? Will the replication be done synchronously or asynchronously?
   - How should the original data and its replicas be related? Decide the consistency type, full vs. partial replication, etc.

As in other types of large-scale distributed systems, data replication is useful to achieve important system properties due to system node failures: high availability, system reliability, and scalability. While it is well understood and easy to achieve replication of immutable information (typically files) in P2P systems,

it becomes more challenging to implement data replication in techniques under highly dynamic nature of large P2P systems. Indeed, replicating documents that could change over time requires addressing the consistency issues.

Replicating objects in all sites, which significantly reduce data access cost is not realistic because it generates a large bandwidth consumption. Replicating objects to multiple sites has several issues such as: selection of objects for replication, the granularity of replicas, and choosing an appropriate site for hosting new replica. A replication scheme should manage the frequent failure of nodes in the network to provide good success rate by maintaining replicas in other suitable peers.

Data replication can also be used for maximizing hit probability of access request for the contents in P2P community, maximizing content searching (look-up) time, minimizing the number of hops visited to find the requested content, minimizing the content cost, distributing peer load.

There are a number of advantages and disadvantages to replication.

The following are the advantages of replication:

- **High availability, reliability, and fault tolerance:** Data replication means storing copies of the same data at multiple peers, thus improving availability and scalability. Full documents (or just chunks) can be replicated. Since the same data can be found at multiple peers, availability is assured in case of peer failure. Also, the throughput of the system is not affected in case of a scale-out as the operations with the same data are distributed across multiple peers.
- **Scalability:** Service capacity increased due to server load can be decreased. Response time and QoS requirements can be greatly improved.
- **Performance:** Increased performance due to data access.

The following are the disadvantages of replication:

- **Increased overhead on update:** When an update is required, a database system must ensure that all replicas are updated.
- **Require more disk space:** Storing replicas of same data at different sites consumes more disk space.
- **Expensive:** Concurrency control and recovery techniques will be more advanced and hence, more expensive. In general, replication enhances the performance of read operations and increases the availability of data to read-only transactions. However, update transactions incur greater overhead. Controlling concurrent updates by several translations to replicated data is more complex than using the centralized approach to concurrency control.

7 Conclusions

Data replication and synchronization techniques have recently attracted a lot of attention of researchers from the P2P computing community. Such techniques are fundamental to increase data availability, reliability, and robustness of P2P

applications. However, several issues arise, such as data consistency and designing cost-efficient solutions, due to the highly dynamic nature of P2P systems.

This chapter conducts a theoretical survey of replication techniques in P2P systems. We describe different techniques and discuss their advantages and disadvantages. The replication techniques depends on the application in which they will be used. In general a replication technique should take into consideration at the same time: the reduction of access time and bandwidth consumption, choose an optimal number of replicas and a balanced workload between replicas. Data replication is useful to achieve high-data availability, system reliability, and scalability and can also be used for maximizing hit probability of access request for the contents in P2P community, maximizing content searching (look-up) time, minimizing the number of hops visited to find the requested content, minimizing the content cost, distributing peer load. But the benefits of replication, of course, do not come without overheads of creating, maintaining, and updating the replicas. If the application has read-only nature, replication can greatly improve the performance. But, if the application needs to process update requests, the benefits of replication can be neutralized to some extent by the overhead of maintaining consistency among multiple replicas. If an application requires rigorous consistency and has large numbers of update transactions, replication may diminish the performance as a result of synchronization requirements.

Careful planning should be done when deciding which documents to replicate and at which peers. During replication it is important to find the right replication factor.

References

1. Xhafa, F., Fernandez, R., Daradoumis, T., Barolli, L., Caballé, S.: Improvement of JXTA Protocols for Supporting Reliable Distributed Applications in P2P Systems. In: Enokido, T., Barolli, L., Takizawa, M. (eds.) NBiS 2007. LNCS, vol. 4658, pp. 345–354. Springer, Heidelberg (2007)
2. Barolli, L., Xhafa, F., Durresi, A., De Marco, G.: M3PS: A JXTA-based Multiplatform P2P System and Its Web Application Tools. International Journal of Web Information Systems 2(3/4), 187–196 (2006)
3. Arnedo, J., Matsuo, K., Barolli, L., Xhafa, F.: Secure Communication Setup for a P2P based JXTA-Overlay Platform. IEEE Transactions on Industrial Electronics 58(6), 2086–2096 (2011)
4. Barolli, L., Xhafa, F.: JXTA-Overlay: A P2P Platform for Distributed, Collaborative, and Ubiquitous Computing. IEEE Transactions on Industrial Electronics 58(6), 2163–2172 (2011)
5. Enokido, T., Aikebaier, A., Takizawa, M.: Process Allocation Algorithms for Saving Power Consumption in Peer-to-Peer Systems. IEEE Transactions on Industrial Electronics 58(6), 2097–2105 (2011)
6. Waluyo, A.B., Rahayu, W., Taniar, D., Scrinivasan, B.: A Novel Structure and Access Mechanism for Mobile Data Broadcast in Digital Ecosystems. IEEE Transactions on Industrial Electronics 58(6), 2173–2182 (2011)
7. Zhang, J., Honeyman, P.: A Replicated File System for Grid Computing. Concurrency and Computation: Practice and Experience 20(9), 1113–1130 (2008)

8. Elghirani, A., Subrata, R., Zomaya, A.Y.: Intelligent Scheduling and Replication: a Synergistic Approach. Concurrency and Computation: Practice and Experience 21(3), 357–376 (2009)
9. Nicholson, C., Cameron, D.G., Doyle, A.T., Millar, A.P., Stockinger, K.: Dynamic Data Replication in LCG. Concurrency and Computation: Practice and Experience 20(11), 1259–1271 (2008)
10. Shirkey, C.: What is P2P..and What isn't. O'Reilly Network (November 2000)
11. Gnutella, `http://gnutella.wego.com/`
12. NAPSTER, `http://www.napster.com/`
13. WinMX, `http://www.frontcode.com/`
14. FREENET, `http://frenet.sourceforge.net/`
15. GROOVE, `http://www.groove.net/`
16. Martins, V., Pacitti, E., Valduriez, P.: Survey of Data Replication in P2P Systems. Technical Report (2006)
17. Bernstein, P., Goodman, N.: The Failure and Recovery Problem for Replicated Databases. In: Proc. of the Second Annual ACM Symposium on Principles of Distributed Computing, pp. 114–122. ACM Press, New York (1983)
18. Mustafa, M., Nathrah, B., Suzuri, M., Osman, M.: Improving Data Availability Using Hybrid Replication Technique in Peer-to-Peer Environments. In: Proc. of 18th International Conference on Advanced Information Networking and Applications (AINA-2004), pp. 593–598. IEEE CS Press (2004)
19. Loukopoulos, T., Ahmad, I.: Static and Adaptive Data Replication Algorithms for Fast Information Access in Large Distributed Systems. In: Proc. of 20th International Conference on Distributed Computing Systems (ICDCS 2000), pp. 385–392. IEEE CS Press (2000)
20. Xhafa, F., Potlog, A., Spaho, E., Pop, F., Cristea, V., Barolli, L.: Evaluation of Intragroup Optimistic Data Replication in P2P Groupware Systems. Concurrency Computat.: Pract. Exper (2012), doi:10.1002/cpe.2836
21. Potlog, A.D., Xhafa, F., Pop, F., Cristea, V.: Evaluation of Optimistic Replication Techniques for Dynamic Files in P2P Systems. In: Proc. of Sixth International Conference on on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC 2011), Barcelona, Spain, pp. 259–165 (2011)
22. Xhafa, F., Kolici, V., Potlog, A.D., Spaho, E., Barolli, L., Takizawa, M.: Data Replication in P2P Collaborative Systems. In: Proc. of Seventh International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC 2012), Victoria, Canada, pp. 49–57 (2012)
23. Coulon, C., Pacitti, E., Valduriez, P.: Consistency Management for Partial Replication in a High Performance Database Cluster. In: Proc. of the 11th International Conference on Parallel and Distributed Systems (ICPADS 2005), pp. 809–815 (2005)
24. Lv, Q., Cao, P., Cohen, E., Li, K., Shenker, S.: Search and Replication in Unstructured Peer-to-Peer Networks. In: Proc. of 16th ACM International Conference on Supercomputing (ICS 2002), pp. 84–95 (2002)
25. Keyani, P., Larson, B., Senthil, M.: Peer Pressure: Distributed Recovery from Attacks in Peer-to-Peer Systems. In: Gregori, E., Cherkasova, L., Cugola, G., Panzieri, F., Picco, G.P. (eds.) NETWORKING 2002. LNCS, vol. 2376, pp. 306–320. Springer, Heidelberg (2002)
26. Clarke, I., Sandberg, O., Wiley, B., Hong, T.W.: Freenet: A Distributed Anonymous Information Storage and Retrieval System. In: Federrath, H. (ed.) Anonymity 2000. LNCS, vol. 2009, pp. 46–66. Springer, Heidelberg (2001)

27. On, G., Schmitt, J., Steinmetz, R.: The Effectiveness of Realistic Replication Strategies on Quality of Availability for Peer-to-Peer Systems. In: Proc. of the Third International IEEE Conference pn Peer-to-Peer Computing, pp. 57–64 (2003)
28. Leontiadis, E., Dimakopoulos, V.V., Pitoura, E.: Creating and Maintaining Replicas in Unstructured Peer-to-Peer Systems. In: Nagel, W.E., Walter, W.V., Lehner, W. (eds.) Euro-Par 2006. LNCS, vol. 4128, pp. 1015–1025. Springer, Heidelberg (2006)
29. Kangasharju, J., Ross, K.W., Turner, D.A.: Optimal Content Replication in P2P Communities. Manuscript, pp. 1–26 (2002)
30. Gray, J., Helland, P., O'Neil, P., Shasha, D.: The Dangers of Replication and a Solution. In: Proc. of International Conference on Management of Data (SIGMOD 1996), pp. 173–182 (1996)
31. Lubinski, A., Heuer, A.: Configured Replication for Mobile Applications. In: Databases and Information Systems, pp. 101–112. Kluwer Academic Publishers, Dordrecht (2000)
32. Rohm, U., Bohm, K., Schek, H., Schuldt, H.: FAS - A Freshness-Sensitive Coordination Middleware for a Cluster of OLAP Components. In: Proc. of 28th International Conference on Very Large Data Bases (VLDB 2002), pp. 754–765 (2002)
33. Bernstein, P.A., Hadzilacos, V., Goodman, N.: Concurrency Control and Recovery in Database Systems (1987)
34. Kemme, B., Alonso, G.: A New Approach to Developing and Implementing Eager Database Replication Protocols. ACM Transactions on Database Systems 25(3), 333–379 (2000)
35. Pacitti, E., Minet, P., Simon, E.: Fast Algorithms for Maintaining Replica Consistency in Lazy Master Replicated Databases. In: Proc. of the 25th International Conference on Very Large Data Bases (VLDB 1999), pp. 126–137 (1999)
36. Saito, Y., Shapiro, M.: Optimistic Replication. ACM Comput. Surv. 37(1), 42–81 (2005)
37. Chundi, P., Rosenkranz, D.: Deferred Updates and Data Placement in Distributed Databases (1996)
38. Goel, S., Buyya, R.: Data Replication Strategies in Wide Area Distributed Systems. In: Enterprise Service Computing: From Concept to Deployment, pp. 211–241. IGI Global (2007)
39. Yamamoto, H., Maruta, D., Oie, Y.: Replication Methods for Load Balancing on Distributed Storages in P2P Networks. The Institute of Electronics, Information and Communication Engineers E-89-D(1), 171–180 (2006)
40. Sheppard, E.: Continuous Replication for Business-Critical Applications. White Paper, pp. 1–7 (2012)
41. Van Der Lans, R.F.: Data Replication for Enabling Operational BI., White Paper on Business Value and Architecture, pp. 1–26 (2012)
42. Ulusoy, O.: Research Issues in Peer-to-Peer Data Management. In: Proc. of International Symposium on Computer and Information Sciences (ISCIS 2007), pp. 1–8 (2007)
43. Estepa, A.N., Xhafa, F., Caballé, S.: A P2P Replication-Aware Approach for Content Distribution in e-Learning Systems. In: Proc. of Sixth International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS 2012), pp. 917–922 (2012)
44. Terano, T., Asai, K., Sugeno, M.: Fuzzy Systems Theory And Its Applications. Academic Press, Inc., Harcourt Brace Jovanovich Publishers (1992)

Leveraging High-Performance Computing Infrastructures to Web Data Analytic Applications by Means of Message-Passing Interface

Alexey Cheptsov and Bastian Koller

High Performance Computing Center Stuttgart,
Nobelstr. 19, 70569 Stuttgart, Germany
{cheptsov,koller}@hlrs.de

Abstract. Modern computing technologies are increasingly getting data-centric, addressing a variety of challenges in storing, accessing, processing, and streaming massive amounts of structured and unstructured data effectively. An important analytical task in a number of scientific and technological domains is to retrieve information from all these data, aiming to get a deeper insight into the content represented by the data in order to obtain some useful, often not explicitly stated knowledge and facts, related to a particular domain of interest. The major issue is the size, structural complexity, and frequency of the analyzed data' updates (i.e., the 'big data' aspect), which makes the use of traditional analysis techniques, tools, and infrastructures ineffective. We introduce an innovative approach to parallelise data-centric applications based on the Message-Passing Interface. In contrast to other known parallelisation technologies, our approach enables a very high-utilization rate and thus low costs of using productional high-performance computing and Cloud computing infrastructures. The advantages of the technique are demonstrated on a challenging Semantic Web application that is performing web-scale reasoning.

Keywords: Data-as-a-Service, Performance, Parallelisation, MPI, OMPIJava.

1 Introduction

The novel trends of the linked and open data [1] have enabled a principally new dimension of data analysis, which is no longer limited to internal data collections, i.e., "local data", but spans over a number of heterogeneous data sources, in particular from the Web, i.e., "global data". Today's data processing application have to increasingly access interlinked data from many different sources, e.g., known social networks as Twitter and Facebook, or web-scale knowledge bases as Linked Life Data [2] or Open PHACTS [3]. Along the Semantic Web – one of the most challenging data-centric application domains – offers the volume of the integrated data that has already reached the order of magnitude of billions of triples ("subject-predicate-object" relation entities in a Semantic Graph) [4] and is expected to further grow in the future (see the actual Linking Open Data cloud diagram [5]). However, existing data processing and analysis technologies are still far from being able to scale

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_7

to demands of global and, in case of large industrial corporations, even of local data, which makes up the core of the "big data" problem. With regard to this, the design of the current data analysis algorithms requires to be reconsidered in order to enable the scalability on a big data demand.

The problem has two major aspects:

1. The solid design of current algorithms makes the integration with other techniques that would help increase the analysis quality impossible.
2. Sequential design of the algorithms prevents porting them to parallel computing infrastructures and thus do not fulfill high performance and other QoS user requirements.

With regard to the first issue – the low performance of the design patterns used in conventional data analytic and web applications – the SOA approach [6], e.g., as implemented in the LarKC platform [7], enables the execution of the most computation-intensive parts of application workflows (such as one shown in Figure 1) on a high-performance computing system, whereas less performance-critical parts of the application can be running in "usual" places, e.g., a web server or a database.

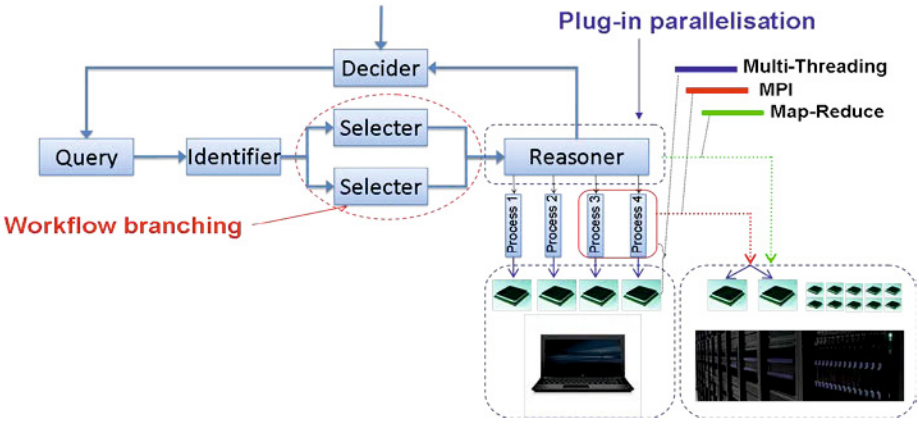


Fig. 1. Workflow-based design of a Semantic Web reasoning application (LarKC) and main parallelisation patterns

The second identified issue – lack of parallelisation models for running the applications on High-Performance Computing infrastructures – is however more essential and poses the major obstacle to endorsing large-scale parallelism to the data-centric development. The traditional serial computing architectures increasingly prove ineffective when scaling Web processing algorithms to analyze the big data. On the other hand, large-scale High-Performance Computing (HPC) infrastructures, both in academic domain and industry, have very special requirement to the applications running on them, which are not confirmed with the most of Web applications. While the existing data-centric parallelisation frameworks, such as MapReduce/Hadoop [19], have proved very successful for running on relatively small-scale **clusters of workstations** and **private Clouds** (in the literature there are no evidence of Hadoop being evaluated on parallel computing architectures with more than 100 nodes), the

use of **HPC systems** (offering several hundred thousands of nodes and computation performance on the exascale range) remains out of scope of the current frameworks' functionality. The reason for this is twofold. First, the existing frameworks (here we basically refer to Hadoop [13], which dominates on the current software market) are implemented as a set of services developed in Java programming language, which has traditionally found quite a limited support on HPC systems due to security, performance, and other restrictions. On the other hand, the well established, mature, optimized, and thus very efficient parallelisation technologies that have been developed for HPC, such as the Message-Passing Interface (MPI) [15], do not support Java – the programming language that is used for developing most of the current data-centric applications and databases, in particular for the Semantic Web.

Whereas the modern parallelisation frameworks cannot take a full advantage of the deployment in HPC environments, due to their design features, the traditional parallelisation technologies fail to meet the requirements of data-centric application development in terms of the offered programming language support as well as service abilities. The known approaches address this problem in two directions [12]. The first is adapting data-centric (MapReduce-based) frameworks for running in HPC environments. One of the promising researches, done by the Sandia laboratory, is to implement the MapReduce functionality from scratch based on a MPI library [8]. Another work that is worth mentioning is done by the Open MPI consortium to port some basic Hadoop interfaces to a MPI environment, which is called MR+ [9]. The common problem of all these approaches is that they are restricted to a MapReduce-like programming model, which decreases their value when used in a wide range of applications.

The alternative approach is to develop Java bindings for a standard MPI library, which is already optimized on a supercomputing system [10]. The benefit of using MPI in parallel Java applications is twofold. First, the MPI programming model is very flexible (support of several types of domain decomposition etc.), and allows the developer to go beyond the quite restricted MapReduce's key/value-based model in designing the parallel execution pattern. Second, MPI is the most efficient solution for reaching application's high performance, which is in particular thanks to a number of optimizations on the network interconnect level. Moreover, the HPC community has already done some efforts to standardize a Java interface for MPI [32], so that Java is increasingly getting recognized as one of the mainstream programming languages for HPC applications as well.

Both MapReduce- and MPI-based approaches to develop parallel Java applications are promising for data-centric application development, each having certain pro and contra arguments over each other. In this chapter, we concentrate on an approach based on MPI. In particular, we discuss OMPIJava – a trend-new implementation of Java bindings for Open MPI – one of the currently most popular Message-Passing Interface implementations for the C, C++, and Fortran supercomputing applications. The rest of the chapter is organized as follows. Section 2 discusses data-centric application parallelisation techniques and compares MapReduce- and MPI-based approaches. Section 3 introduces the basics of the OMPIJava tool and discusses its main features and technical implementation details. Section 4 shows performance evaluation results of OMPIJava based on standard benchmark sets. Section 5 presents an example of a challenging Semantic Web application – Random Indexing – implemented with OMPIJava and shows performance benchmarks for it. Section 6 discusses the use of performance analysis tools for parallel MPI applications. Section 7 concludes the chapter and discusses directions of future work.

2 Data-Centric Parallelisation with MPI

By "data-centric parallelisation" we mean a set of techniques for: (i) identification of non-overlapping application's dataflow regions and corresponding to them instructions; (ii) partitioning the data into subsets; and (iii) parallel processing of those subsets on the resources of the high-performance computing system. In case of Semantic Web applications, the parallelisation relies mainly on partitioning (decomposing) the RDF data set (used as a major data standard) [20] on the level of statements (triples).

MapReduce is the mainstream framework for developing parallel data-centric applications [19]. MapReduce and its most prominent implementation in Java, Hadoop [13], have got a tremendous popularity in modern data-intensive application scenarios. In MapReduce, the application's workflow is divided into three main stages (see Figure 2): map, process, and reduce.

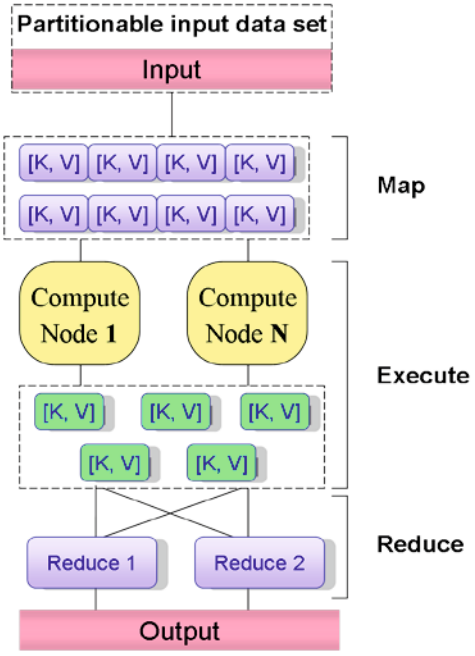


Fig. 2. Schema of data processing with MapReduce

In the map stage, the input data set is split into independent chunks and each of the chunks is assigned to independent tasks, which are then processed in a completely parallel manner (process stage). In the reduce stage, the output produced by every map task is collected, combined, and the consolidated final output is then produced. The Hadoop framework is a service-based implementation of MapReduce for Java. Hadoop considers a parallel system as a set of master and slave nodes, deploying on them services for scheduling tasks as jobs (Job Tracker), monitoring the jobs (Task Tracker), managing the input and output data (Data Node), re-executing the failed

tasks, etc. This is done in a way that ensures very high service reliability and fault tolerance properties of the parallel execution. In Hadoop, both the input and the output of the job are stored in a special distributed file system. In order to improve the reliability, the file system also provides an automatic replication procedure, which however introduces an additional overhead to the internode communication.

The Message-Passing Interface (MPI) is a process-based standard for parallel applications implementation. MPI processes are independent execution units that contain their own state information, use their own address spaces, and only interact with each other via inter-process communication mechanisms defined by MPI. Each MPI process can be executed on a dedicated compute node of the high-performance architecture, i.e., without competing with the other processes in accessing the hardware, such as CPU and RAM, thus improving the application performance and achieving the algorithm speed-up. In case of the shared file system, such as Lustre [21], which is the most utilized file system standard of the modern HPC infrastructures, the MPI processes can effectively access the same file section in parallel without any considerable disk I/O bandwidth degradation. With regard to the data decomposition strategy presented in Figure 3a, each MPI process is responsible for processing the data partition assigned to it proportionally to the total number of the MPI processes (see Figure 3b). The position of any MPI process within the group of processes involved in the execution is identified by an integer R (rank) between 0 and N-1, where N is a total number of the launched MPI processes. The rank R is a unique integer identifier assigned incrementally and sequentially by the MPI run-time environment to every process. Both the MPI process's rank and the total number of the MPI processes can be acquired from within the application by using MPI standard functions, such as presented in Listing 1.

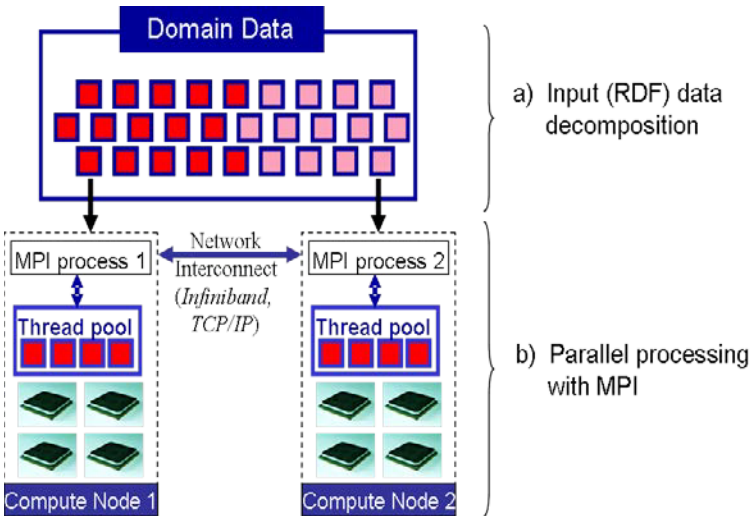


Fig. 3. Data decomposition and parallel execution with MPI

```
import java.io.*;
import mpi.*;

class Hello {
  public static void main(String[] args) throws
       MPIException
  {
    int my_pe, npes; // rank and overall number of MPI
         processes
    int N;       // size of the RDF data set (number of
         triples)

    MPI.Init(args); // intialization of the MPI RTE

    my_pe = MPI.COMM_WORLD.Rank();
    npes  = MPI.COMM_WORLD.Size();

    System.out.println("Hello␣from␣MPI␣process" + my_pe +
         "␣out␣of␣" + npes);
    System.out.println("I'm␣processing␣the␣RDF␣triples␣
         from␣" + my_pe/npes + "␣to␣" + (my_pe+1)/npes);

    MPI.Finalize(); // finalization of the MPI RTE
  }
}
```

Listing 1: Simple example of Java application using MPI bindings

A typical data processing workflow with MPI can be depicted as shown in Figure 4. The MPI jobs are executed by means of the *mpirun* command, which is an important part of any MPI implementation. *Mpirun* controls several aspect of parallel program execution, in particular launches MPI processes under the job scheduling manager software like OpenPBS [22]. The number of MPI processes to be started is provided with the "*-np*" parameter to *mpirun*. Normally, the number of MPI processes corresponds to the number of the compute nodes, reserved for the execution of parallel job. Once the MPI process is started, it can request its rank as well as the total number of the MPI processes associated with the same job. Based on the rank and total processes number, each MPI process can calculate the corresponding subset of the input data and process it. The data partitioning problem remains beyond the scope of this work; particularly for RDF, there is a number of well-established approaches, e.g., horizontal [23], vertical [24], and workload driven [25] decomposition.

Since a single MPI process owns its own memory space and thus cannot access the data of the other processes directly, the MPI standard foresees special communication functions, which are necessary, e.g., for exchanging the data subdomain's boundary values or consolidating the final output from the partial results produced by each of the processes. The MPI processes communicate with each other by sending messages, which can be done either in "point-to-point" (between two processes) or collective way (involving a group of or all processes). More details about the MPI communication can also be found in a previous publication about OMPIJava [27].

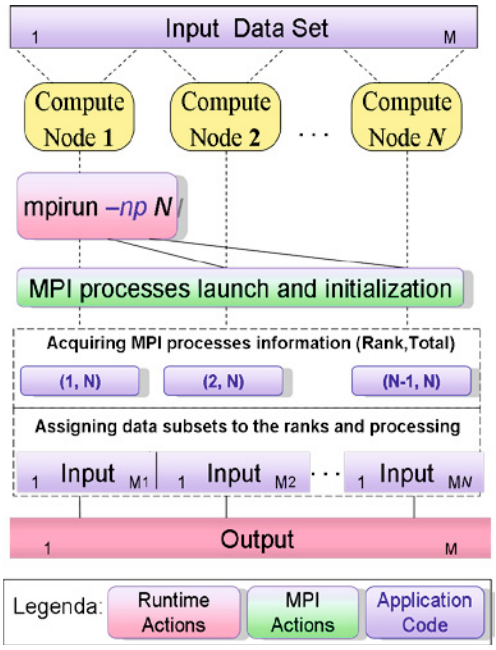


Fig. 4. Typical MPI data-centric application's execution workflow

3 OMPIJava – Java Bindings for Open MPI

Although the official MPI standard only recognizes interfaces for C, C++, and Fortran languages, there has been a number of standardization efforts made toward creating MPI bindings for Java. The most complete API set, however, has been proposed by mpiJava [28] developers. There are only a few approaches to implement MPI bindings for Java. These approaches can be classified in two following categories:

- Pure Java implementations, e.g., based on RMI (Remote Method Invocation) [29], which allows Java objects residing in different virtual machines to communicate with each other, or lower-level Java sockets API.
- Wrapped implementations using the native methods implemented in C languages, which are presumably more efficient in terms of performance than the code managed by the Java run-time environment.

In practice, none of the above-mentioned approaches satisfies the contradictory requirements of the Web users on application portability and efficiency. Whereas the pure Java implementations, such as MPJ Express [30] or MPJ/Ibis [14][18], do not benefit from the high speed interconnects, e.g., InfiniBand, and thus introduce communication bottlenecks and do not demonstrate acceptable performance on the majority of today's production HPC systems [31], a wrapped implementation, such as mpiJava [32], requires a native C library, which can cause additional integration and interoperability issues with the underlying MPI implementation.

In looking for a tradeoff between the performance and the usability, and in view of the complexity of providing Java support for high speed cluster interconnects, the most promising solution seems to be to implement the Java bindings directly in a native MPI implementation in C.

Despite a great variety of the native MPI implementations, there are only a few of them that address the requirements of Java parallel applications on process control, resource management, latency awareness and management, and fault tolerance. Among the known sustainable open-source implementations, we identified Open MPI [33] and MPICH2 [34] as the most suitable to our goals to implement the Java MPI bindings. Both Open MPI and MPICH2 are open-source, production quality, and widely portable implementations of the MPI standard (up to its latest 2.0 version). Although both libraries claim to provide a modular and easy-to-extend framework, the software stack of Open MPI seems to better suit the goal of introducing a new language's bindings, which our research aims to. The architecture of Open MPI [16] is highly flexible and defines a dedicated layer used to introduce bindings, which are currently provided for C, F77, F90, and some other languages (see also Figure 6). Extending the OMPI-Layer of Open MPI with the Java language support seems to be a very promising approach to the discussed integration of Java bindings, taking benefits of all the layers composing Open MPI's architecture.

We have based our Java MPI bindings on the mpiJava code, originally developed in HPJava [35] project and currently maintained on SourceForge [26]. mpiJava provides a set of Java Native Interface (JNI) wrappers to the native MPI v.1.1 communication methods, as shown in Figure 7. JNI enables the programs running inside a Java run-time environment to invoke native C code and thus use platform-specific features and libraries [36], e.g., the InfiniBand software stack. The application-level API is constituted by a set of Java classes, designed in conformance to the MPI v.1.1 and the specification in [28]. The Java methods internally invoke the MPI-C functions using the JNI stubs. The realization details for mpiJava can be obtained from [17][37].

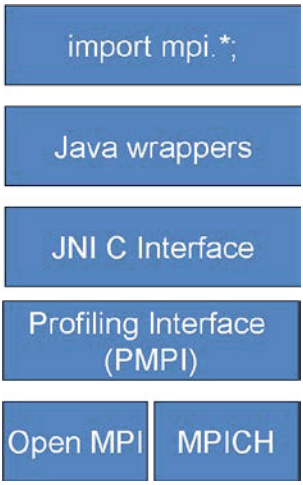


Fig. 5. OMPIJava architecture

Open MPI is a high performance, production quality, and the MPI-2 standard compliant implementation. Open MPI consists of three combined abstraction layers that provide a full featured MPI implementation: (i) OPAL (Open Portable Access Layer) that abstracts from the peculiarities of a specific system away to provide a consistent interface adding portability; (ii) ORTE (Open Run-Time Environment) that provides a uniform parallel run-time interface regardless of system capabilities; and (iii) OMPI (Open MPI) that provides the application with the expected MPI standard interface. Figure 6 shows the enhanced Open MPI architecture, enabled with the Java bindings support.

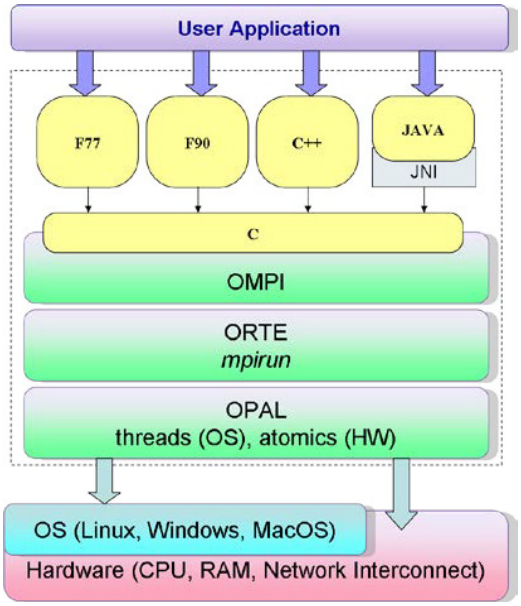


Fig. 6. Open MPI architecture

The major integration tasks that we performed were as follows:

- extend the Open MPI architecture to support Java bindings,
- extend the previously available mpiJava bindings to MPI-2 (and possibly upcoming MPI-3) standard,
- improve the native Open MPI configuration, build, and execution system to seamlessly support the Java bindings,
- redesign the Java interfaces that use JNI in order to better conform to the native realization,
- optimize the JNI interface to minimize the invocation overhead,
- create test applications for performance benchmarking.

Both Java classes and JNI code for calling the native methods were integrated into Open MPI. However, the biggest integration effort was required at the OMPI (Java classes, JNI code) and the ORTE (run-time specific options) levels. The implementation

of the Java class collection followed the same strategy as for the C++ class collection, for which the opaque C objects are encapsulated into suitable class hierarchies and most of the library functions are defined as class member methods. Along with the classes implementing the MPI functionality (MPI package), the collection includes the classes for error handling (Errhandler, MPIException), datatypes (Datatype), communicators (Comm), etc. More information about the implementation of both Java classes and JNI-C stubs can be found in previous publications [17][31].

4 OMPIJava Performance Evaluation

In order to evaluate the performance of our implementation, we prepared a set of Java benchmarks based on those well-recognized in the MPI community, such as NetPIPE [38] or NAS [39]. Based on those benchmarks, we compared the performance of our implementation based on Open MPI and the other popular implementation (MPJ Express) that follows a "native Java" approach. Moreover, in order to evaluate the JNI overhead, we reproduced the benchmarks also in C and ran them with the native Open MPI. Therefore, the following three configurations were evaluated:

- **ompiC** - native C implementation of Open MPI (the actual trunk version), built with the GNU compiler (v.4.6.1),
- **ompiJava** - our implementation of Java bindings on top of ompiC, running with Java JDK (v.1.6.0), and
- **mpj** - the newest version of MPJ Express (v.0.38), a Java native implementation, running with the same JDK.

We examined two types of communication: point-to-point (between two nodes) and collective (between a group of nodes), varying the size of the transmitted messages. We did intentionally not rely on the previously reported benchmarks [40] in order to eliminate the measurement deviations that might be caused by running tests in a different hardware or software environment. Moreover, in order to ensure a fair comparison between all these three implementations, we ran each test on the absolutely same set of compute nodes. The point-to-point benchmark implements a "ping-pong" based communication between two single nodes; each node exchanges the messages of growing sizes with the other node by means of blocking Send and Receive operations. As expected, *ompiJava* implementation was not as efficient as the underlying *ompiC*, due to the JNI function calls overhead, but showed much better performance than the native Java-based mpj (Figure 7). Regardless of the message size, *ompiJava* achieves around eight times higher throughput than *mpj* (see Figure 8).

The collective communication benchmark implements a single blocking message gather from all the involved nodes. Figure 9 shows the results collected for $P = 2_k$ (where $k=2\text{-}7$) nodes, with a varying size of the gathered messages. The maximal size of the aggregated data was 8 GByte on 128 nodes. Figure 10 demonstrates the comparison of collective gather performance for all tested implementations on the maximal number of the available compute nodes (128). Whereas the InfiniBand-aware *ompiJava* and *ompiC* scaled quite well, the native Java-based *mpj* has shown very poor performance; for the worst case (on 128 nodes) a slow-down up to 30 times compared with *ompiJava* was observed.

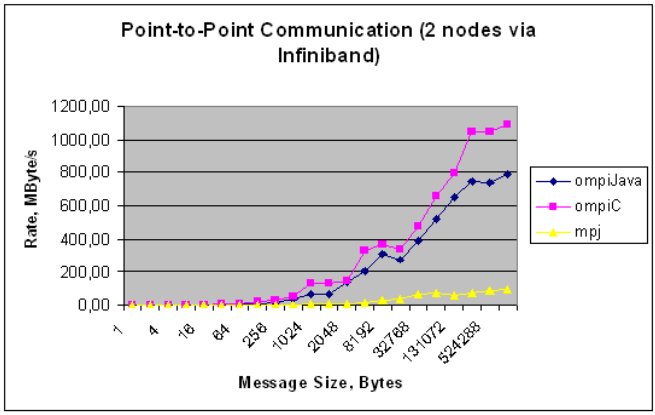
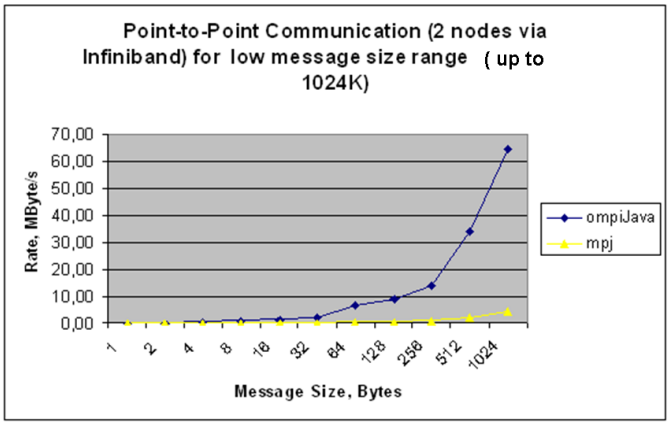
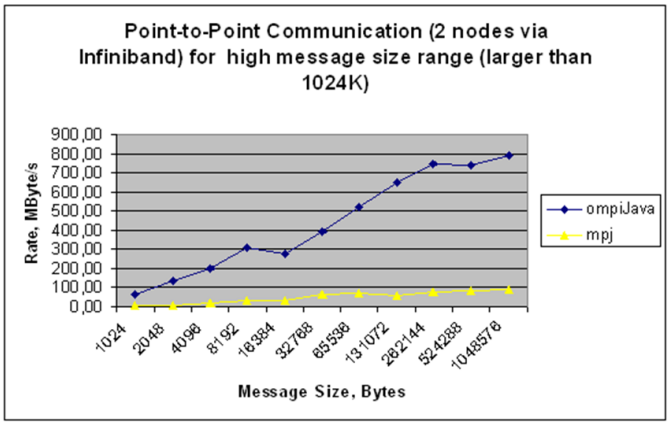


Fig. 7. Message rate for the point-to-point communication



a)



b)

Fig. 8. Comparison of the message rate for *ompiJava* and *mpj* for a) low and b) high message size range

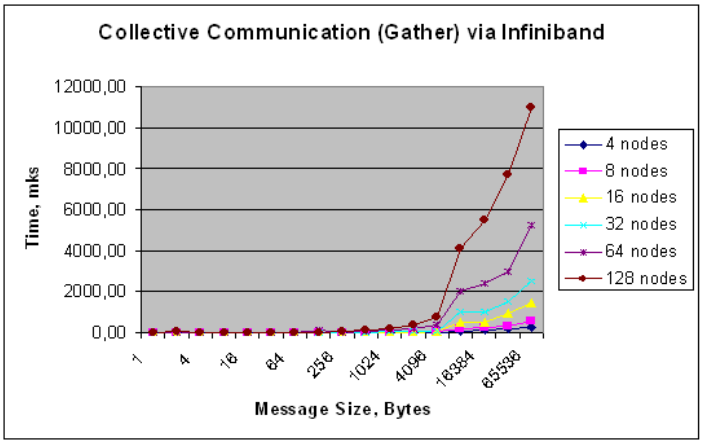


Fig. 9. Collective gather communication performance of *ompiJava*

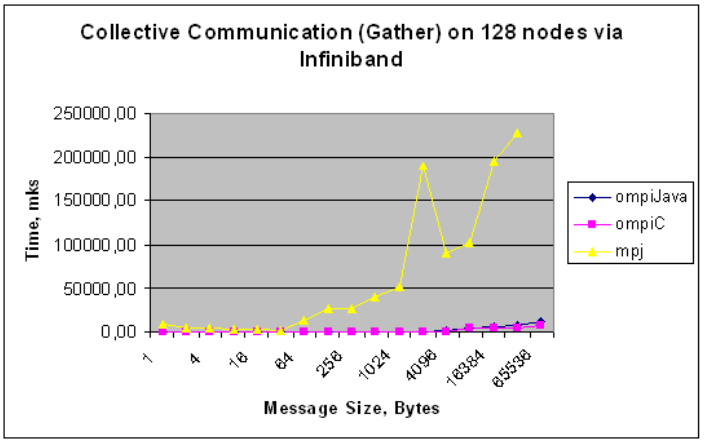


Fig. 10. Collective gather communication performance on 128 nodes

5 Parallelisation of Random Indexing with MPI

Random indexing [41] is a word-based co-occurrence statistics technique used in resource discovery to improve the performance of text categorization. Random indexing offers new opportunities for a number of large-scale Web applications performing the search and reasoning on the Web scale [42]. We used Random Indexing to determine the similarity index (based on the words' co-occurrence statistic) between the terms in a closed document collection, such as Wikipedia or Linked Life Data (see Figure 11).

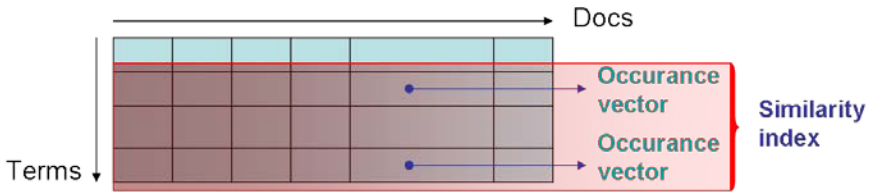


Fig. 11. Similarity index computation in a document collection

The main challenges of the Random Indexing algorithm can be defined in the following:

- Very large and high-dimensional vector space. A typical random indexing search algorithm performs traversal over all the entries of the vector space. This means, that the size of the vector space to the large extent determines the search performance. The modern data stores, such as Linked Life Data or Open PHACTS consolidate many billions of statements and result in vector spaces of a very large dimensionality. Performing Random indexing over such large data sets is computationally very costly, with regard to both execution time and memory consumption. The latter poses a hard constraint to the use of random indexing packages on the serial mass computers. So far, only relatively small parts of the Semantic Web data have been indexed and analyzed.
- High call frequency. Both indexing and search over the vector space is highly dynamic, i.e., the entire indexing.

The MPI implementation of Airhead search [43] is based on a domain decomposition of the analyzed vector space and involves both point-to-point and collective gather and broadcast MPI communication (see the schema in Figure 12).

In order to compare the performance of OMPIJava with MPJ-Express, we performed the evaluation for the largest of the available data sets reported in [43] (namely, Wiki2), which comprises 1 Million of high density documents and occupies 16 GByte disk storage space. The overall execution time (wall clock) was measured. Figure 13a shows that both *ompijava* and *mpj* scale well until the problem size is large enough to saturate the capacities of a single node. Nevertheless, *ompijava* was around 10% more efficient over alternative tools (Figure 13b).

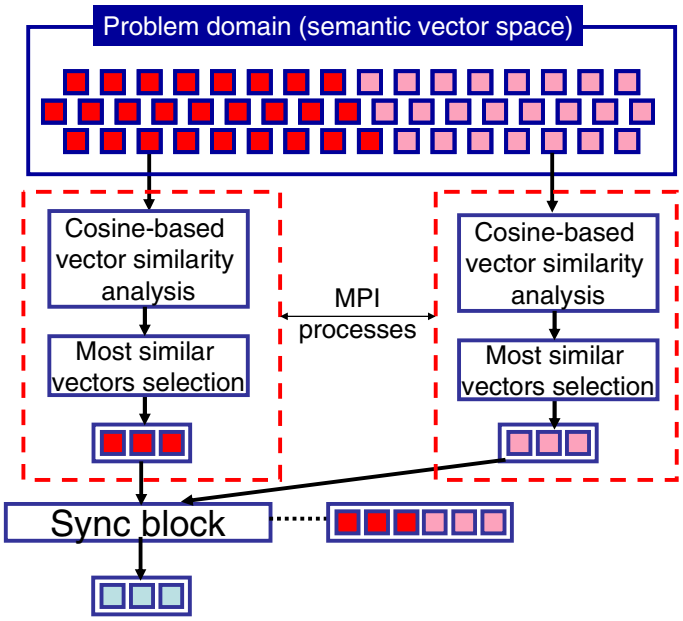
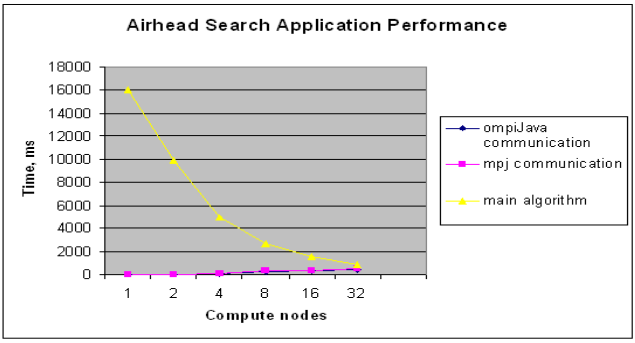
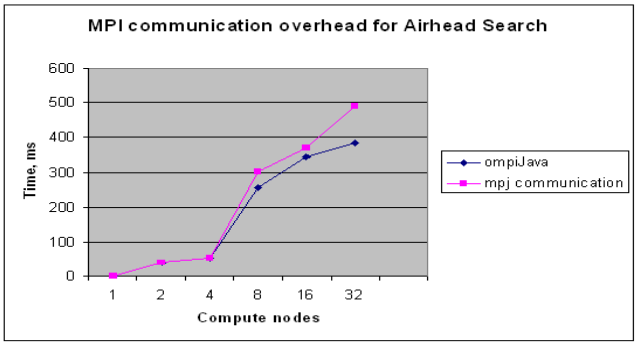


Fig. 12. MPI-based parallel implementation of Airhead Search



a)



b)

Fig. 13. Airhead performance with *ompiJava* and *mpj*

6 Performance Analysis Tools

Development of parallel communication patterns with MPI is quite a nontrivial task, in particular for large-scale use cases, which consist of hundreds and even thousands of parallel processes. The synchronization among the MPI processes of the parallel application can be a key performance concern. Among the typical problems the following appear most frequently:

- nonoptimal balancing of the MPI processes load (i.e., wrong data decomposition),
- wrong configuration of the communication pattern preventing the applications scalability to the growing number of compute nodes,
- incorrect usage of the MPI communication functions (e.g., when point-to-point communication are used instead of the collective ones, which lowers the performance and the scalability).

One of the advantages of the C-based Java binding implementation as compared with the "native-Java" approach is the possibility to use numerous performance optimization tools available for the traditional HPC applications. This is leveraged by the special profiling interface provided by the MPI standard - PMPI (see Figure 5). Using PMPI, performance analysis tools can inject the measurement code directly in the parallel application's object file and capture and aggregate statistics about the application execution at run-time. Among the parameters measured with PMPI are duration of a single MPI communication, total number of communications, processes that are involved in the communication, etc. The profiling code is dynamically linked with the MPI library and thus, does not require any changes in either the application code or the MPI library. The captured events are stored in trace files using a special format, such as OTF - the Open Trace Format, which can then be analyzed in order to retrieve and visualize the application's communication profile.

In our pilot investigations, we evaluated the ability of the Extrae [44] profiling library, developed by the Barcelona Supercomputing Center, to collect event traces of the MPI-parallelised Airhead Search application. For this purpose, we linked Extrae with our Java-enabled version of Open MPI and run the instrumented version of Airhead on the cluster. The traces collected as result of the execution were visualized with the Paraver [45] tool (see Figure 14), similar to any other MPI application in C or Fortran.

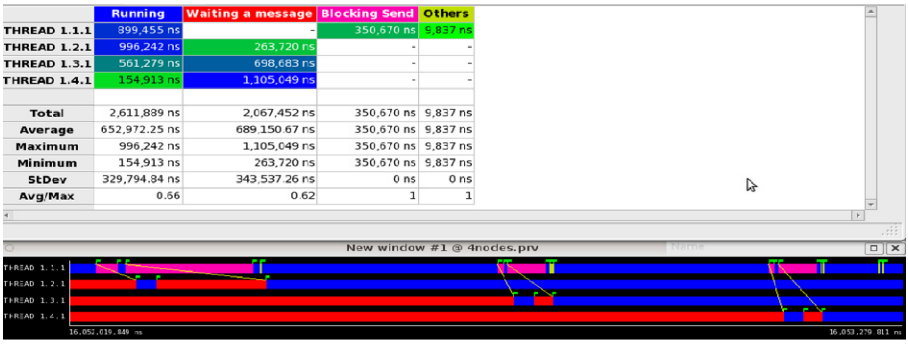


Fig. 14. MPI Global Broadcast Communication visualization for four MPI processes with the Paraver tool

7 Conclusion and Future Directions

High-Performance Computing is a relatively new trend for the Web development, which, however, has gained a tremendous popularity because of emerging big data applications. The Message Passing Interface provides a very promising approach for developing parallel data-centric applications. Unlike its prominent alternatives, the MPI functionality is delivered on the library-level, and thus, does not require any considerable development efforts to parallelise an existing serial application. Apart from a very flexible parallelisation strategy, which not only allows for a number of diverse parallelisation options, either on the code, data, or both levels, but also delivers a very efficient communication mechanism that takes the full advantage of the modern supercomputing communication networks. Using MPI, Semantic Web applications can enjoy the full backing of the high-performance computing architectures. We would like to point out, that the current work is neither no case an attempt to undermine the value of data-centric parallel implementations (like Hadoop), nor it is a replacement for any current data processing infrastructures. However, many of the current parallel data processing systems can benefit from adopting MPI and the tool introduced in this chapter – OMPIJava.

The chapter discussed a new implementation of Java bindings for MPI that is integrated in one of the most popular open source MPI-2 libraries - Open MPI. The integration allowed us to deliver a unique software environment for flexible development and execution of parallel MPI applications, integrating the Open MPI framework's capabilities, such as portability and usability, with those of mpiJava, such as an extensive set of Java-based API for MPI communication. We evaluated our implementation for Random Indexing, which is one of the most challenging Semantic Web applications in terms of the computation demands currently. The evaluation has confirmed our initial considerations about the high efficiency of MPI for parallelising Java applications. In the following, we are going to investigate further capabilities of MPI for improving the performance of data-centric applications, in particular by means of MPI-IO (MPI extension to support efficient file input-output). We will also concentrate on promoting the MPI-based parallelisation strategy to the other challenging and performance-demanding applications, such as Reasoning. We believe that our implementation of Java bindings of MPI will attract Semantic Web development community to increase the scale of both its serial and parallel applications. The successful pilot application implementations done based on MPI, such as materialization of the finite RDFS closure presented in [47], offer a very promising outlook regarding the future perspectives of MPI in the Semantic Web domain.

The future work for the presented activity will concentrate on promoting both MPI standard and the new (OMPIJava) implementation to Semantic Web applications as well as improving the current realization of Java bindings in Open MPI. With regard to promotion activities, we will be introducing our data-centric and MPI-based parallelisation approach to further challenging data-intensive applications, such as Reasoning [46]. Regarding this application, there are highly successful MPI implementations in C, e.g., the parallel RDFS graph closure materialization presented

in [47], which are indicatively much more preferable over all the existing Java solutions in terms of performance. Our implementation will allow the developed MPI communication patterns to be integrated in existing Java-based codes, such as Jena [11] or Pellet [48], and thus drastically improve the competitiveness of the Semantic Web application based on such tools. The development activities will mainly focus on extending the Java bindings to the full support of the MPI-3 specification. We will also aim at adding Java language-specific bindings into the MPI standard, as a reflection of the Semantic Web value in supercomputing. The integration activities will concentrate on adapting the performance analysis tools to the specific of Java applications. Unfortunately, the existing performance analysis tools, such as Extrae discussed in the previous section, does not provide a deep insight in the intrinsic characteristics of the Java Virtual Machine, which however might be important for the application performance optimization as the communication profile tailoring. For this purpose, the traditional performance analysis tools for the Java applications, such as ones provided by the Eclipse framework, must be extended with the communication profiling capabilities. Several EU projects, such as JUNIPER [49], are already working in this direction.

Acknowledgment. Authors would like to thank the Open MPI consortium for the support with porting mpiJava bindings, to the EU-ICT JUNIPER project for the support with the Java platform and parallelisation, as well as the developers of the Airhead library, in particular David Jurgens, for the provided use case.

References

1. Gonzalez, R.: Closing in on a million open government data sets (2012), http://semanticweb.com/closinginona-millionopengovernmentdatasets_b29994
2. Linked Life Data repository website, http://linkedlifedata.com/
3. OpenPHACTS project website, http://www.openphacts.org/
4. Coffman, T., Greenblatt, S., Marcus, S.: Graph-based technologies for intelligence analysis. Communications of ACM 47, 45–47 (2004)
5. Linked Open Data initiative, http://lod-cloud.net
6. Cheptsov, A., Koller, B.: A service-oriented approach to facilitate big data analytics on the Web. In: Topping, B.H.V., Iványi, P. (eds.) Proceedings of the Fourteenth International Conference on Civil, Structural and Environmental Engineering Computing. Civil-Comp Press, Stirlingshire (2013)
7. Cheptsov, A.: Semantic Web Reasoning on the internet scale with Large Knowledge Collider. International Journal of Computer Science and Applications, Technomathematics Research Foundation 8(2), 102–117 (2011)
8. Plimpton, S.J., Devine, K.D.: MapReduce in MPI for large-scale graph algorithms. Parallel Computing 37, 610–632 (2011)
9. Castain, R.H., Tan, W.: MR+. A technical overview (2012), http://www.open-mpi.de/video/mrplus/Greenplum_RalphCastain-2up.pdf

10. Cheptsov, A.: Enabling High Performance Computing for Semantic Web applications by means of Open MPI Java bindings. In: Proc. the Sixth International Conference on Advances in Semantic Processing (SEMAPRO 2012) Conference, Barcelona, Spain (2012)
11. McCarthy, P.: Introduction to Jena. IBM Developer Works (2013), http://www.ibm.com/developerworks/xml/library/j-jena
12. Gonzalez, R.: Two kinds of big data (2011), http://semanticweb.com/two-kinds-ofbig-datb21925
13. Hadoop framework website, http://hadoop.apache.org/mapreduce
14. Bornemann, M., van Nieuwpoort, R., Kielmann, T.: Mpj/ibis: A flexible and efficient message passing platform for Java. Concurrency and Computation: Practice and Experience 17, 217–224 (2005)
15. MPI: A Message-Passing Interface standard. Message Passing Interface Forum (2005), http://www.mcs.anl.gov/research/projects/mpi/mpistandard/mpi-report-1.1/mpi-report.htm
16. Gabriel, E., et al.: Open MPI: Goals, concept, and design of a next generation MPI implementation. In: Kranzlmüller, D., Kacsuk, P., Dongarra, J. (eds.) EuroPVM/MPI 2004. LNCS, vol. 3241, pp. 97–104. Springer, Heidelberg (2004)
17. Baker, M., et al.: MPI-Java: An object-oriented Java interface to MPI. In: Rolim, J.D.P. (ed.) IPPS-WS 1999 and SPDP-WS 1999. LNCS, vol. 1586, pp. 748–762. Springer, Heidelberg (1999)
18. van Nieuwpoort, R., et al.: Ibis: a flexible and efficient Java based grid programming environment. Concurrency and Computation: Practice and Experience 17, 1079–1107 (2005)
19. Dean, J., Ghemawat, S.: MapReduce - simplified data processing on large clusters. In: Proc. OSDI 2004: 6th Symposium on Operating Systems Design and Implementation (2004)
20. Resource Description Framework (RDF). RDF Working Group (2004), http://www.w3.org/RDF/
21. Lustre file system - high-performance storage architecture and scalable cluster file system. White Paper. Sun Microsystems, Inc. (December 2007)
22. Portable Batch System (PBS) documentation, http://www.pbsworks.com/
23. Dimovski, A., Velinov, G., Sahpaski, D.: Horizontal partitioning by predicate abstraction and its application to data warehouse design. In: Catania, B., Ivanović, M., Thalheim, B. (eds.) ADBIS 2010. LNCS, vol. 6295, pp. 164–175. Springer, Heidelberg (2010)
24. Abadi, D.J., Marcus, A., Madden, S.R., Hollenbach, K.: Scalable Semantic Web data management using vertical partitioning. In: Proc. The 33rd International Conference on Very Large Data Bases (VLDB 2007) (2007)
25. Curino, C., et al.: Workload-aware database monitoring and consolidation. In: Proc. SIGMOD Conference, pp. 313–324 (2011)
26. OMPIJava tool website, http://sourceforge.net/projects/mpijava/
27. Cheptsov, A., et al.: Enabling high performance computing for Java applications using the Message-Passing Interface. In: Proc. of the Second International Conference on Parallel, Distributed, Grid and Cloud Computing for Engineering (PARENG 2011) (2011)
28. Carpenter, B., et al.: mpiJava 1.2: API specification. Northeast Parallel Architecture Center. Paper 66 (1999), http://surface.syr.edu/npac/66
29. Kielmann, T., et al.: Enabling Java for High-Performance Computing: Exploiting distributed shared memory and remote method invocation. Communications of the ACM (2001)

30. Baker, M., Carpenter, B., Shafi, A.: MPJ Express: Towards thread safe Java HPC. In: Proc. IEEE International Conference on Cluster Computing (Cluster 2006) (2006)
31. Judd, G., et al.: Design issues for efficient implementation of MPI in Java. In: Proc. of the 1999 ACM Java Grande Conference, pp. 58–65 (1999)
32. Carpenter, B., et al.: MPJ: MPI-like message passing for Java. Concurrency and Computation - Practice and Experience 12(11), 1019–1038 (2000)
33. Open MPI project website, http://www.openmpi.org
34. MPICH2 project website, http://www.mcs.anl.gov/research/projects/mpich2/
35. HP-JAVA project website, http://www.hpjava.org
36. Liang, S.: Java Native Interface: Programmer's Guide and Reference. Addison-Wesley (1999)
37. Vodel, M., Sauppe, M., Hardt, W.: Parallel high performance applications with mpi2java - a capable Java interface for MPI 2.0 libraries. In: Proc. of the 16th Asia-Pacific Conference on Communications (APCC), Nagoya, Japan, pp. 509–513 (2010)
38. NetPIPE parallel benchmark website, http://www.scl.ameslab.gov/netpipe/
39. Bailey, D., et al.: The NAS Parallel Benchmarks. RNR Technical Report RNR-94.007 (March 1994), http://www.nas.nasa.gov/assets/pdf/techreports/1994/rnr-94-007.pdf
40. MPJ-Express tool benchmarking results, http://mpj-express.org/performance.html
41. Sahlgren, M.: An introduction to random indexing. In: Proc. Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering (TKE 2005), pp. 1–9 (2005)
42. Jurgens, D.: The S-Space package: An open source package for word space models. In: Proc. of the ACL 2010 System Demonstrations, pp. 30–35 (2010)
43. Assel, M., et al.: MPI realization of high performance search for querying large RDF graphs using statistical semantics. In: Proc. The 1st Workshop on High-Performance Computing for the Semantic Web, Heraklion, Greece (May 2011)
44. Extrae performance trace generation library website, http://www.bsc.es/computer-sciences/extrae
45. Paraver performance analysis tool website, http://www.bsc.es/computer-sciences/performance-tools/paraver/general-overview
46. Fensel, D., van Harmelen, F.: Unifying reasoning and search to web scale. IEEE Internet Computing 11(2), 95–96 (2007)
47. Weaver, J., Hendler, J.A.: Parallel materialization of the finite RDFS closure for hundreds of millions of triples. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 682–697. Springer, Heidelberg (2009)
48. Sirin, E., et al.: Pellet: a practical owl-dl reasoner. Journal of Web Semantics (2013), http://www.mindswap.org/papers/PelletJWS.pdf
49. Cheptsov, A., Koller, B.: JUNIPER takes aim at Big Data. inSiDE - Journal of Innovatives Supercomputing in Deutschland 11(1), 68–69 (2011)

ReHRS: A Hybrid Redundant System for Improving MapReduce Reliability and Availability

Jia-Chun Lin[1], Fang-Yie Leu[2], and Ying-ping Chen[1]

[1] Department of Computer Science, National Chiao Tung University, Taiwan
kellylin1219@gmail.com, ypchen@cs.nctu.edu.tw
[2] Department of Computer Science, TungHai University, Taiwan
leufy@thu.edu.tw

Abstract. MapReduce is a parallel programming framework proposed by Google. Recently, it has become a popular technology for solving data-intensive applications. However, current MapReduce implementations provide insufficient redundant mechanisms for their master servers, consequently causing the fact that the master servers' services cannot continue and all jobs cannot proceed and complete when the master servers unexpectedly fail. To solve this problem, this chapter proposes a master server redundant mechanism called the Reliable Hybrid Redundant System (ReHRS for short), in which a hot-standby server is employed to maintain the latest metadata of the master sever so as to achieve a fast takeover, and a warm-standby server is employed to further enhance system reliability and extend the operation of MapReduce when both the master server and hot-standby server cannot work properly. We proposed a failure detection algorithm to detect the failure of the master server and hot-standby server, and provided appropriate takeover processes to continue their operations. Additionally, we introduced a dynamic warmup mechanism for the warm-standby server to warm itself up such that it can quickly act as the hot-standby server when necessary. The extensive simulation and experiment results show that the ReHRS significantly speeds up the takeover process as compared with three state-of-the-art schemes.

Keywords: MapReduce, single-point-of-failure, reliability, availability, reliable hybrid redundant system.

1 Introduction

MapReduce [1] is a distributed programming model introduced by Google to process a vast amount of data in parallel on large-scale machines/clusters. With this model, Google processes more than 20 petabytes of data per day [1]. Due to the feature of easy programming and fault tolerance, MapReduce has become popular and been widely utilized by many organizations/institutes, such as Amazon, Yahoo, etc., to tackle their data-intensive applications in recent years. Based on this model, Apache develops an open-source software framework called Hadoop [2]. Other MapReduce implementations can be found in [3][4][5][6].

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_8

MapReduce performs a job by breaking it into smaller map tasks and reduce tasks, running these tasks in parallel on a large-scale cluster of commodity machines, called a MapReduce cluster, and utilizing a distributed file system, such as Google File System [7] or Hadoop Distributed File System [8], to store the job's input and output data. In general, a MapReduce implementation, e.g., Apache Hadoop [2], has two master servers. One is called JobTracker, which coordinates all jobs running on the MapReduce clusters, performs task assignment for each job, and monitors the progresses of all map and reduce tasks. The other is called NameNode, which manages the distributed filesystem namespace and processes all read and write requests.

These two master servers can run either on the same machine or on two separate machines. But the machine(s) might fail or crash due to various reasons, such as hardware and/or software faults, network link issues, and bad configuration [9]. Yahoo has experienced three NameNode failures caused by hardware problems [9]. In a system with ten thousands of highly reliable servers with MTBF of 30 years, a node fails each day in average [10]. Although JobTracker and NameNode are run on reliable hardware, they may fail some day. When JobTracker or NameNode crashes, the operation of a MapReduce cluster will be interrupted, i.e., all MapReduce jobs regardless of what states they are right now cannot proceed and be completed. This is unacceptable for time-critical data analysis for decision making and business intelligence. Unreliable JobTracker and NameNode will also impact the operations of those companies using MapReduce to process their data.

Redundancy mechanisms are common methods to improve system reliability [11][12]. Some systems [1][13][14][15][17][18] employed cold-standby redundancy. When a master node fails, a cold-standby node is used to takeover for it. This mechanism can significantly enhance system reliability since the failure rate of a node in its cold-standby mode is zero [19]. However, the cold-standby node has to restart the operation of the mater node from scratch since it does not hold any states of the master node, consequently leading to a long downtime. Some other systems [2][16] use warm-standby redundancy to improve reliability and shorten system downtime. Hadoop [2] provides a checkpoint node to periodically back up the file-system namespace of NameNode. When NameNode fails, the namespace copy held by the checkpoint node can be used to manually restart NameNode. But the namespace copy might not be out of date when NameNode crashes. To solve this problem, systems [16][20][21][22] utilized hot-standby redundancy to achieve a fast takeover. Hadoop [2] provides a backup node to maintain the up-to-date copy of the namespace of NameNode all the time. It might crash before the failure of NameNode, thus unable to extend the operation of NameNode. Besides, Hadoop does not offer any redundancy for its another master server JobTracker, therefore, causing another single point of failure.

In this chapter, we propose a master server redundant mechanism called the Reliable Hybrid Redundant System (ReHRS for short) which employs a hot-standby server (HSS for short) and a warm-standby server (WSS for short) to enhance the reliability and availability of the MapReduce master server. The functions of the HSS and WSS are the same as those of the master server, but the two servers do not serve clients and workers while in their standby modes. The HSS synchronizes itself with the master

server to achieve a fast takeover and continue any unfinished operations when the master server fails. The WSS periodically wakes up to backup the master server's metadata. After that, it sleeps again to reduce its failure probability. To continue the operations of the master server and HSS when any of them unexpectedly fails, we present a failure detection algorithm for the two servers to mutually detect each other's failure and launch an appropriate takeover process to continue each other's operation. In addition, we introduced a dynamic warmup mechanism for the WSS to warm itself up when discovering that the master server or HSS is unstable. The goal is to enable the WSS to quickly play the role of the HSS before the master server or HSS actually fails.

Extensive experiments and simulations are conducted to demonstrate and compare the ReHRS and three existing state-of-the-art schemes, including No-Redundant scheme (NR for short) which is similar to the design of Hadoop [2] for JobTracker, Hot-Standby-Only scheme (HSO for short) [21], and Warm-Standby-Only scheme (WSO for short) [16] in terms of takeover delays and impacts on the performances and resource consumptions of JobTracker and NameNode. In addition, the performance of the WSS is also evaluated.

The rest of this chapter is organized as follows. Section 2 introduces background and related work of this chapter. Section 3 presents the ReHRS. The simulation and experimental results are described and discussed in Section 4. Section 5 concludes this chapter and outlines our future studies.

2 Background and Related Work

2.1 Background

Figure 1 shows the execution flow of a job *J* on a MapReduce cluster. In step 1, a client requests a job ID from JobTracker. Then, he/she requests a set of workers' locations from NameNode in step 2. In step 3, based on the worker locations replied by NameNode, the client stores all his/her job resources, including a job JAR file, configuration files, and a number of input chunks, in the corresponding worker storages. In steps 4 and 5, the client submits *J* to JobTracker, and JobTracker initiates *J*, respectively. After retrieving the chunk information of *J* from NameNode in steps 6 and 7, JobTracker in step 8 assigns each map task of *J* to a worker, called mapper, and assigns each reduce task of *J* to a worker, called reducer. A mapper or reducer before executing its assigned task has to retrieve the corresponding job resources from the distributed file system by consulting NameNode. Each mapper runs the assigned map task to produce intermediate <key,value> results, stores the results locally, and then notifies JobTracker with the location of the results. Once all map tasks are finished, JobTracker informs all the reducers to start their reduce tasks. Each reducer then acquires a part of intermediate <key,value> results from all mappers, runs the assigned reduce task, and stores the result it generates into the distributed file system with the help of NameNode. After all reducers finish their tasks, JobTracker informs the client of the completion of *J*.

It is clear that JobTracker and NameNode are two critical components during the job execution. If JobTracker fails, clients cannot submit jobs, and mappers cannot notify JobTracker of the locations of the intermediate results that they generate, consequently, causing all subsequent reducers unable to start their tasks. On the other hand, a failed NameNode cannot provide input chunk information for JobTracker to perform task assignments, worker locations for mappers and reducers to obtain their required job resources, and available worker locations for reducers to store the final results, implying that the corresponding jobs cannot proceed or be completed. To ensure normal operation of a MapReduce cluster, both JobTracker and NameNode must be reliable and available since the startup of the system. That is why we would like to enhance the reliabilities and availabilities of these two master servers.

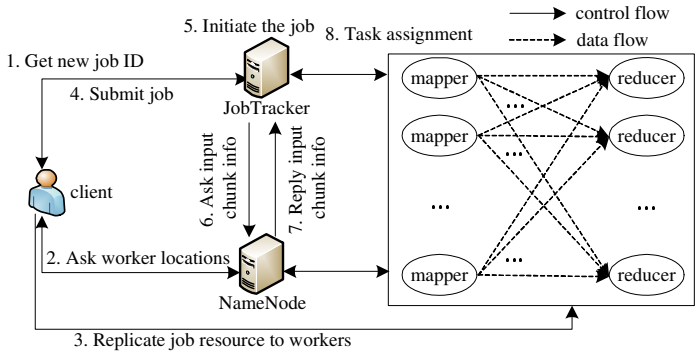


Fig. 1. The execution flow of a MapReduce job *J* on a MapReduce cluster

2.2 Related Work

Redundant mechanisms are the common methods used by a system to improve its reliability. As mentioned above, these methods can be divided into four types: cold-standby redundancy, warm-standby redundancy, hot-standby redundancy, and fully parallel redundancy [11][12].

Cold-Standby Redundancy

Many systems [1][13][14][15][17][18] used cold-standby mechanisms to achieve a higher reliability. For example, Zheng [17] enhanced MapReduce fault tolerance by assigning several backup nodes to each map task based on data locality [26]. When a map task fails on one node, one of its backup nodes will retrieve the required data chunk from local or nearby disks and perform the task immediately. But the task has to be executed from the very beginning. To prevent a straggler, i.e., a node with a poor performance, from slowing down a job execution, MapReduce [1] additionally executes a backup task for each task on another available cold-standby node so that the corresponding job can be more quickly completed. Current MapReduce implementation only adopted a cold-standby mechanism to provide fault tolerance for its

workers, rather than for its master server. The main reason is that a cold-standby node does not keep any state of the master-server. Hence, when the master server fails, the cold-standby node cannot takeover for it and recover it to the state before the failure occurs, implying that the takeover is useless.

Warm-Standby Redundancy

In a warm-standby redundancy mechanism, the states of the master server are periodically replicated to a warm-standby node. After that, the node sleeps to reduce its failure probability so as to improve system reliability. When the master server fails, the state replica can be used to restart the operation of the master server. The checkpoint node provided by Hadoop [2] is an example. It periodically backs up the namespace of NameNode. When NameNode fails, the namespace copy held by the checkpoint node can be used to restart NameNode, but the checkpoint node might not be able to provide the latest namespace when NameNode fails, consequently failing to provide a complete takeover. Besides, automatic takeover is not supported by the checkpoint node. This scheme, as the WSO mentioned above, will be evaluated later in this chapter. The warm-standby redundancy has been also employed by other systems. For instance, Leu et al. [16] improved the intrusion detectors' fault-tolerant capability in their grid intrusion detection platform by deploying a set of detectors. When one detector fails, another will be assigned to continue the failed detector's unfinished task.

Hot-Standby Redundancy

Some systems [2][16][20][21][22] utilized hot-standby mechanisms (also called *ordinary parallel* [12]) to speedup their failover processes. Hadoop [2] provides a backup node to maintain an in-memory, up-to-date copy of the file-system namespace of NameNode. The backup node can continue the operation of NameNode when NameNode fails. Nevertheless, the backup node might crash before the failure of NameNode since these two nodes run simultaneously in parallel, therefore, resulting in insufficient reliability and availability for MapReduce. Besides, Hadoop does not offer any redundant mechanism for its another master server JobTracker, thus leading to another single point of failure. Paratus [20] introduced an instantaneous failover approach by frequently replicating a primary virtual machine's states to a backup virtual machine. When the primary one crashes, the backup can immediately takeover for it. In the backup scheme [21], one of a set of hot-standby nodes is elected to takeover for a failed primary server so that Hadoop operation can be continued. However, the takeover process might not be able to finish in a short period of time since the new primary server before providing its services has to setup IP configuration and retrieve complete transient metadata from other worker nodes. This scheme, as the HSO mentioned above, will be evaluated later in this chapter. Alvaro et al. [22] also used several hot-standby nodes to provide NameNode with a fast failover in their presented data-centric programming model, but they did not address JobTracker failure.

Fully Parallel Redundancy

Fully parallel mechanisms are employed by many schemes [24][24][25] to increase their system reliability and offer a transparent takeover. He et al. [23] proposed a symmetric replication model for multiple metadata servers to achieve a reliable distributed storage system. In this model, a server failure will not disrupt the metadata service and lose service states, but group communication [27][28] for metadata update among all servers might lead to a longer write latency and higher network overhead. Chen et al. [24] introduced a replication-based mechanism to build highly available metadata servers. In this mechanism, a server replicates newly updated metadata to other servers so that when it fails, another can takeover for it immediately. But metadata replication and synchronization may be a huge burden for these servers and might increase request response time since subsequent requests cannot be served immediately. Marozzo et al. [25] managed master node failure by presenting a Peer-to-Peer (P2P) MapReduce framework, in which all master nodes are partitioned into sets to avoid the case in which metadata synchronization is performed throughout the system. In this framework, each master node in a set has to hold other master nodes' latest job states and act as a backup node for these master nodes. Nevertheless, this design may also decline the performance of the master nodes inside a set.

3 The Proposed System

Figure 2 depicts the architecture of the ReHRS, in which the master server and HSS periodically send heartbeats to each other and detect each other's failure. When discovering that the other server is not stable, they request the WSS to warm itself up. When making sure that the other server is failed, the surviving server initiates a corresponding takeover process to continue the operation of the failed one.

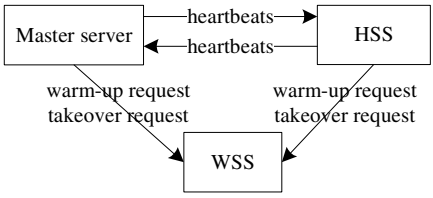


Fig. 2. The architecture of the ReHRS

In Sections 3.1, 3.2, and 3.3, we will, respectively, describe what the master server's metadata is composed of, how the HSS synchronizes its status with the master server, and how the WSS periodically backs up the metadata. The proposed failure detection algorithm, warmup mechanism, and takeover processes will detail in Sections 3.4, 3.5, and 3.6.

3.1 Metadata

In the ReHRS, the master server maintains two types of metadata. The first is persistent metadata (P-metadata for short), which is the execution result of an operation and is infrequently or never changed since it is generated. The master server usually keeps it in its P-metadata file/database. For examples, the information concerning jobs and the access control lists of files are respectively JobTracker's and NameNode's typical P-metadata.

The other type is transient metadata (T-metadata for short), which refer to frequently updated data. Hence, it is usually kept in the master server's memory to accelerate all required processing. For instance, the progresses/statuses of jobs sent by cluster workers that run these jobs and the storage locations of those data chunks delivered by the cluster workers that hold these chunks are, respectively, JobTracker's and NameNode's T-metadata.

3.2 State/Metadata Synchronization

In the ReHRS, whenever a write operation is initiated or completed, the master server generates a log record L_v and sends L_v to the HSS so that the two servers can keep the same state/metadata, where $v \geq 1$ is an incremental record ID.

Table 1. The fields of L_v

Record ID	Timestamp	P-metadata	Client/Worker ID	Operation type

L_v as shown in Table 1 consists of five fields. The timestamp keeps the time point when L_v is generated, the client/worker ID illustrates the client/worker that requests the operation, and the operation type field shows which type that L_v belongs to. L_v can be any of the following three operation types: initiated operation (IOP for short), response-required finished operation (RR-FOP for short), and response-unrequired finished operation (RU-FOP for short).

IOP means that L_v records information concerning an operation initiated by the master server. Since the execution result (i.e., P-metadata) has not been generated, the P-metadata field is null. RR-FOP means that L_v records information concerning an operation requested by a client/worker, and this operation have been finished by the master server. So the P-metadata and client/worker ID fields must be filled with the corresponding values. RU-FOP means that L_v records information concerning an operation both initiated and finished by the master server. Hence, the client/worker ID field must be null, but the P-metadata field must be non-null. The synchronization process between the master server and HSS is as follows.

1. The master server first inserts L_v into its journal and sends L_v to the HSS.
2. On receiving L_v, the HSS inserts it into its journal and checks the operation type of L_v.

(a) If it is "IOP", the HSS marks the state of L_v as "completed" and replies the master server with a message <v, "IOP", "cmp"> where cmp stands for completion, telling the master server that it has successfully stored L_v. With the record, the HSS can realize that the corresponding operation has been initiated.
(b) If it is "RR-FOP", implying that the P-metadata and client/worker ID of L_v are not null, the HSS updates its own P-metadata file/database with the P-metadata recorded in L_v, marks the state of L_v as "synchronous", and replies the master server with a message <v, "RR-FOP", "syn">, and waits for the completion message of the operation returned by the master server.
(c) If it is "RU-FOP", the HSS updates its own P-metadata file/database with the P-metadata recorded in L_v, replies the master server with a message <v, "RU-FOP", "cmp">, and marks the state of L_v as "completed", indicating that this operation has been completely performed and can be ignored when the HSS acts as the master server in the future.

3. On receiving the reply R from the HSS, the master server checks R.
(a) If R = <v, "IOP", "cmp">, the master server marks the state of L_v as "completed" for reminding itself that the HSS has recorded L_v.
(b) If R = <v, "RR-FOP", "syn">, the master server immediately updates its own P-metadata file/database with the P-metadata recorded in L_v, responds to the corresponding client/worker with the P-metadata, and marks the state of L_v as "completed" for reminding itself that the HSS has recorded the operation. After that, it replies a message <v, "cmp"> to the HSS.
(c) If R = <v, "RU-FOP", "cmp">, the master server immediately updates its own P-metadata file/database with the P-metadata recorded in L_v and marks the state of L_v as "completed".
4. Upon receiving <v, "cmp">, the HSS marks the state of L_v as "completed", indicating that it can ignore this operation when taking over for the master server.

The above process has three synchronizations flows as illustrated in Figure 3. With this process, the HSS can synchronize itself with the master server. Further, the HSS can realize which operation is initiated, finished, and whether the corresponding response is returned to the concerning requesters or not through all log records it maintains. This information also enables the HSS to continue any unfinished operations when the master server fails.

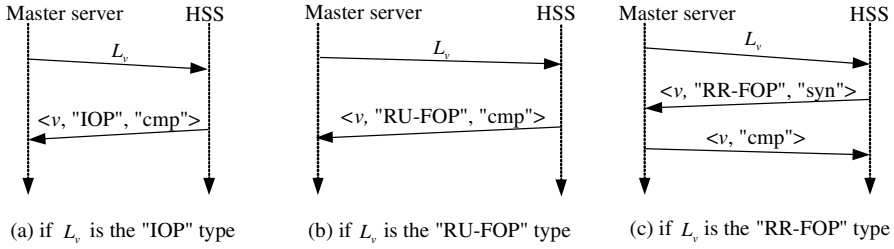


Fig. 3. Synchronization flows between the master server and HSS

On the other hand, upon receiving T-metadata TM from cluster workers, the master server uses TM to update its in-memory T-metadata and forwards TM to the HSS without generating a log record. On receiving TM, the HSS accordingly update its in-memory T-metadata without replying a synchronization message. The purpose is to reduce the burden of the master server and underlying network since T-metadata is updated frequently.

3.3 Periodical P-Metadata Backup/Update

If the WSS does not keep any metadata of the master server, it will take a long time to takeover for the HSS when required. To alleviate this problem, the WSS periodically wakes up and requests the log records it lacks from the HSS to update its P-metadata file/database. This can not only avoid increasing the master server's burden, but also guarantee that the updated P-metadata file/database is consistent with that of the master server since the HSS's and master server's journals are synchronized. The update process is as follows.

1. The WSS first sends a message <"update", $maxID$> to the HSS, where $maxID$ is the maximum ID of the log records currently collected in the WSS's journal.
2. On receiving <"update", $maxID$>, the HSS retrieves log records $L_{maxID+1}$, $L_{maxID+2}$, …, and L_K from its journal where K is the largest record ID of the log record currently collected in the HSS's journal. After that, it sends these log records to the WSS.
3. On receiving these log records, the WSS sequentially inserts them into its journal.
4. For each L_j where $maxID < j \leq K$, the WSS checks to see whether it should update its P-metadata file/database with the P-metadata recorded in L_j or not. If L_j is of the "IOP" type, it skips the update since the P-metadata filed of L_j is empty. Otherwise, it performs the update.
5. The WSS sleeps again.

Note that the WSS does not backup the master server's T-metadata. The reason is stated above.

3.4 Failure Detection

Due to unpredictable errors or faults, both the master server and HSS might fail at any moment. In the ReHRS, the two servers mutually send a heartbeat to each other every predefined sending time period (STP) to indicate that they are still alive and available. We assume that at least one network link is available for the master server, HSS, and WSS to communicate, and the heartbeat transmission delays between the master server and HSS are constant. In other words, the two servers can receive each other's heartbeats in each receiving time period (RTP) when both of them operate normally, where RTP≈STP. However, due to system busy or unstable, any of them might delay sending its heartbeats.

The master server and HSS use the failure detection algorithm listed in Figure 4 to detect each other's failure. Whenever RTP times out, each of them, denoted by E, checks to see whether it has received a heartbeat from the other, denoted by Q, during the RTP or not. If no, α is increased by 1, where α represents the total number of consecutive heartbeats that E has not received from Q.

When $\alpha = x$, implying that E has not received Q's heartbeats for x consecutive RTPs where x is a predefined threshold, $x \in N^+$, E assumes that Q is unstable, claims itself a commander, and requests the WSS to warm itself up. This can mitigate the phenomenon that due to holding out-of-date metadata, the WSS might be unable to make itself ready to act as the HSS in a short period of time. The details of the WSS's warmup mechanism will be described later. When $\alpha = y$, E assumes that Q has failed. If E is the master server, it informs the WSS to takeover for Q, i.e., the HSS, without changing its role. However, if E is the HSS, it immediately takes over for the master server and requests the WSS to act as the HSS.

```
The failure detection algorithm:
Input: heartbeats from Q;
Output: a warm-up or takeover decision;
Procedure:
Let α = 0;
while RTP times out
{  if E has not received  Q's heartbeats during the RTP
   {  α = α + 1;
      if α = x    /* x ∈ N+ is a predefined threshold. */
      { E claims itself as a commander and requests the WSS to  warm itself up;
      }
      else if α = y    /* y > x. */
      {
           if E is the HSS
           { E takes over for Q and requests the WSS to take over for the HSS;
             stop;
           }
           Else        /* E is the master server. */
           { E requests the WSS to take over for Q; stop;
           }
      }
   }
}
```

Fig. 4. The failure detection algorithm for the master server and HSS, where α is the number of consecutive RTPs during which E has not received heartbeats from Q

3.5 Dynamic Warmup Mechanism

The warmup mechanism initiates when the master server or HSS behaves unstable, and this mechanism stops when both the master server and HSS operate normally.

This dynamic property enables the WSS to update its metadata/status to the latest one before it is requested to act as the HSS so as to speedup its takeover process.

When both the master server and HSS are unstable, the WSS may be requested to warm itself up by both of them. In this situation, the WSS might receive redundant P-metadata sent by the two commanders. To avoid the WSS from storing duplicate log records and disordering the sequence of P-metadata, each time when the WSS receives a log record from a commander, it checks to see whether it has received the record or not. The process is as follows.

1. The WSS sends a message <"warm-up", $maxID$> to the commander.
2. Upon receiving <"warm-up", $maxID$>, the commander retrieves log record L_r from its journal, computes a hash value H_r for L_r, and sends the pair $\langle L_r, H_r \rangle$ to the WSS where r ranges from $maxID + 1$ to Z, and Z is the largest ID of the log records collected in the journal. Meanwhile, the commander sends the T-metadata that it currently has to the WSS.
3. On receiving the T-metadata, the WSS immediately loads it in its memory.
4. On receiving $\langle L_r, H_r \rangle$, the WSS compares H_r with all the hash values previously stored in its hash pool. If $H_r = H_r{}'$ where $H_r{}'$ is a hash value in the hash pool, L_r will be dropped. Otherwise, the WSS inserts L_r into its journal and stores H_r in the hash pool.
5. Following the sequence of $r = maxID + 1, maxID + 2, \ldots, Z$, the WSS checks the operation type of L_r to see whether it needs to update its P-metadata file/database or not. If L_r is of the "IOP" type, the WSS skips the update. Otherwise, it performs the update.

When the commander requests the WSS to stop warming up, it stops sending data to the WSS. Then the WSS goes to sleep once completing the current warm-up process.

3.6 Takeover Process

Assume that the master server, HSS, and WSS are respectively run on three different nodes N_1, N_2, and N_3. Figure 5 illustrates all possible state transitions of these nodes during the lifetime of the ReHRS. Notation N_u^S represents node N_u's state (also called role) is S, where u= 1, 2, 3 and $S \in \{M, H, W, n/a\}$ in which M, H, W, and n/a , respectively, stand for the master server, HSS, WSS, and unavailable. The ReHRS starts with the state $\{N_1^M, N_2^H, N_3^W\}$ and crashes in the state $\{N_1^{n/a}, N_2^{n/a}, N_3^{n/a}\}$. Note that the transitions from $\{N_1^M, N_2^H, N_3^W\}$ to $\{N_1^M, N_2^H, N_3^{n/a}\}$, $\{N_1^{n/a}, N_2^M, N_3^{n/a}\}$, $\{N_1^M, N_2^{n/a}, N_3^{n/a}\}$, or $\{N_1^{n/a}, N_2^{n/a}, N_3^{n/a}\}$ are not depicted in Figure 5 since N_3 is unlikely to fail while in its warm-standby mode. In addition, node repair is not considered in this chapter, i.e., any node cannot act as a server once it is out of work. Thus a state with $N_u^{n/a}$ cannot transit to the states with N_u^M, N_u^H, or N_u^W.

Let M_{bef}, H_{bef}, and W_{bef} (M_{aft}, H_{aft}, and W_{aft}) are respectively nodes act as the master server, HSS, and WSS before (after) a state transition. The state transition from $\{N_1^M, N_2^H, N_3^W\}$ to $\{N_1^{n/a}, N_2^M, N_3^H\}$ means that N_1 fails, and N_2 (N_3) changes its role from the HSS (the WSS) to the master server (the HSS). Hence, $<M_{bef}, H_{bef}, W_{bef}>=<N_1, N_2, N_3>$ and $<M_{aft}, H_{aft}, W_{aft}>=<N_2, N_3, n/a>$ in which $W_{aft} = n/a$ because no extra

nodes will act as the WSS. The state $\{N_1^M, N_2^H, N_3^W\}$ transiting to $\{N_1^M, N_2^{n/a}, N_3^H\}$ implies that N_2 fails, N_1 continues serving as the master server, and N_3 acts as the HSS. Thus $<M_{aft}, H_{aft}, W_{aft}>=<N_1, N_3, n/a>$. When $\{N_1^{n/a}, N_2^M, N_3^H\}$ transits to $\{N_1^{n/a}, N_2^M, N_3^{n/a}\}$, or $\{N_1^M, N_2^{n/a}, N_3^H\}$ changes to $\{N_1^M, N_2^{n/a}, N_3^{n/a}\}$, no takeover will be performed since no nodes can substitute for the failed ones. When a state changes to $\{N_1^{n/a}, N_2^{n/a}, N_3^M\}$, N_3 will be the master server, i.e., $<M_{aft}, H_{aft}, W_{aft}>=<N_3, n/a, n/a>$. If a state turns to $\{N_1^{n/a}, N_2^{n/a}, N_3^{n/a}\}$, $<M_{aft}, H_{aft}, W_{aft}>=<n/a, n/a, n/a>$.

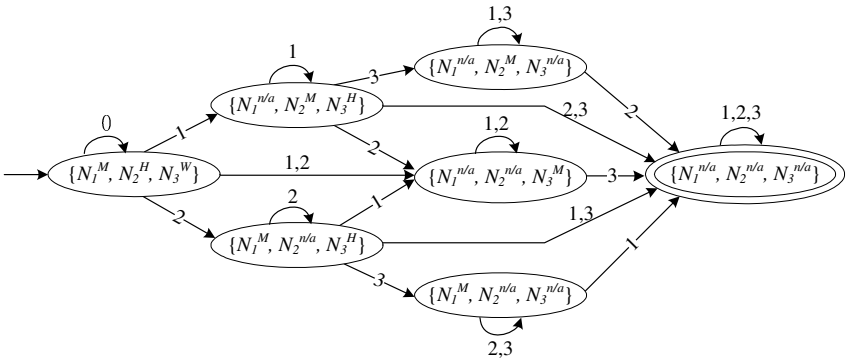


Fig. 5. The state transition graph for nodes N_1, N_2, and N_3 during the lifetime of the ReHRS, where number 0 shown on the arrow stands for all nodes normally operate, and 1, 2, and 3 represent that N_1, N_2, and N_3 fail, respectively

The Process of Taking over for the HSS

H_{aft} takes over for H_{bef} only when $H_{aft} \neq H_{bef}$, H_{bef} fails, and H_{aft} is not *n/a*. The takeover process is as follows. H_{aft} changes its IP address to H_{bef}'s and notifies M_{aft} (recall the master server) of the completion of the takeover. Then it starts all the HSS's functions, including receiving P-metadata and T-metadata sent by M_{aft}, synchronizing P-metadata with M_{aft}, sending its heartbeats to M_{aft}, and initiating the failure detection algorithm to monitor M_{aft}'s heartbeats.

However, H_{aft} might be still warming itself up when it is requested to takeover for H_{bef}, i.e., having not finished the warmup process. In this situation, H_{aft} keeps updating its in-memory T-metadata with the new T-metadata received from M_{aft}. However, for each newly receiving log record sent by M_{aft}, H_{aft} buffers it in its memory and instantly responds to M_{aft} with a corresponding synchronization or completion message. All buffered log records will be sequentially inserted in H_{aft}'s journal, and the non-null P-metadata conveyed in these log records will be sequentially applied to H_{aft}'s P-metadata file/database when the warmup process completes. With this takeover process, the response delays of M_{aft} can be dramatically reduced, and the sequence of P-metadata can be preserved.

The Process of Taking over for the Master Server

M_{aft} takes over for M_{bef} only when $M_{aft} \neq M_{bef}$, M_{bef} fails, and M_{aft} is not *n/a*. The process is as follows. First, M_{aft} changes its IP address to M_{bef}'s and starts serving clients

and workers. M_{aft} will also replicate new T-metadata to H_{aft}, synchronize new log records with H_{aft}, send its heartbeats to H_{aft}, and initiate the failure detection algorithm to monitor H_{aft}'s heartbeats if H_{aft} is now not *n/a*. Further, M_{aft} has to continue those operations unfinished by M_{bef}. As mentioned above, a completed operation has two log records to indicate its beginning and completion. M_{aft} first extracts the operation ID from each "IOP" type log record, e.g., Log_1, and finds the other log record of the same operation ID, e.g., Log_2. If M_{aft} cannot find Log_2, M_{aft} reperforms the corresponding operation since it considers that M_{bef} has not finished it. If M_{aft} finds that Log_2 is a "RU-FOP" record, M_{aft} ignores it because the corresponding operation has been completed by M_{bef}. Similarly, if Log_2 is a "RR-FOP" record, and its state is "completed", the record will be neglected. But if the state is "synchronous", implying that M_{bef} failed before responding to the corresponding client/worker, M_{aft} immediately responds the client/worker with the P-metadata conveyed in the record. By dealing with unfinished operations, M_{aft} continues what has not been done by M_{bef}. Therefore, all running jobs can proceed and be completed successfully.

4 Performance Evaluation

We implemented the ReHRS, NR [2], HSO [21], and WSO [16] in Java language and built a test cluster consisting of 1030 virtual nodes and 27 switches as illustrated in Figure 6. All nodes are connected through a 1 Gbps Ethernet network created by the Network Simulation (NS2) [29], which is a simulation tool supporting TCP, routing, and multicast protocols over wired and wireless networks. Each node runs Ubuntu 11.04 with an AMD Athlon(tm) 2800+ CPU, 1 GB memory, and a 80 GB disk drive. During the experiments, 1024 nodes are deployed as workers. The remaining six nodes connected to different switches act as JobTracker, JobTracker's HSS, JobTracker's WSS, NameNode, NameNode's HSS, and NameNode's WSS when the ReHRS is tested. When the HSO (WSO) is evaluated, the six nodes act as JobTracker, JobTracker's two hot-standby servers (two warm-standby servers), NameNode, and NameNode's two hot-standby servers (two warm-standby servers). As the NR is employed, two of the six nodes are deployed to act as JobTracker and NameNode.

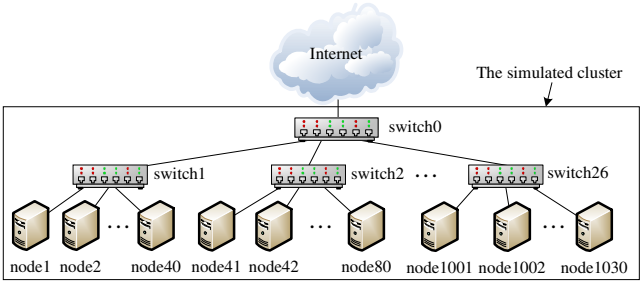


Fig. 6. The topology of our simulated cluster

4.1 Takeover Delays Evaluation

We evaluated the takeover delays for these schemes on eight different JobTracker and NameNode failure types as listed in Table 2 and four different STPs, i.e., STP =1, 10, 100, and 1000 ms. In this study, takeover delays is defined as the time period from the moment when a server fails to the moment when another node acts as the server, comprising the server failure detection time, IP address reconfiguration time, P-metadata retrieval time, and T-metadata retrieval time.

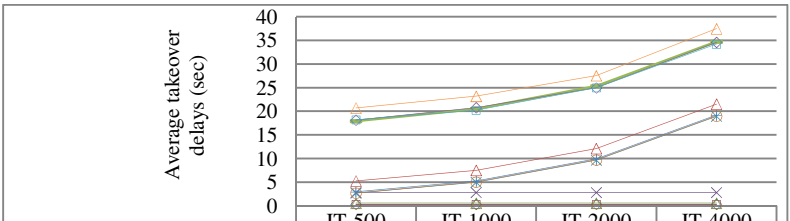
Table 2. Failure types considered in the experiment

Failure Type	Description
JT-500	JobTracker fails while 500 jobs are running (Total T-metadata is about 97.66 MB).
JT-1000	JobTracker fails while 1000 jobs are running (Total T-metadata is about 195.32 MB).
JT-2000	JobTracker fails while 2000 jobs are running (Total T-metadata is about 390.63 MB).
JT-4000	JobTracker fails while 4000 jobs are running (Total T-metadata is about 781.25 MB).
NN-50%	NameNode fails when 50% of its memory is occupied by T-metadata.
NN-60%	NameNode fails when 60% of its memory is occupied by T-metadata.
NN-70%	NameNode fails when 70% of its memory is occupied by T-metadata.
NN-80%	NameNode fails when 80% of its memory is occupied by T-metadata.

When simulating each JobTracker (NameNode) failure type, we randomly disconnected JobTracker (NameNode) from the cluster for 30 times and estimate the average time required by each tested scheme to takeover for the failed JobTracker (NameNode). The basic workloads of JobTracker and NameNode are processing 500 write requests and 1000 read requests per second. Each write request generates two log records respectively indicating the initiation and completion of the corresponding operation, and each log record is 100 bytes in length. The T-metadata size of a running job is about 200 KB. The ReHRS's and WSO's warm-standby nodes back up their master server's P-metadata every 2 hours, yielding a backup data (i.e., log records) 686.65 ($= \frac{500*100*2*60*60*2}{1024*1024}$) MB in length. Assume that the heartbeat transmission delay is constant when each scheme is tested, implying that RTP=STP for all schemes. We set $y = 3$ for all tested schemes except the NR. In other words, the ReHRS, HSO, and WSO launch a takeover process when not receiving JobTracker's or NameNode's heartbeats for three consecutive RTPs. In addition, we assume that JobTracker and NameNode fail without any unstable behavior, so the time period from the moment when the failures occur to the moment when the corresponding takeover is initiated is at most 3 RTPs, i.e., 3 STPs.

Figure 7 illustrates the average takeover delays for all tested schemes on the four JobTracker failure types and four STPs except the NR since the NR does not provide any redundancy mechanism for JobTracker and NameNode. With the NR, the failed JobTracker must be recovered manually. The results show that the ReHRS outperforms the other three schemes. The ReHRS, respectively, takes 0.30, 0.32, 0.55, and 2.82 sec in average to takeover for the failed JobTracker when STP=1, 10, 100, and 1000 ms. The takeover delays are not influenced by these JobTracker failure types

because the HSS is always synchronous with JobTracker. The takeover delays only comprises the time of detecting JobTracker's failure and configuring the HSS's IP address to JobTracker's. According to our estimations repeated for 30 times, the average IP configuration time is about 294 ms with the standard deviation 24 ms for these STPs, implying that the ReHRS's takeover delays are mainly determined by STP. The speedups of the ReHRS's takeover delays on STP=1 ms is 9.4 (=2.82/0.30) times that on STP=1000 ms, 1.83 (=0.55/0.30) times that on STP=100 ms, 1.07 (=0.32/0.30) times that on STP=10 ms, showing that ReHRS with a smaller STP can considerably speedup its takeover process.



	JT-500	JT-1000	JT-2000	JT-4000
ReHRS on STP=1	0.30	0.30	0.30	0.30
ReHRS on STP=10	0.32	0.32	0.32	0.32
ReHRS on STP=100	0.55	0.55	0.55	0.55
ReHRS on STP=1000	2.82	2.82	2.82	2.82
HSO on STP=1	2.68	4.99	9.66	18.96
HSO on STP=10	2.70	5.02	9.69	18.98
HSO on STP=100	2.93	5.25	9.91	19.21
HSO on STP=1000	5.25	7.50	12.12	21.52
WSO on STP=1	17.92	20.50	25.30	34.76
WSO on STP=10	18.13	20.76	24.95	34.61
WSO on STP=100	18.21	20.25	24.98	34.23
WSO on STP=1000	20.72	23.20	27.54	37.49

Fig. 7. Average takeover delays (sec) for the ReHRS, HSO, and WSO on different JobTracker failure types and STPs. Note that the takeover delays of the NR is not shown since this scheme does not provide any redundancy mechanisms.

The average takeover delays of the HSO is higher as the number of running jobs increases since the HSO's hot-standby server only maintains P-metadata rather than T-metadata, i.e., the HSO's takeover delays consists of not only JobTracker failure detection time and IP address configuration time, but also JobTracker's T-metadata retrieval time. The speedups of the takeover delays on STP=1 ms range from 1.96 (=5.25/2.68) to 1.14 (=21.52/18.96) times that on STP=1000 ms, range from 1.09 (=2.93/2.68) to 1.01 (=19.21/18.96) times that on STP=100 ms, range from 1.01

(=2.70/2.68) to 1 (=18.98/18.96) times that on STP=10 ms, showing that a smaller STP cannot effectively reduce the takeover delays of the HSO. Table 3 lists the speedups of the takeover delays of the ReHRS as compared with those of the HSO. When STP=1 ms, the speedups range from 8.93 (=2.68/0.30) to 63.20 (=18.96/0.30) times those of the HSO on the four JobTracker failure types. As STP increases, the ReHRS's speedups decrease since its JobTracker failure detection times prolong its takeover delays. Nevertheless, the ReHRS is still faster than the HSO.

The takeover delays of the WSO is even longer than those of the ReHRS and HSO as more jobs are running because the WSO's takeover completes only when the P-metadata and T-metadata that the warm-standby server lacks are completely retrieved. In addition, we can see that reducing STP cannot effectively improve the WSO's takeover speed since WSO's takeover delays are dominated by the P-metadata retrieval time. Table 4 lists the takeover speedups of the ReHRS as compared with those of the WSO. It is apparent that the ReHRS outperforms the WSO.

Table 3. The speedups of the takeover delays of the ReHRS as compared with those of the HSO on JobTracker failure types

Failure type / STP (ms)	JT-500	JT-1000	JT-2000	JT-4000
1	8.93	16.63	32.20	63.20
10	8.44	15.69	30.28	59.31
100	5.33	9.55	18.02	34.93
1000	1.86	2.66	4.30	7.63

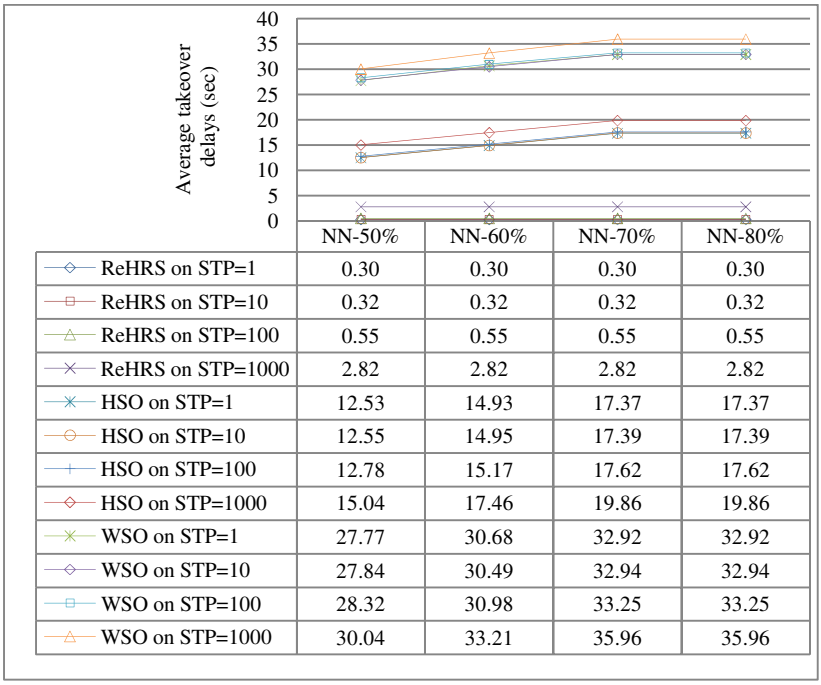
Table 4. The speedups of the takeover delays of the ReHRS as compared with those of the WSO on JobTracker failure type

Failure type / STP (ms)	JT-500	JT-1000	JT-2000	JT-4000
1	59.73	68.33	84.33	115.87
10	56.66	64.88	77.97	108.16
100	33.11	36.82	45.42	62.24
1000	7.35	8.23	9.77	13.29

Figure 8 shows the average takeover delays for all tested schemes (except the NR) on the four NameNode failure types. Similarly, the size of T-metadata occupying NameNode's memory does not influence the ReHRS's takeover delays on each STP. Hence, the ReHRS's results are identical to those shown in Figure 7. But these NameNode failure types considerably impact the takeover delays of the HSO and WSO, especially when NameNode holds more and more T-metadata in its memory. The results also indicate that these STPs are unable to lower the HSO's and WSO's takeover delays.

To show the impacts of these schemes with different STPs on the MapReduce master server, we estimated the average CPU utilizations, memory utilizations, read-request processing times, and write-request processing times of the master server. Here, NameNode was chosen as the target since its performance in processing read/write requests is easily influenced by STP. Note that NameNode is tested under the same workload mentioned above.

Figures 9 and 10, respectively, plot the average CPU and memory utilizations of NameNode when these schemes are employed on different STPs. The NR on any STPs did not affect the CPU and memory consumptions of NameNode in processing the workload. The reason is stated above. When the HSO is employed, more and more NameNode's resources are utilized as STP decreases. This is because that NameNode has to send its heartbeats to another two hot-standby servers and meanwhile detect their failures. The WSO causes the least overhead for NameNode as compared with the ReHRS and HSO. Nevertheless, NameNode has a heavy load when STP=1 ms because NameNode has to frequently send its heartbeats to the monitor node employed by the WSO.



	NN-50%	NN-60%	NN-70%	NN-80%
ReHRS on STP=1	0.30	0.30	0.30	0.30
ReHRS on STP=10	0.32	0.32	0.32	0.32
ReHRS on STP=100	0.55	0.55	0.55	0.55
ReHRS on STP=1000	2.82	2.82	2.82	2.82
HSO on STP=1	12.53	14.93	17.37	17.37
HSO on STP=10	12.55	14.95	17.39	17.39
HSO on STP=100	12.78	15.17	17.62	17.62
HSO on STP=1000	15.04	17.46	19.86	19.86
WSO on STP=1	27.77	30.68	32.92	32.92
WSO on STP=10	27.84	30.49	32.94	32.94
WSO on STP=100	28.32	30.98	33.25	33.25
WSO on STP=1000	30.04	33.21	35.96	35.96

Fig. 8. Average takeover delays (sec) for the ReHRS, HSO, and WSO on different JobTracker failure types and STPs. Similarly, the takeover delays of the NR is absent for the same reason stated above

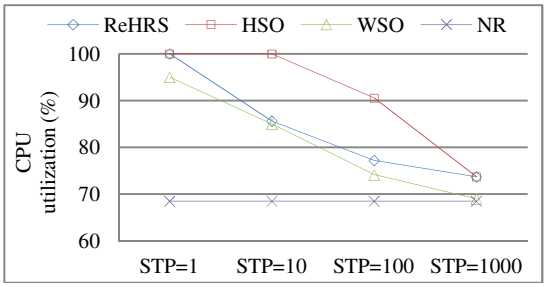


Fig. 9. The CPU utilizations of NameNode when the ReHRS, HSO, and WSO are employed with different STPs

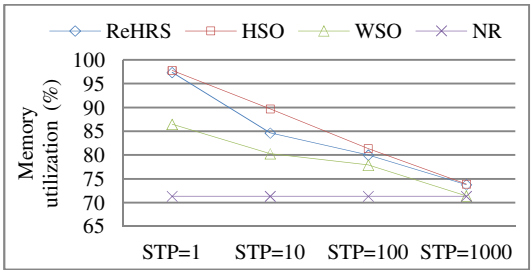
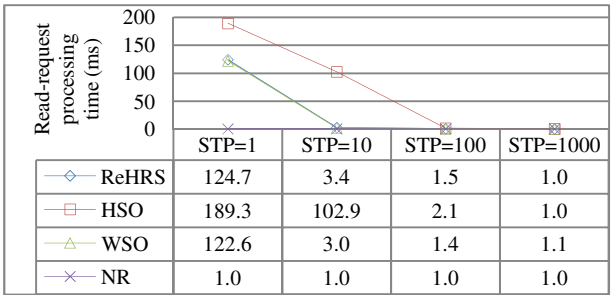


Fig. 10. The memory utilizations of NameNode when the ReHRS, HSO, and WSO are employed with different STPs

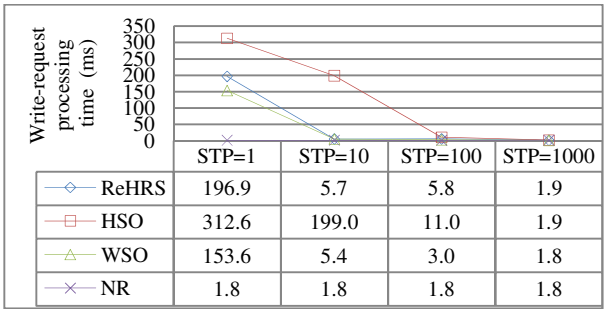
Figures 11 and 12, respectively, show the average read-request and write-request processing times of NameNode when these schemes are employed on different STPs. Because the NR does not run any failure detection, its performance exactly reflects the performance of NameNode, therefore is used as a baseline for comparison. When STP=1 ms, the ReHRS, HSO, and WSO all result in high request processing times, but the HSO consumed the highest processing time. When STP=10 ms, the processing times on the ReHRS and WSO dramatically decreased, but the processing time on the HSO was still very high. The reason is that the frequent heartbeat sending and receiving in the HSO burdens NameNode, consequently decreasing NameNode's performance. When STP increase to 100 ms or 1000 ms, the NameNode's request-processing times on the four schemes were very close, implying that the ReHRS, HSO, and WSO on the two STP settings did not impact the performance of NameNode.

The above results show that STP is an influential factor determining NameNode's performance in processing read and write requests. A smaller STP consumes more NameNode's resources and decreases NameNode's performance, while a larger STP has less impact on NameNode's resource consumption and performance.



	STP=1	STP=10	STP=100	STP=1000
ReHRS	124.7	3.4	1.5	1.0
HSO	189.3	102.9	2.1	1.0
WSO	122.6	3.0	1.4	1.1
NR	1.0	1.0	1.0	1.0

Fig. 11. The read-request processing times of NameNode when all tested schemes are employed with different STPs



	STP=1	STP=10	STP=100	STP=1000
ReHRS	196.9	5.7	5.8	1.9
HSO	312.6	199.0	11.0	1.9
WSO	153.6	5.4	3.0	1.8
NR	1.8	1.8	1.8	1.8

Fig. 12. The write-request processing times of NameNode when all tested schemes are employed with different STPs

4.2 Warmup Performance Evaluation

To show the performance of the WSS, we estimated the time required by the ReHRS's WSS to finish its warmup process, which is denoted by T_{remain}, and it can be calculated as

$$T_{remain} = T_{total} - T_{early} \tag{1}$$

where T_{total}, as shown in Figure 13, is the total time required by the WSS to finish its warmup process when it receives a warmup request from Q. If E fails right after the WSS finishes its latest P-metadata update, T_{total} comprises only the T-metadata retrieval time. However, if the failure occurs when the WSS just starts to update its P-metadata, T_{total} includes both P-metadata and T-metadata retrieval times. T_{early} is the time period from the moment when the WSS is requested to warm itself up to the moment when it is requested to takeover for the HSS. During T_{early}, the WSS keeps warming itself up to retrieve the P-metadata and T-metadata that it lacks. Given a fixed T_{total}, T_{remain} will be longer if T_{early} is shorter. On the contrary, if the

master server/HSS is unstable for a long time before it fails, T_{early} will be longer, meaning that the WSS has more time to make itself ready for acting as the HSS before the master server/HSS actually fails.

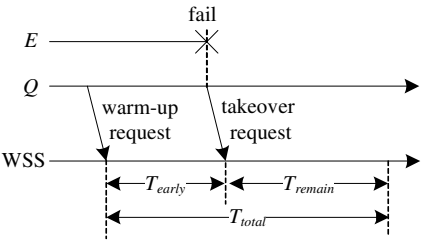


Fig. 13. The T_{total}, T_{early}, and T_{remain} for the ReHRS's WSS, in which E is the master server if Q is the HSS, and E is the HSS if Q is the master server

We evaluate the T_{total} of the WSS on each failure type listed in Table 2 and STP=1000 ms for 30 times to obtain the average T_{total}. The results are presented in Table 5. Then we use five different cases shown in Table 6 to estimate the average T_{remain} for the WSS. Each case represents how long JobTracker and NameNode behave unstable before they crash.

Table 5. Average T_{total} of the WSS on different failure types with STP=1000 ms

Failure type	T_{total} (sec)
JT-500	17.60
JT-1000	20.22
JT-2000	25.02
JT-4000	34.48
NN-50%	27.43
NN-60%	30.34
NN-70%	32.56
NN-80%	35.05

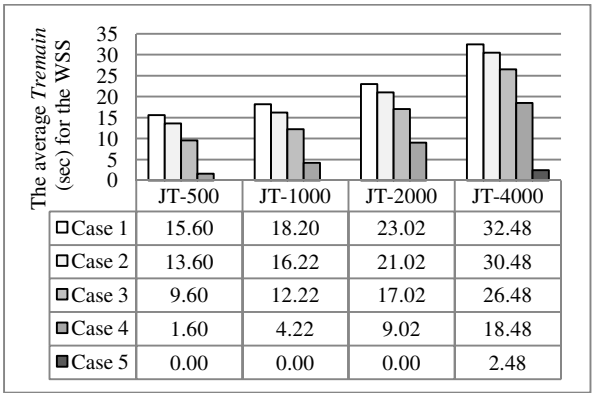
Table 6. Five cases of T_{early}, in which STP=1000 ms

Case	Description
1	T_{early} is 2 STPs, i.e., $T_{early} = 2$ sec.
2	T_{early} is 4 STPs, i.e., $T_{early} = 4$ sec.
3	T_{early} is 8 STPs, i.e., $T_{early} = 8$ sec.
4	T_{early} is 16 STPs, i.e., $T_{early} = 16$ sec.
5	T_{early} is 32 STPs, i.e., $T_{early} = 32$ sec.

Figures 14 and 15 illustrate the average T_{remain} for the WSS. For each case, T_{remain} increases when more jobs are running or more T-metadata occupy NameNode's memory. This is because that the WSS needs more time to retrieve the T-metadata it lacks. For each failure type, it is clear that the cases with a longerT_{early} lead to a shorter T_{remain}. In Case 5, the WSS can almost finish its warmup process

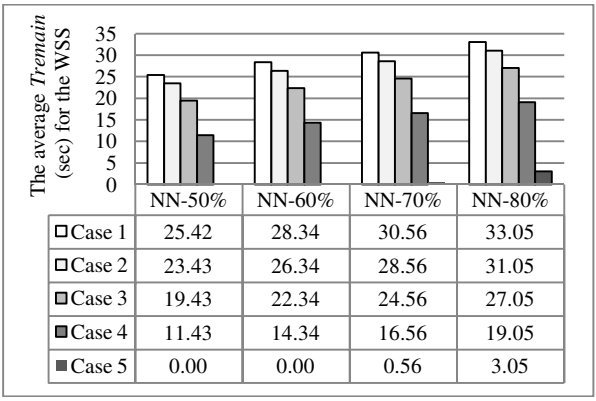
before JobTracker and NameNode actually fail. The results demonstrate that the warmup mechanism can adapt to the unstable behaviors of JobTracker and NameNode and enables the WSS to proactively warm itself up and act as the HSS quickly.

During T_{remain}, the WSS is unlikely to fail since its reliability is high. For example, assume that the WSS in its warmup or warmup mode follows a Poisson process with a failure rate $\lambda = 0.0001$ per hour, and when T_{remain} =33.05 sec (i.e., the maximum T_{remain} in our results), the reliability of the Warmup is about 0.99999 ($= e^{-0.0001*\frac{33.05}{60*60}}$), implying that the WSS has a very high possibility to finish its warmup process and fully acts as the HSS. Similarly, the surviving server (i.e., Q shown in Figure 13) is unlikely to crash during T_{remain} if it has the same failure rate, implying that the operation of the MapReduce master server can be continued.



	JT-500	JT-1000	JT-2000	JT-4000
Case 1	15.60	18.20	23.02	32.48
Case 2	13.60	16.22	21.02	30.48
Case 3	9.60	12.22	17.02	26.48
Case 4	1.60	4.22	9.02	18.48
Case 5	0.00	0.00	0.00	2.48

Fig. 14. The average T_{remain} for the ReHRS's WSS on different JobTracker failure types with STP=1000 ms



	NN-50%	NN-60%	NN-70%	NN-80%
Case 1	25.42	28.34	30.56	33.05
Case 2	23.43	26.34	28.56	31.05
Case 3	19.43	22.34	24.56	27.05
Case 4	11.43	14.34	16.56	19.05
Case 5	0.00	0.00	0.56	3.05

Fig. 15. The average T_{remain} for the ReHRS's WSS on different NameNode failure types with STP=1000 ms

5 Conclusion and Future Work

In this chapter, we propose the ReHRS to conquer the single-point-failure problem for MapReduce and improve MapReduce reliability and availability by employing the HSS to maintain the latest metadata of the master server and utilizing the WSS to further extend MapReduce lifetime. To show that the ReHRS is capable of providing a fast takeover, we evaluated the ReHRS and three schemes, the NR, HSO, and WSO, in a large simulated MapReduce cluster given eight different JobTracker and NameNode failure types and four different STPs. The simulation and experimental results show that the ReHRS's takeover delays are shorter than those of the NR, HSO, and WSO regardless of which failure type or STP is employed, implying that the ReHRS can effectively raise the availability of the MapReduce master server. In addition, the results demonstrate that the warmup mechanism enables the WSS to warm itself up while JobTracker and NameNode are unstable and shorten the time required to act as the HSS.

Acknowledgments. The work was partially supported by the GREENs project of TungHai University and the National Science Council, Taiwan under Grants NSC 101-2221-E-009-003-MY3 and NSC 101-2628-E-009-024-MY3. The authors are grateful to the National Center for High-performance Computing for computer time and facilities.

References

1. Dean, J., Ghemawat, S.: Mapreduce: Simplified Data Processing on Large Clusters. Communication of the ACM 51(1), 107–113 (2008)
2. Apache Hadoop, `http://hadoop.apache.org` (March 07, 2012)
3. The Disco project, `http://discoproject.org` (March 17, 2012)
4. Gridgain, `http://www.gridgain.com` (April 15, 2012)
5. MapSharp, `http://mapsharp.codeplex.com` (May 7, 2012)
6. Skynet, `http://skynet.rubyforge.org` (May 13, 2012)
7. Ghemawat, S., Gobioff, H., Leung, S.T.: The Google file system. In: Proceedings of the ACM Symposium on Operating Systems Principles, pp. 29–43. ACM, New York (2003)
8. Shvachko, K., Kuang, H., Radia, S., Chansler, R.: The Hadoop Distributed File System. In: Proceedings of the IEEE Symposium on Mass Storage Systems and Technologies, Incline Village, NV, USA, pp. 1–10 (2010)
9. Hadoop Wiki, NameNodeFailover, `http://wiki.apache.org/hadoop/NameNodeFailover` (September 9, 2011)
10. Dean, J.: Designs, lessons and advice from building large distributed Systems, Keynote slides at `http://www.cs.cornell.edu/projects/ladis2009/talks/dean-keynote-ladis2009.pdf` (September 20, 2011)
11. Loques, O.G., Kramer, J.: Flexible Fault Tolerance for Distributed Computer Systems. IEE Proceedings-E on Computers and Digital Techniques 133(6), 319–337 (1986)
12. Shooman, M.L.: Reliability of Computer Systems and Networks: Fault Tolerance, Analysis, and Design. John Wiley & Sons Inc., New York (2002)

13. Sinaki, G.: Ultra-Reliable Fault Tolerant Inertial Reference Unit for Spacecraft. In: Proceedings of the Annual Rocky & Mountain Guidance and Control Conference, San Diego, CA, pp. 239–248 (1994)
14. Pandey, D., Jacob, M., Yadav, J.: Reliability Analysis of a Powerloom Plant with Cold-Standby for its Strategic Unit. Microelectronics and Reliability 36(1), 115–119 (1996)
15. Kumar, S., Kumar, D., Mehta, N.P.: Behavioural Analysis of Shell Gasification and Carbon Recovery Process in a Urea Fertilizer Plant. Microelectronics and Reliability 36(4), 671–673 (1996)
16. Leu, F.Y., Yang, C.T., Jiang, F.C.: Improving Reliability of a Heterogeneous Grid-based Intrusion Detection Platform using Levels of Redundancies. Future Generation Computer Systems 26(4), 554–568 (2010)
17. Zheng, Q.: Improving MapReduce Fault Tolerance in the Cloud. In: Proceedings of the IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum, Atlanta, CA, pp. 1–6 (2010)
18. Zaharia, M., Konwinski, A., Joseph, A.D., Katz, R., Stoica, I.: Improving MapReduce Performance in Heterogeneous Environments. In: Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation, San Diego, CA, pp. 29–42 (2008)
19. Cha, J.H., Mi, J., Yun, W.Y.: Modelling a General Standby System and Evaluation of its Performance. Applied Stochastic Models in Business and Industry 24(2), 159–169 (2008)
20. Du, Y., Yu, H.: Paratus: Instantaneous Failover via Virtual Machine Replication. In: Proceedings of 8th International Conference on Grid and Cooperative Computing, Lanzhou, Gansu, China, pp. 307–312 (2009)
21. Wang, F., Qiu, J., Yang, J., Dong, B., Li, X., Li, Y.: *Hadoop High Availability through Metadata Replication*. In: Proceedings of the First International Workshop on Cloud Data Management, pp. 37–44. ACM (2009)
22. Alvaro, P., Condie, T., Conway, N., Elmeleegy, K., Hellerstein, J.M., Sears, R.C.: BOOM: Data-centric Programming in the Datacenter. Technical Report UCB/EECS-2009-113, EECS Department, University of California, Berkeley (July 2009)
23. He, X., Ou, L., Engelmann, C., Chen, X., Scott, S.L.: Symmetric Active/Active Metadata Service for High Availability Parallel File Systems. Journal of Parallel and Distributed Computing 69(12), 961–973 (2009)
24. Chen, Z., Xiong, J., Meng, D.: Replication-based Highly Available Metadata Management for Cluster File Systems. In: Proceedings of the IEEE International Conference on Cluster Computing, Heraklion, Greece, pp. 292–301 (2010)
25. Marozzo, F., Talia, D., Trunfio, P.: A Peer-to-Peer Framework for Supporting MapReduce Applications in Dynamic Cloud Environments. In: Cloud Computing: Principles, 1st edn. Springer (2010)
26. White, T.: Hadoop: The Definitive Guide. O'Reilly Media, Yahoo! Press (June 5, 2009)
27. Chockler, G.V., Keidar, I., Vitenberg, R.: Group Communication Specifications: A Comprehensive Study. ACM Computing Surveys 33(4), 427–469 (2001)
28. Défago, X., Schiper, A., Urbán, P.: Total Order Broadcast and Multicast Algorithms: Taxonomy and Survey. ACM Computing Surveys 36(4), 372–421 (2004)
29. Issariyakul, T., Hossain, E.: Introduction to Network Simulator NS2. Springer Science Media (2009) ISBN: 978-0-387-71759-3

Analysis and Visualization of Large-Scale Time Series Network Data

Patricia Morreale, Allan Goncalves, and Carlos Silva

Department of Computer Science, Kean University, Union, NJ USA
{pmorreal,goncalal,salvadca}@kean.edu

Abstract. Large amounts of data ("big data") are readily available and collected daily by global networks worldwide. However, much of the real-time utility of this data is not realized, as data analysis tools for very large datasets, particularly time series data are cumbersome. A methodology for data cleaning and preparation needed to support big data analysis is presented, along with a comparative examination of three widely available data mining tools. This methodology and offered tools are used for analysis of a large-scale time series dataset of environmental data. The case study of environmental data analysis is presented as visualization, providing future direction for data mining on massive data sets gathered from global networks, and an illustration of the use of big data technology for predictive data modeling and assessment.

1 Introduction

Increasingly large data sets are resulting from global data networks. For example, the United States government's National Oceanic and Atmospheric Administration (NOAA) compiles daily readings of weather conditions from monitoring stations located around the world. These records are freely available. While such a large amount of data is readily available, the value of the data is not always evident. In this chapter, large time series data was mined and analyzed using data mining algorithms to find patterns. In this specific instance, the patterns identified could result in better weather predictions in the future.

Building on earlier work [1,2,3,4], two separate datasets from NOAA were used. One was the Global Summary of Day (GSOD) dataset [5], which currently has data from 29,620 stations. The second was the Global Historical Climatology Network (GHCN) dataset [6], which currently has data from 77,468 stations. In previous projects, the datasets were downloaded from NOAA's FTP servers and made locally available on our local database server. Each of the GSOD stations collect 10 different types of data (precipitation, snow depth, wind speeds, etc.), while the GHCN stations can collect over 80 different types of data, although the majority only collect 3 or 4 types. Some stations have been collecting data for over 100 years. Both datasets combined consist of over 2.62 billion rows (records) in our database.

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_9

Data from both datasets was mined using Weka, RapidMiner, and Orange, which are free data mining programs. Each of these programs has a variety of data mining algorithms which were applied to our data. However, before mining, the data had to be converted into a format which allowed it to be properly mined. The problem was solved by developing custom Java programs to rearrange the data.

In this illustrated example, the goal was to find any patterns existing in the data, help predict future significant weather events (snowstorm, hurricane, etc.), and visualize results in a meaningful format. It is also hoped that this work mining large-scale datasets will help others do the same with any dataset of similar magnitude. Although global data was available, the portion of the dataset used was New Jersey, as it has its own special microclimate. By using data from some of New Jersey's extreme weather events as starting points, this research was able to look for patterns that may assist in predicting such events in the future. Examples of extreme events in New Jersey include the unpredicted snowstorm of October 2011, the December 26, 2010 snowstorm (24"-30" accumulation), and a tornado Supercell that hit the state in August 2008.

2 Big Data Applications

The objective was to use the very large amount of data previously imported into a local database server and run data mining algorithms against it to find patterns. This approach was similar to one taken to find relationships in medical data from patients with diabetes [7]. Fields such as telemedicine and environmental sustainability offer great opportunities for big data analysis and visualization. For environmental big data analysis, a variety of popular data mining software was used to evaluate the data to see if one product provided superior results. The software products used were Weka [8], RapidMiner [9], and Orange [10]. Additionally, the raw data was graphed [11, 12] to see if any patterns were identified through visual inspection which the mining software might overlook. By mining the data, a trend was expected in environment/weather. Possible results could be evidence of global warming, colder winters, warmer summers (heat waves), stronger/weaker storms, or more/less large storms. This chapter is an extension of [1, 2], with a specific application of environmental sustainability.

2.1 Environmental Sustainability

Environmental sustainability encompasses several stages. Initially, environmental sustainability referred to development that minimized environmental impact. However, in established areas, such as urban communities, environmental sustainability includes detection of potential problems, monitoring the impact of potential or actual problems, and working to reduce adverse impact of identified threats to environmental sustainability.

The repetitive nature of threats to environmental sustainability in urban environments, such as the underpass that consistently floods during heavy rains, or the air quality that predictably degrades over the course of a workday, is the stuff of urban

legend. Neighborhood residents and regular visitors to the area may generally know the hazards of a particular urban spot, but sharing the knowledge of destructive or hazardous patterns with organizations which might be able to remediate or prevent such regular environmental degradation is not easily done. Rather, at each instance of an environmental threat, the flooded underpass or poor air is addressed as a public safety crisis and personnel and resources are deployed in an emergency manner to provide appropriate traffic rerouting or medical attention.

In addition to critical events which threaten urban environmental conditions, more insidious, slowly evolving circumstances which may result in future urban crises are not monitored. For example, traffic volume on highly used intersections or bridges is not monitored for increasing noise or volume, which might result in a corresponding increase in hazardous emissions or stress fractures. When urban threats are identified, the response process can be aggravated as traffic comes to a standstill or is rerouted, hindering, or delaying emergency response personnel.

Both immediate and slower threats to urban environmental sustainability are dealt with on an 'as occurring' basis, with no anticipation or preventive action taken to avert or decrease the impact of the urban threat. The result, over the past years, has been an increase in urban crisis management, rather than an increased understanding of how our urban environments could be better managed for best use of all our resources – including resources for public safety and environmental sustainability.

The increasing age of urban infrastructures, and the further awareness of environmental hazards in our midst highlights that the management of environmental threats on an 'as needed' basis is no longer feasible, particularly as the cost of managing an environmental crisis can exceed the cost of preventing an environmental crisis. With the potential to gather data in a real-time manner from urban sites, the opportunity for anticipatory preparation and preventive action prior to urban environmental events has become possible. Specifically, street level mapping, using real-time information, is now possible, with the integration of new tools and technology, such as geographic information systems and sensors.

Environmental sustainability in an urban environment is challenging. While numerous measures of environmental sustainability, including air quality, rainfall, and temperature, are possible, the group assessment of these measured parameters is not as easily done. Air quality alone is composed of a variety of measurements, such as airborne particulate matter (PM10), nitrogen dioxide (NO2), Ozone (O3), carbon monoxide (CO) and carbon dioxide (CO2). Road traffic is the main cause of NO2 and CO. While simple environmental solutions such as timing traffic lights are identified as saving billions in fuel consumption and reducing air-pollution (i.e., improving air quality) by as much as 20%, the technological underpinnings to accomplish this have not been developed and deployed on an appropriate scale for urban data gathering and correlation. Furthermore, the use of predictive models and tools, such as data mining, to identify patterns in support or opposed to environmental sustainability is not commonly done in an urban setting. While hurricane, earthquake, and other extreme weather events occur and the aftermath is dramatically presented, more mundane but not less impacting events such as urban flash flooding, chemical spills on city roads, and other environmental events are not anticipated or measured while occurring or

developing, with intent to reduce in scope and damage. By gathering data locally, for assessment and prediction, areas, and events can be identified that might harm environmental sustainability. This knowledge can be used to avoid or disable what might have previously been an urban environmental disaster.

2.2 Data Mining for Trend Identification

Data mining [13] can be referred to as 'knowledge discovery in databases', and is a key element of the "big data" analysis project. Of the four core data mining tasks:

- Cluster analysis
- Predictive modeling
- Anomaly detection
- Association analysis

both predictive modeling and anomaly detection are used in the big data analysis project detailed here. "Predictive modeling" can be further defined by two types of tasks: classification, used for discrete target variables, and regression, which is used for continuous target variables.

Forecasting the future value of a variable, such as would be done in a model of an urban ecosystem, is a regression task of predictive modeling, as the values being measured and forecast are continuous-valued attributes. In both tasks of predictive modeling, the goal is to develop a model which minimizes the error between predicted and true values of the target variables. By doing so, the objective is to identify crucial thresholds that can be monitored and assessed in real-time so that any action or alert may be automatic and high responsive.

"Anomaly detection" is also crucial to the success of big data modeling. Formally stated anomaly detection is the task of identifying events or measured characteristics which are different from the rest of the data or the expected measurement. These anomalies are often the source of the understanding of rare or infrequent events. However, not all anomalies are critical events, meriting escalation, and further investigation. A good anomaly detection mechanism must be able to detect non-normal events or measurements, and then validate such events as being outside of expectations – a high detection rate and low false alarm rate is desired, as these define the critical success rate of the application.

2.3 Sensor Networks and Visualization

Sensor networks have become part of our everyday lives and attract wide interest from industry due to their potential diversity of applications, with a strong expectation that outdoor and environmental uses will dominate the application space [14]. However, actual deployment experience is limited and application development has been further restricted [15]. Previous work in sensors for structure monitoring [16], urban flash flood awareness [17], and mobile emissions monitoring [18] has been initiated, but not to the extent and geographical contextual presentation outlined here.

The overall objective of the big data environmental network implementation is the gathering of environmental information in real-time and storing the data in a database so that, the data can be visually presented in a geographic context for maximum understanding. Ideally, the big data environmental network is an implementation of a wireless environmental sensing network for urban ecosystem monitoring and environmental sustainability. By measuring environmental factors and storing the data for comparison with future data gathered, the changes in data measured over time can be assessed. Furthermore, if a change in one measured variable is detected, examination of another measured variable may be needed to correlate the information, and determine if the measured conditions are declining or advancing over time. Known as 'exception mining,' this assessment can also be visually presented in a geographical context, for appropriate understanding and preventive or divertive action.

2.4 Network Application Design

The network design used was that of a system of distributed sensors, reporting to a base station, which was then connected to a server between the network and the application. The focus was on total application design, from data storage in the relational database, to final interpretation, and presentation in the geographical context.

A. Visual Presentation

Application development included a collection of data, which was archived into a database. This was accomplished by using SQL, a relational database system. To clearly understand and visualize the importance, functionality, and advantages provided by the wireless sensor network, the data must be clearly represented. In order to do so, a programming language or framework is needed that provides the ability to quickly gather data and accurately represent each of our sensors.

After consideration of availability, scalability, and recognition, as well as understanding the well-defined API and number of tutorials available, the Google Maps framework was selected. The Google Maps API allows the user to use Google Maps on individual websites, with JavaScript. In additional, a number of different utilities are available.

Google Maps provides an additional advantage, as the nodes represented on the map and the data contained in each one of the nodes can be setup using XML and additional nodes or markers can be added with ease. Once the decision to use Google Maps was made, the software development effort shifted from data representation efforts to working on accessing the actual real-time data from the wireless sensor network server. For Google Maps to process this new data and properly represent it, an XML file is generated. Fig. 1 shows the exchange between the web server and the wireless sensor network server.

As data is sent in from the nodes, it is passed through the base station to the Perl XML Parser, which parses the incoming data and filters out unwanted packets. The result remaining is the desired data packet set.

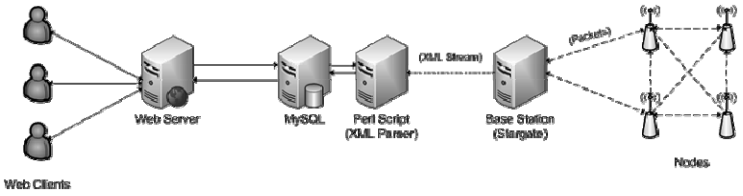


Fig. 1. Data exchange between web server and wireless sensor network server

The wireless sensor network data is gathered in a database, which is then presented in a visual context. Google Maps can be used to present the information in a geographical context. By clicking on the sensor node, real-time information is presented to the viewer, in appropriate context. The raw data, collected from the sensor, is appropriately converted to standard units for display and understanding.

In addition to real-time data presentation in a geographical context, a temporal presentation, using time and date information, has also been developed. An illustration of this can be seen in Fig. 2. A query by sensor presents the specific details of that sensor at one point in time, as well as providing a comparison of the sensor's status for prior dates and times. More than one sensor measurement can be overlaid on the chart, which permits correlation of events and times precisely with sensors.

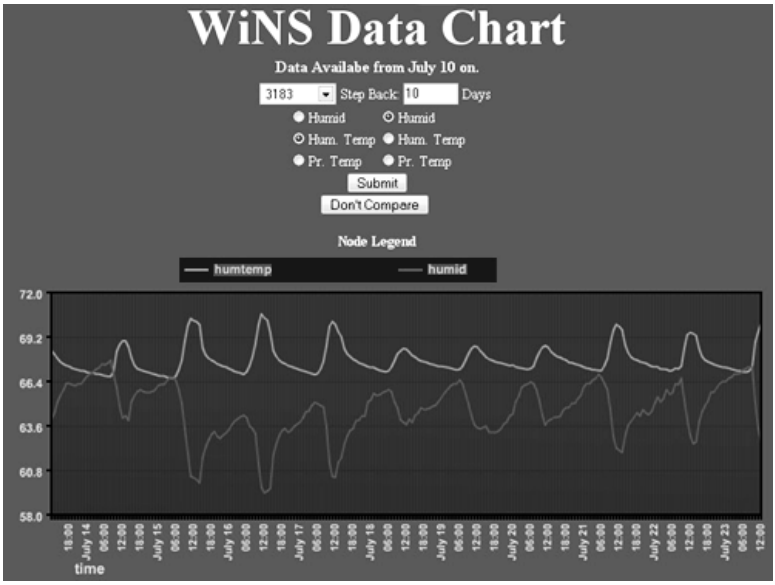


Fig. 2. Visual presentation of data from two sensors with date

While data has been gathered by sensors before, the correlation and presentation of this environmental real-time information in a geographical context, mined from a very large dataset, in addition to providing support for historical and temporal comparison, is innovative.

B. Integration of Geovisualization and Data Mining

The presentation of data in real-time for contextual understanding is only one aspect of the big data application. The archived information in the database permits before and after animations to be developed, using time gradients to show how the measured variables have changed in the preceding time, or are expected to change in the future, based on predictive algorithms, taking into account the reported variables, such as wind speed, in the case of a chemical spill and the potential migration over an urban community, for example.

Once the data of the multiple network nodes has been collected for a long period of time, it is desirable to reveal patterns, if any, in the data. For example, with regard to spatial-temporal variations of air quality along the streets or at intersections, patterns regarding daily and seasonal cycles of air quality, change, and decay over distances from the intersection can be directly visualized through an interactive and animated visualization environment. Furthermore, the visualization tool can be used together with numerical data mining algorithms to model quantitative relationships between air quality and other factors such as weather and traffic conditions. Numerical data mining algorithms such as supervised learning can be easily integrated here. The findings can then be applied to simulate the spatial-temporal air quality variations given arbitrary weather and traffic conditions for the purpose of predictive modeling.

Human visual perception offers a broadband channel for information flow and excellent pattern recognition capabilities that facilitate knowledge discover and the detection of spatial-temporal relations [19]. An effective tool to explore geographic data and communicate geographic information to private or public audiences, geovisualization has long been used for data exploration and pattern recognition. The approach presented and discussed here integrates geovisualization with data mining to reveal spatial-temporal patterns embedded in the data collected by the sensor network over time. While prior work [20] has approached such an idea, the work presented here is the first to visually present the data correlations.

2.5 Big Data Analysis for Environmental Sustainability

Wireless sensor network applications, and the very large dataset of gathered data associated with them, are an emerging area of technology which will benefit organizations and governments with valuable real-time data. In order to properly use such data, a strong, dynamic, and user-friendly interface is needed which allows individuals to clearly see how measured conditions, such as environmental circumstances, are changing over time. The visual depiction of urban environmental events, for example, will permit anticipatory or preventive actions to be taken in advance of adverse human and ecological impact. The use of data mining techniques on the data gathered by the wireless sensor network permits the identification of past patterns and developing trends in air quality or urban flooding, for example. The network and interface illustrated here accomplishes the goals of real-time information gathering and display for environmental sustainability and further work is underway to improve and refine the solution presented here.

Additional research underway includes a case study where a number of exploratory spatial data analysis (ESDA) techniques will be tested to facilitate the visual detection of spatial-temporal patterns of air quality in relation to weather conditions. ESDA techniques being tested in the case study include temporal brushing [21] and temporal focusing [22], temporal reexpression through multiscale data aggregation [23] and static visual bench marking [24]. Animation of the temporal data will enable common users to visualize the change of air quality over space and time. With temporal brushing and focusing, the use is not only a passive viewer of the information, but can interact with the animation and learn actively. Temporal reexpression through multiscale data aggregation provides an opportunity to directly visualize the daily and seasonal cycles of air quality change. Finally, using static visual benchmarking, the air quality level at any recorded time spot can be compared visually with health standards and give the viewer a direct alert on how high air quality is affecting human health. Efforts continue to integrate the geovisualization environment with a knowledge discovery procedure for data mining.

3 Visualization and Pattern Identification in Big Data

Visualization of massively large datasets presents two significant problems. First, the dataset must be prepared for visualization, and traditional dataset manipulation methods fail due to lack of temporary storage or memory. The second problem is the presentation of the data in the visual media, particularly real-time visualization of streaming time series data. Visualization of data patterns, particularly 3D visualization, represents one of the most significant emerging areas of research. Particularly for geographic and environmental systems, knowledge discovery and 3D visualization is a highly active area of inquiry. Recent advances in association rule mining for time series data or data streams makes 3D visualization and pattern identification on time series data possible.

In streaming time series data the problem is made more challenging due to the dynamic nature of the data. Emerging algorithms permit the identification of time-series motifs [25] which can be used as part of a real-time data mining visualization application. Geographic and environmental systems frequently use sensor networks or other unmanned reporting stations to gather large volumes of data which are archived in very large databases [26]. Not all the data gathered is important or significant. However the sheer volume of data often clouds and obscures critical data which causes it to be ignored or missed.

The research presented here outlines an ongoing research project working to visualize the data from national repositories in two very large datasets. Problems encountered include dataset navigation, including storage and searching, data preparation for visualization, and presentation.

Data filtering and analysis are critical tasks in the process of identifying and visualizing the knowledge contained in large datasets, which is needed for informed decision making. This research is developing approaches for time series data which will permit pattern identification and 3D visualization. Research outcomes include as-

sessment of data mining techniques for streaming time series data, as well as interpretive algorithms, and visualization methods which will permit relevant information to be extracted and understood quickly and appropriately.

3.1 Large-Scale Data for Visualization

This research works with datasets from the National Oceanic and Atmospheric Administration (NOAA), a federal agency in the U.S., focused on the condition of the oceans and the atmosphere. The purpose of the research project is to take meteorological data and analyze it to identify patterns that could help to predict future weather events. Data from the GHCN (Global Historical Climatology Network) dataset [3] was initially used. Earlier research had worked with NOAA's Integrated Surface Dataset (ISD) [4]. Both of these datasets are open access and the volume of streaming time series data was significant and growing.

The GHCN dataset consists of meteorological data from over 76,000 stations worldwide with over 50 different searchable element types. Examples of element types include minimum and maximum temperature, precipitation amounts, cloudiness levels, and 24-hour wind movement. Each station collects data on different element types.

3.2 Time Series Data Analysis

Searching for temporal association rules in time series databases is important for discovering relationships between various attributes contained in the database and time. Association rules mining provides information in an "if-then" format. Because time series data is being analyzed for this research, time lags are included to find more interesting relationships within the data. A software package from Universidad de La Rioja's EDMANS Groups was used to preprocess and analyze the time series from the NOAA datasets. The software package is called *KDSeries* and was created using *R*, a language for statistical computing.

The *KDSeries* package contains several functions that preprocess the time series data so knowledge discovery becomes easier and more efficient. The first step in preprocessing is filtering. The time series are filtered using a sliding-window filter chosen by the user. The filters included in *KDSeries* are Gaussian, rectangular, maximum, minimum, median, and a filter based on the Fast Fourier Transform. Important minimum and maximum points of the filtered time series are then identified. The optima are used to identify important episodes in the time series. The episodes include increasing, decreasing, horizontal and over, below, or between a user-defined threshold.

After simple and complex episodes are defined, each episode is view as an item to create a transactional database. Another R-based software package, *arules*, makes this possible. *Arules* provide algorithms that seek out items that appear within a window of a width defined by the user. From there, temporal association rules are then extracted from the database.

The first algorithm being used to extract the temporal association rules is the Eclat algorithm. Eclat (Equivalence Class Clustering and Bottom-up Lattice Traversal) is an efficient algorithm that generates frequent item sets in a depth-first manner. Other

algorithms such as Aproiri and FP-growth will then be used to extract association rules and compared and contrasted with each other. This work is ongoing.

3.3 Methodology

The data used is located on NOAA's FTP site in the form of .dly files. Each station has its own .dly file which is updated daily (if the station still collects data). Each .dly file has all the data that has ever been collected for that station. Whenever new data is added for a station, it is appended to the end of the current .dly file, which presents the problem that each file must be downloaded over again to keep our local database current. To obtain the data, a Java-based program was built that would download every file in the folder holding the .dly files. The Java program used the Apache Commons Net library to download the files from the NOAA FTP server.

After downloading all of the .dly files, the Java program opens an input stream to each of the downloaded files (one at a time). Each line of the .dly files contains a separate data record, so the Java program would read in each line and use it to form a MySQL "INSERT" statement that would be used to place data into a local database. At one point, space was exhausted on the local machine, and researchers had to upgrade the hard disk from ~200 GB to 256 GB to continue inserting data into the relational database.

Once all of the data was placed into the local database, a web interface was built that allowed users to search the dataset (Fig. 3). The interface allows users to search by country, state (within the United States), date range, and values that are <, <=, >, >=, !=, or == to any chosen value. Because the dataset contains the value -9999 for any record that is invalid or was not collected, the web interface also has the option to exclude any -9999 values from the results. The results are output with each line containing a different data result, and each result consisting of month, day, year, and data value.

Fig. 3. Query screen of web interface to NOAA GHCD dataset

For visualizing the data, the first step was to plot the location of each station using Google Earth. The NOAA FTP server also has a file that lists the longitude, latitude, and elevation of each station, so this information was placed into a separate table in our database. Next, a PHP script from the Google Earth website was customized to support queries to the database for the location of each station and then format the results in KML. This KML data is then loaded into the browser-based version of Google Earth on the same webpage. A separate PHP script was built that allows the user to search for stations in the entire world, by country, or by state (if searching within the US) (Fig. 4) and results from the query are graphed (Fig. 5).

Fig. 4. Web interface to station selection from GHCD dataset

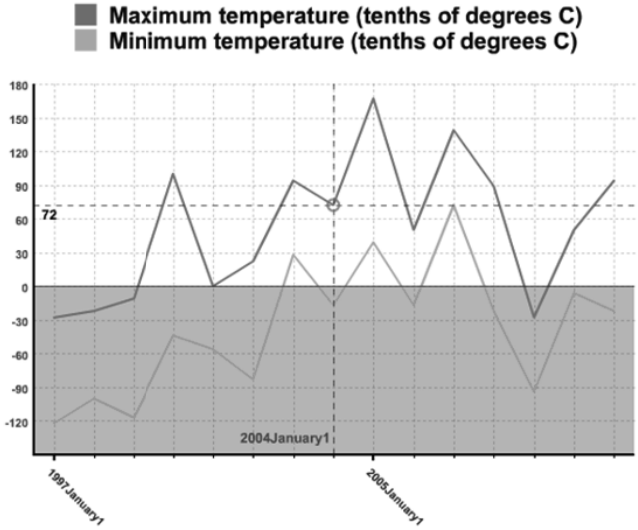


Fig. 5. Real-time visualization of a user query to the GHCD dataset

This visualization can be integrated into a Google Earth display (Fig. 6).

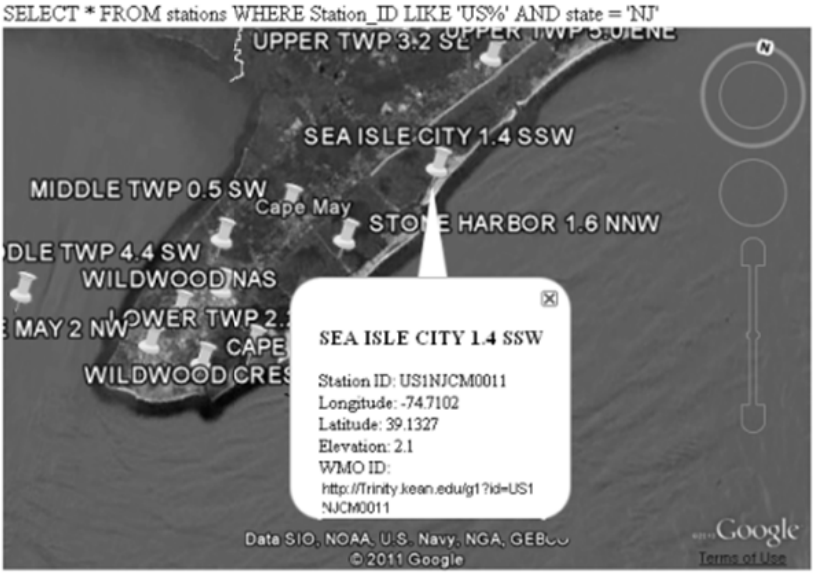


Fig. 6. NOAA GHCD reporting stations in Google Earth

4 Preparing Big Data for Analysis

Data type dissimilarities in big data are common. Significant time can be spent on data cleaning and validation. The effort presented here resulted in a methodology to pull data from the database and convert it to a data mining friendly format, with conservative use of computer memory and storage.

The two NOAA datasets were stored in separate tables. For the GSOD dataset, each station always collects the same 10 types of weather data (wind speed, precipitation, etc.), each type of data was given its own column in a table within the database. Unfortunately, the GHCN stations can collect any of over 80 different types of weather data, so it would be impractical to assign each data type its own column. For this reason, one column was used to store the type of data the record would hold, and a second column to store the actual value collected.

After populating the database, it was learned that the GHCN data needed to be reorganized so that it could be mined. In the GSOD dataset, all types of data are stored in the same record, which makes it very easy for mining software to compare all aspects of one day to another day. In contrast, the GHCN dataset has multiple records for each day to accommodate for its wide variety of data types, which we cannot use with data mining algorithms. This is primarily because the GHCN data types are stored as different measurements and cannot be directly compared to each other. For example, snowfall is measured in millimeters, while maximum temperature is measured in tenths of Celsius degrees.

It would be a massive undertaking to remake the GHCN table within the local database (it would also be quite inefficient if we made a column for each data type), so a Java program was developed to pull data from the database and convert it to a

data mining friendly format. The program queries for data within a specific date range and station range (numbers were assigned to the stations), and for the specific data types which we wanted to retrieve. The program then compiles the results into a file, in which each station has a single record for each day. The file can be read like a table in which each data type has its own column. This lets data be efficiently retrieved from the database, while still allowing the data to be properly organized for mining.

A subset of the data is requested at one time as the conversion process can take hours or days if too much data is requested. Most mining programs cannot handle such large amounts of data, and not all element types are commonly used. Some mining algorithms will not run if a data type is missing too many entries, which would be the case if some of the less commonly collected data types were used. This program is used to retrieve data from both the GSOD and GHCN datasets to reduce time wasted on retrieving the same data multiple times, and to remove the need to have mining programs connect to our database server.

A second Java program was written that converts commas to tabs. The mining program Orange does not read CSV files, but does read tab-delimited files. Originally, commas were used to separate data values, so a second program was made that would quickly convert all commas to tabs in order to analyze the data with Orange.

5 RapidMiner vs. Weka vs. Orange

Three programs were used to mine the data. The most success came from RapidMiner, with quite a bit less success experienced with Weka and Orange. All the three programs have a similar setup in which different functions are dragged and dropped onto the interface screens and then connected together to run the chosen algorithms. Each of the programs gave operational issues at times, but RapidMiner seemed to be the most stable and useful.

Initially, mining began with Weka, but a few key issues made it clear that Weka should be dropped early on. First, Weka's "Knowledge Flow" program, which contains mining functions, was not able to connect to our local database server (before we built our conversion program). Second, Weka has a lot of mining algorithms, but little explanation of how to use them. Frustrated with these problems, RapidMiner was tried.

RapidMiner has a large amount of mining functions, and it has an extension that gives it some functions from Weka. It also has a file import wizard that helps ensure that data is correctly imported into the program. Most importantly, there is a small guide for each mining algorithm on the bottom right-hand corner of the screen that is automatically shown whenever a function is selected. The guide explains exactly what a function is for, how to use it, and what its inputs and outputs are. RapidMiner also has a search feature that lets users quickly find the algorithms they want to use. In addition, it has a wizard called "Automatic System Construction" that runs different algorithms on data to determine which ones may yield results [11]. We did not have much success with this feature. Out of all the programs, only RapidMiner provided results.

Despite those positive features, there were two main problems with RapidMiner. The first problem was that it would crash if functions with more than ~5MB of input

data were used. The second problem was that many functions would only run if numeric data were converted to nominal data. This means that those numeric values would be treated as though they were words, thus entirely removing their important numeric properties. The nominal values have no relation to each other, such as difference (between two values), but are treated as separate, equal instances.

Orange was the last program we tried using. Before using Orange the CSV files were converted to tab-delimited files, but this was not really an issue. A feature that stands out in Orange is that Orange will only permit the addition of a new function if it can be attached to one that has already been chosen. This removes some guesswork, allows us to easily see what we can use, and identify functions that we may have otherwise overlooked. Like RapidMiner, Orange would sometimes crash if it received too much data.

6 Mining the Data

In an effort to find patterns, a variety of algorithms were used. There are three main algorithms that provided some form of results, and those were the

- association rules algorithms
- various decision tree algorithms
- Naïve Bayes algorithm

All algorithms were run on a computer using a 2.66 GHz Intel Core 2 Duo E8200 processor, 3 GB of RAM, and the 32-bit version of Windows XP. Times listed below will be for algorithms analyzing a 5.39MB file containing 238,839 records from the GHCN dataset.

The purpose of an *association rules algorithm* is to find relationships between different columns of data (data types). An example could be if a day's minimum temperature was above 75˚F then the month is June, July, or August (summer months in New Jersey). Weka had originally given some very simple relationships like this, but nothing of significance. RapidMiner required conversion of most numeric values to nominal values, so when these algorithms were used the rules produced were not helpful. The average time for running an association rules algorithm in RapidMiner on the GHCN file was 8.3 seconds.

The purpose of the *decision tree algorithms* is very similar to the association rules in that it tries to find relationships between different data columns. These relationships are then used to form a tree that leads from one column to another until you arrive at a leaf node. A data column which will be a leaf node in the tree must be selected (assign it as a label), as this will be the value which you are trying to predict. RapidMiner always assembles the decision trees in any arrangement it chooses, so it is not possible to designate the position of specific data columns in the traversal of the tree.

RapidMiner did not produce any significant results with these algorithms. The program produced trees, but they were not very useful, most likely because it again required numeric values to be converted to nominal values. Some of the different decision trees tried included regular decision trees, CHAID (shown in Fig. 7), and ID3. The average time for running a regular decision tree algorithm in RapidMiner on the GHCN file was 12.5 seconds.

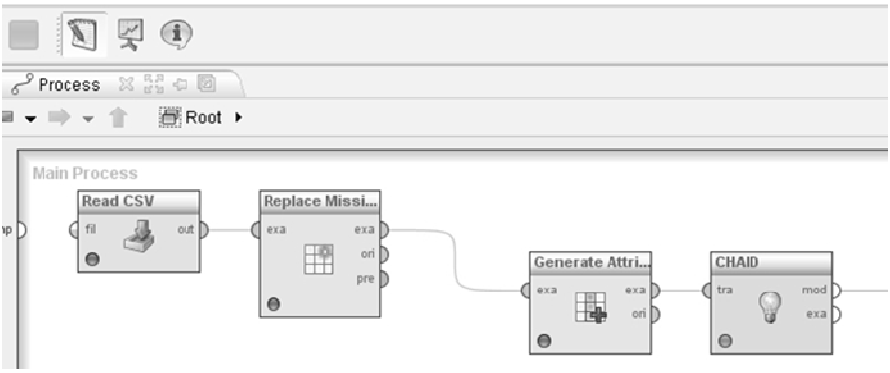


Fig. 7. Connection of functions used with the CHAID decision tree algorithm in RapidMiner

The *Naïve Bayes algorithm* is a classification algorithm used to group temperature ranges. Using RapidMiner's "Generate Attribute" function, we were able to create new data columns that took the minimum and maximum temperature values and grouped them into ranges of very cold, cold, middle, warm, and very warm. The different temperature ranges used were picked by the group. The results are shown in Fig. 8. After running the Naïve Bayes algorithm on this new data, it was determined that the number of very cold days in New Jersey has increased significantly over the past 7 years, jumping from ~2.3% of days in 2005 to ~24.7% of days in 2011. The average time for running the Naïve Bayes algorithm in RapidMiner on the GHCN file was 7.1 seconds.

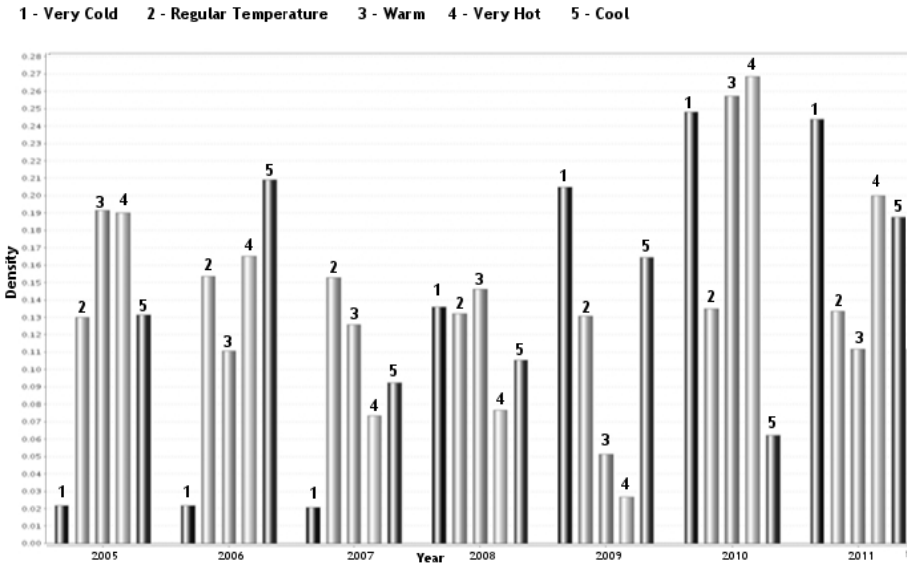


Fig. 8. Graph showing temperature changes in New Jersey over the last 7 years (2005-2012) using data from the GHCN dataset. This graph was generated from the results of the Naïve Bayes algorithm.

7 Visual Inspection of Raw Data

In addition to mining the data using algorithms, raw data was graphed using the program DPlot [12]. Various data values were graphed against each other and against the dates they were recorded. Overall, only one significant pattern resulted from this approach, which was discovered while graphing maximum temperature values for the last 7 years in New Jersey using the GHCN dataset. This graph shows that after every 0.8 or 0.12 Celsius degrees there is a gap of 0.4 Celsius degrees without any recorded values. In addition, the blocks of 0.8 and 0.12 degrees alternate almost continuously along the graph (shown in Fig. 9). This is very odd, and there is no explanation for the missing values.

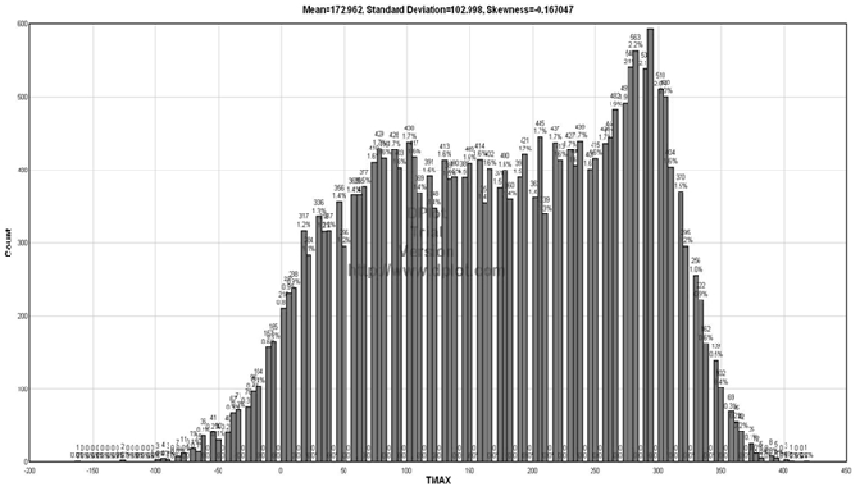


Fig. 9. Histogram showing maximum temperature distributions over the past 7 years in New Jersey (using the GHCN dataset). There is a pattern of missing values shown in the graph.

8 Visualization and Presentation in Context

Overall, the data mining programs did not yield results of great significance on the datasets used thus far. However, the work with the data mining tools yielded more information. There was little new information learned from the algorithms or from the actual results. Most of the association rules were either nonsense due to conversion from numeric to nominal types or very basic rules that are already commonly known (like colder temperatures are seen in the winter). Most decision trees also gave similar results or refused to give anything at all (many times the results were a tree with a single node). Some of this may also be due to the fact that there were only a few commonly collected data types (precipitation, minimum/maximum temperature) in both datasets. To make matters worse, many data types in the GHCN dataset may

have been similar (value of 0) or missing too many values to form accurate associations or predictions. To obtain results of significance or find previously unknown patterns there are two things needed:

1. many more data types that are continuously collected by all weather stations and
2. mining algorithms that support a greater range of numeric data types.

9 Conclusions

Although no new patterns were identified by this project, the greater benefit was learning how to organize data for mining and how to mine large amounts of data. The approach taken here for data mining is valid, however:

1. stronger connections between the data types were needed for pattern identification,
2. the mining programs did not handle the data properly, and
3. a greater range of data types is needed to obtain more significant results.

The methodology outlined here can be applied to a wide range of other fields, as long as a dataset with a large number of continuously collected data types is used. The health industry in particular may benefit from mining patient data to find hidden links between different patients with the same diseases or illnesses. Such medical data is almost continuously being collected in large amounts.

Reorganizing the data for use by mining algorithms is an important part of the process. Writing a program to retrieve the data from the NOAA database and saving it locally in a different format was not difficult, but the conversion process sometimes took a very long time. The bulk of this time was taken to retrieve the files from the database. To reduce such time, researchers can either build the database in the needed format to begin with (not always possible), or convert the data from a file stored on the same machine. The process to request and fetch the data from the database is the most time consuming portion of the large dataset analysis. Storing the converted data in a local file will make sure that the request-and-fetch process only has to be performed once for each set of test data.

As for the different programs used, RapidMiner was far easier to use than Orange and Weka. RapidMiner had its drawbacks with data handling and occasional crashing, but it was simply easier to use thanks to its search feature, its documentation for every function, and the fact that it gave results. Overall, Weka and Orange were lacking as programs, as they were troublesome and not useful.

The three main algorithms that were used (association rules, Naïve Bayes, and decision trees) all took approximately the same amount of time to run against a very large amount of data. To remedy most of the crashes experienced, a computer upgrade to a 64-bit operating system, more RAM, and more/faster processors, is planned. Even without a more capable system, this problem was somewhat overcome by splitting the files into subsets that were mined separately.

Future plans for this research include additional comparative experience with larger datasets, involved data from other areas and case studies from other disciplines.

References

1. Holtz, S., Valle, G., Howard, J., Morreale, P.: Visualization and Pattern Identification in Large Scale Time Series Data. In: IEEE Symposium on Large Scale Data Analysis and Visualization (LDAV 2011), Providence, RI, pp. 17–18 (2011)
2. Morreale, P., Qi, F., Croft, P.: A Green Wireless Sensor Network for Environmental Monitoring and Risk Identification. International Journal on Sensor Networks 10(1/2), 73–82 (2011)
3. Shyu, C., Klaric, M., Scott, G., Mahamaneerat, W.: Knowledge Discovery by Mining Association Rules and Temporal-Spatial Information from Large-Scale Geospatial Image Databases. In: Proceedings of the IEEE International Symposium on Geoscience and Remote Sensing (IGARSS 2006), pp. 17–20 (2006)
4. Zhu, C., Zhang, X., Sun, J., Huang, B.: Algorithm for Mining Sequential Pattern in Time Series Data. In: Proceedings of the IEEE 2009 WRI International Conference on Communications and Mobile Computing, pp. 258–262 (2009)
5. NOAA Integrated Surface Database (GSOD), `http://www.ncdc.noaa.gov/oa/climate/isd/index.php` (retrieved June 12, 2013)
6. NOAA Global Historical Climatology Network (GHCN) Database, `http://www.ncdc.noaa.gov/oa/climate/ghcn-daily/` (retrieved June 12, 2013)
7. Han, J., Rodriguez, J.C., Beheshti, M.: Diabetes Data Analysis and Prediction Model Discovery Using RapidMiner. In: IEEE Proceedings of the 2nd International Conference on Future Generation Communication and Networking (FGCN 2008), pp. 96–99 (2008)
8. Weka's website, `http://www.cs.waikato.ac.nz/ml/weka/` (retrieved June 12, 2013)
9. RapidMiner's website, `http://rapid-i.com/content/view/181/190/` (retrieved June 12, 2013)
10. Orange's website, `http://orange.biolab.si/` (retrieved June 12, 2013)
11. Shafait, F., Reif, M., Kofler, C., Breuel, T.R.: Pattern Recognition Engineering. In: RapidMiner Community Meeting and Conference (RMiner 2010), Dortmund, Germany (2010)
12. DPlot's website, `http://www.dplot.com/` (retrieved June 12, 2013)
13. Thuraisingham, B., Khan, L., Clifton, C., Maurer, J., Ceruti, M.: Dependable Real-time Data Mining. In: Proceedings of the 8th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC 2005), pp. 158–165 (2005)
14. Martinez, K., Hart, J.K., Ong, R.: Environmental Sensor Networks. IEEE Computer, 50–56 (August 2004)
15. Lewis, F.L.: Wireless Sensor Networks. In: Cooke, D.J., Das, S.K. (eds.) Smart Environments: Technologies, Protocols, and Applications. John Wiley, New York (2004)
16. Zimmerman, A.T., Lynch, J.P.: Data Driven Model Updating using Wireless Sensor Networks. In: Proceedings of the 3rd Annual ANCRiSST Workshop (2006)
17. Chang, N., Guo, D.: Urban Flash Flood Monitoring, Mapping, and Forecasting via a Tailored Sensor Network System. In: Proceedings of the 2006 IEEE International Conference on Networking, Sensing and Control, pp. 757–761 (2006)
18. Cordova-Lopez, L.E., Mason, A., Cullen, J.D., Shaw, A., Al-Shamma'a, A.I.: Online vehicle and atmospheric pollution monitoring using GIA and wireless sensor networks. Journal of Physics: Conference Series 76(1) (2007)

19. Gahegan, M., Wachowicz, M., Harrower, M., Rhyne, T.-M.: The Integration of geographic visualization with knowledge discovery in databases and geocomputation. Cartography and Geographic Information Science 28(1), 29–44 (2001)
20. Arici, T., Akgu, T., Altunbasak, Y.: A Prediction Error-Based Hypothesis Testing Method for Sensor Data Acquisition. ACM Transactions on Sensor Networks 2(4), 529–556 (2006)
21. Monmonier, M.: Geographic brushing: Enhancing exploratory analysis of the scatter plot matrix. Geographical Analysis 21(1), 81–84 (1989)
22. MacEachren, A.M., Polsky, C., Haug, D., Brown, D., Boscoe, F., Beedasy, J., Pickle, L., Marrara, M.: Visualizing spatial relationships among health, environmental, and demographic statistics: interface design issues. In: 18th International Cartographic Conference Stockholm, pp. 880–887 (1997)
23. Monmonier, M.: Strategies for the visualization of geographic time-series data. Cartographica 27(1), 30–45 (1990)
24. Harrower, M.: Visual Benchmarks: Representing Geographic Change with Map Animation. Ph.D. dissertation, Pennsylvania State University (2002)
25. Mueen, A., Keogh, E.: Online Discovery and Maintenance of Time Series Motifs. In: Proceedings of 16th ACM Conference on Knowledge Discovery and Data Mining (KDD 2010), pp. 1089–1098 (2010)
26. Morreale, P., Qi, F., Croft, P., Suleski, R., Sinnicke, B., Kendall, F.: Real-Time Environmental Monitoring and Notification for Public Safety. IEEE Multimedia 17(2), 4–11 (2010)

Parallel Coordinates Version of Time-Tunnel (PCTT) and Its Combinatorial Use for Macro to Micro Level Visual Analytics of Multidimensional Data

Yoshihiro Okada

ICER, Kyushu University Library, Kyushu University
744, Motooka, Nishi-ku, Fukuoka, 819-0395 Japan
okada@inf.kyushu-u.ac.jp

Abstract. This chapter treats an interactive visual analysis tool called PCTT, Parallel Coordinates Version of Time-tunnel, for multidimensional data and multi-attributes data. Especially, in this chapter, the author introduces the combinatorial use of PCTT and 2Dto2D visualization functionality for visual analytics of network data. 2Dto2D visualization functionality displays multiple lines those represent four-dimensional (four attributes) data drawn from one (2D, two attributes) plane to the other (2D, two attributes) plane in a 3D space. Network attacks like the intrusion have a certain access pattern strongly related to the four attributes of IP packet data, i.e., source IP, destination IP, source Port, and destination Port. So, 2Dto2D visualization is useful for detecting such access patterns. Although it is possible to investigate access patterns of network attacks at the attributes level of IP packets using 2Dto2D visualization functionality, statistical analysis is also necessary to find out suspicious periods of time that seem to be attacked. This is regarded as the macro level visual analytics and the former is regarded as the micro level visual analytics. In this chapter, the author also introduces such combinatorial use of PCTT for macro level to micro level visual analytics of network data as an example of multidimensional data. Furthermore, the author introduces other visual analytics example about sensor data to clarify the usefulness of PCTT.

Keywords: 3D visualization, Parallel Coordinates, Time-tunnel, Intrusion detection.

1 Introduction

This chapter treats an interactive visual analysis tool for multidimensional and multi-attributes data called PCTT, Parallel Coordinates Version of Time-tunnel (PCTT) [1-3]. Originally, Time-tunnel [1, 2] visualizes any number of multidimensional data records as individual charts in a virtual 3D space. Each chart is displayed on a rectangular plane and the user easily puts more than one different planes overlapped together to compare their data represented as charts in order to recognize the similarity or the difference among them. Simultaneously, a radar chart among those data on any attribute is displayed in the same 3D space to recognize the similarity and the

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_10

correlation among them. In this way, the user can visually analyze multiple multidimensional data through interactive manipulations on a computer screen. However, in Time-tunnel, only one chart is displayed on one rectangular plane. So, if there are a huge number of data records, the user has to prepare accordingly such a huge number of rectangular planes and practically it becomes impossible to interactively manipulate them. To deal with this problem, we enhanced the functionality of Time-tunnel to enable it to display multiple chars like Parallel Coordinates [4] on each rectangular plane. This is called Parallel Coordinates version of Time-tunnel (PCTT) [3]. With this enhanced functionality, the user can visually analyze a huge number of multidimensional data records through interactive manipulations on a computer screen. The user can easily recognize the similarity or the difference among those data visually and interactively.

Parallel Coordinates version of Time-tunnel (PCTT) can be used for the visualization of network data because IP packet data have many attributes and such multiple attribute data can be visualized using Parallel Coordinates. Furthermore, we also introduced 2Dto2D visualization functionality to PCTT for intrusion detection of network data. 2Dto2D visualization functionality displays multiple lines those represent four-dimensional (four attributes) data drawn from one (2D, two attributes) plane to the other (2D, two attributes) plane. Using 2Dto2D visualization, it is easy to understand relationships of four attributes of each data. Network attacks have a certain access pattern strongly related to the four attributes of IP packet data, i.e., source IP, destination IP, source Port, and destination Port. So, 2Dto2D visualization is useful for detecting such access patterns. In this chapter, we show several network-attack patterns visualized using PCTT with 2Dto2D visualization.

Using PCTT with 2Dto2D visualization functionality, it is possible to investigate access patterns of network attacks at the attributes level of IP packets. However, statistical analysis is also necessary to find out suspicious periods of time that seem to be attacked. This is regarded as the macro level visual analytics and the former is regarded as the micro level visual analytics. In this chapter, we introduce such combinatorial use of PCTT for macro level to micro level visual analytics of network data as an example of multidimensional data. We also introduce other visual analytics example to clarify the usefulness of PCTT about sensor data related to our cyber physical systems research project.

The remainder of this chapter is organized as follows. First of all, Section 2 describes related work and points out the difference of our tool from the others. Next, we explain essential mechanisms of IntelligentBox [5] in Section 3 because IntelligentBox is a constructive visual software development system for 3D graphics applications and Time-tunnel is developed as one of its applications. Therefore, Time-tunnel can be combined with other Time-tunnel or other visualization tools. Section 4 describes details of Time-tunnel and its Parallel Coordinates version. And then, Section 5 presents actual network data analysis using PCTT with 2Dto2D visualization as examples of the micro level visual analytics. We also introduce examples of the macro level visual analytics and the combinatorial use of several PCTT in Section 6. In Section 7, we also introduce other visualization examples carried out as the part of our research project about cyber-physical systems. Finally we conclude the chapter in Section 8.

2 Related Work

After the proposal of Parallel Coordinates, many modified versions having a variety of additional features were proposed [6-11]. Our Parallel Coordinates version of Time-tunnel (PCTT) can be used as the same visual analysis tool as original Parallel Coordinates. Furthermore, PCTT visualizes multiple charts like Parallel Coordinates on one individual rectangular plane and it originally provides multiple rectangular planes in a virtual 3D space so that if the user has a huge amount of data records, he/she can analyze them by separating into several groups using multiple rectangular planes to recognize the similarity or the difference among those data visually and interactively. This is one of the advantages of our PCTT. Another popular data analysis method beside Parallel Coordinates is based on star chart or radar chart. As the similar tools, there are Star Glyphs of XmdvTool [12] and Stardinates Tool [13]. Stardinates Tool has combined feature of Parallel Coordinates and Glyphs [12]. There are also researches [14, 15] similar to this. Our PCTT has combinatorial features of Parallel Coordinates and star chart (radar chart) visualization tool with interactive interfaces.

As visualization tools of network data for the intrusion detection, there are many visualization tools [16-30]. The paper [17] proposes several interactive visualization methods for network data and the port scan detection based on PortVis [16], a tool for port-based detection of security events. Most of them are 2D and only volume visualization method uses 3D axes (Port high byte, Port low byte, and Time). The paper [18] proposes a visual querying system for network monitoring and anomaly detection using entropy-based features. The paper [19] proposes ClockView for monitoring large IP spaces, which is a glyph in style of a clock to represent multiple attributes of time-series traffic data in a 2D time table. The paper [20] proposes the use of CLIQUE, a visualization tool of statistical models of expected network flow patterns for individual IP addresses or collections of IP addresses, and Traffic Circle, a standard circle plot tool. As Parallel Coordinates-based visualization tools, there are VisFlowConnect [21] and trellis plots of Parallel Coordinates [22]. As treemap-based visualization tools, there are NAVIGATOR [23], which displays detail information like IP addresses, ports, etc. inside each node of a treemap, and hierarchical visualization [24], which is a 2D map similar to a treemap in a 3D space. Also, there are visualization methods for network data using 3D plots [25] or lines in a 3D space [26-29]. DAEDALUS [29] is a 3D visual monitoring tool of the darknet data. However, there have not been any visualization tools like our PCTT. In this chapter, we also propose 2Dto2D visualization functionality used with PCTT. The concept of 2Dto2D visualization functionality was derived from the visualization tool called nicter Cube [30], and there have not been any visualization tools like our PCTT with 2Dto2D visualization.

3 Essential Mechanisms of IntelligentBox

IntelligentBox is a constructive visual software development system for interactive 3D graphics applications. It provides several components called boxes and it also provides dynamic data-linkage mechanism called 'slot-connection'. In this section, we describe such essential mechanisms of IntelligentBox.

3.1 Model-Display Object (MD) Structure

As shown in Figure 1, each box consists of two objects, a model and a display object. This structure is called MD (Model-Display object) structure. A model holds state values of a box. They are stored in variables called slots. A display object defines how the box appears on a computer screen and defines how the box reacts to user operations. Figure 1 also shows messages between a display object and a model. This is an example of RotationBox. RotationBox has a slot named 'ratio' that holds a double precision number, which means a rotation angle. This value is normalized into zero to one. One means one rotation. Through direct manipulations on a box, its associated slot value changes. Furthermore, its visual image simultaneously changes according to the slot value change. In this way, the box reacts to user's manipulations according to its functionality.

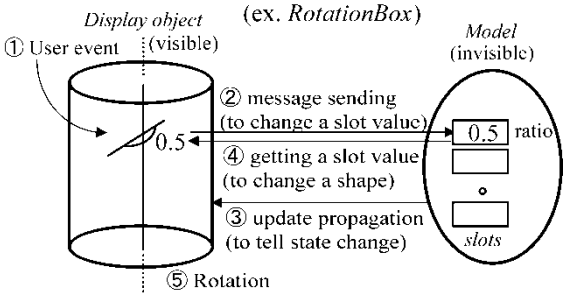


Fig. 1. MD structure of box and its internal messages

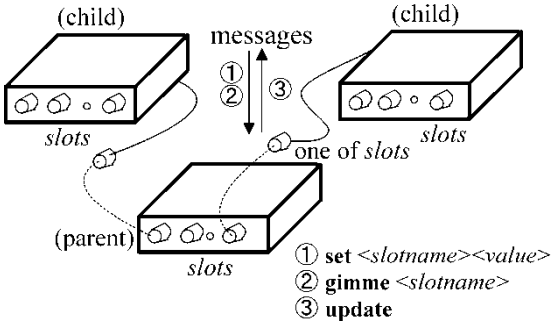


Fig. 2. Standard messages between boxes

3.2 Message-Sending Protocol for Slot-Connections

Figure 2 illustrates data linkage concept among boxes. As shown in the figure, each box has multiple slots. One of its slots can be connected to one of the slots of other box. This connection is called slot-connection. Slot-connection is carried out by several messages when there is a parent-child relationship between two boxes. There are

three standard messages, i.e., a set message, a gimme message and an update message. These messages have the following formats:

(1) Parent box set <slotname> <value>.
(2) Parent box gimme <slotname>.
(3) Child box update.

A <value> in a format (1) represents any value, and a <slotname> in formats (1) and (2) represents a user-selected slot of the parent box that receives these two messages. A set message writes a child box slot value into its parent box slot. A gimme message reads a slot value from a parent box and sets the value into its child box slot. Update messages are issued from a parent box to all of its child boxes to tell them that the parent box slot value has changed.

Each box has three main flags that control the above message flow, i.e., a set flag, a gimme flag, and an update flag. These flags are properties of a display object. A box works as an input device if its set flag is set to true. Contrarily a box works as an output device if its gimme flag is set to true. A box sends update messages if its update flag is set to true. Then, child boxes take an action depending upon the states of the set flag and the gimme flag after they receive an update message or after they individually change their slot values.

4 Time-Tunnel and Its Parallel Coordinates Version

This section describes the system configuration of Time-tunnel, its components, and how Time-tunnel works for the analysis of multidimensional data, especially multiple time- series numerical data.

4.1 System Configuration

Figure 3 shows the component structure of Time-tunnel and Figure 4 shows a screen snapshot of actual Time-tunnel. Time-tunnel consists of three main types of boxes, i.e., data-wing, time-plane, and time-bar.

(1) Data-wing has a shape-like a sheet. It displays one multidimensional data, one time-series numerical data, as a chart on its sheet. For the visualization of multiple data, the user can use multiple data-wings as he/she wants. Each data-wing is connected to time-bar by its hinge. The hinge is also a box that has a rotation functionality called RotationBox. Therefore, by rotation operations on data-wings, the user can put multiple charts overlapped together to compare them as shown in Figure 5. Each multidimensional data, time-series numerical data, of each data-wing is sent to time-bar through RotationBox.
(2) Time-plane also has a shape-like a sheet. Time-plane is connected to time-bar vertically to data-wings. Usually, three time-planes are necessary as shown in Figure 4. Two time-planes are used to specify a time region, i.e., a begin time point and an end time point. As for the visualization of multidimensional data, these time-planes specify a certain set of attributes. As shown in Figure 6, correlation points between any two adjacent charts are displayed inside the time region. The remaining time-plane is used for displaying a radar chart. Figure 6 shows its detail of the radar

chart. Its position data is sent to time-bar to specify a time of data among charts to be displayed as a radar chart. Actually time-plane is connected to time-bar through ExpandBox. Time-plane moves along time-bar by the user manipulations on ExpandBox because ExpandBox is the parent of each time-plane.

(3) Time-bar has a thin, long cylindrical shape. Time-bar works as a time pivot of data-wings. It collects multiple time-series numerical data from each data-wing and displays a radar chart on one of the time-planes. It also displays correlation information between any two adjacent data-wings as scattered points in the time region specified by the two remaining time-planes. Parent-child relationships among data-wings, time-planes, and time-bar are as shown in Figure 3. RotationBox works as the hinge and the parent of data-wing, and time-bar is the parent of each RotationBox. ExpandBox becomes the parent of time-plane, and it works for positioning the time-plane.

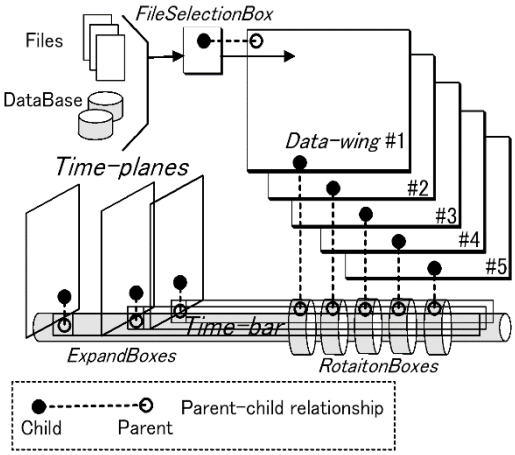


Fig. 3. Component structure of Time-tunnel

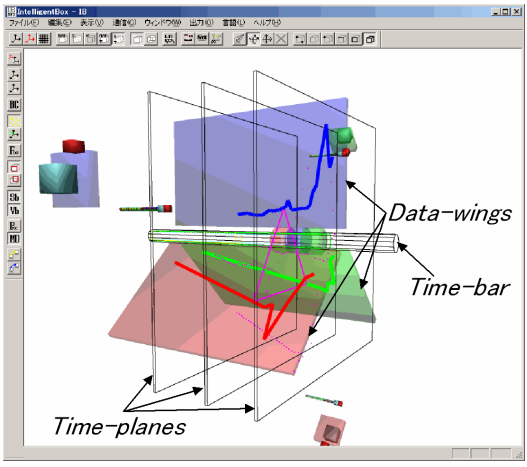


Fig. 4. Screen image of Time-tunnel

4.2 Parallel Coordinates Version of Time-Tunnel (PCTT)

When visualizing a large number of time-series numerical data records, the user has to prepare exactly the same number of data-wings. For example, when there are three time-series numerical data records, three data-wings are used as shown in Figure 4. In this case, each chart data is regarded as one multidimensional data record, i.e., one database record with multi-attributes. In addition, as shown in Figure 5, the user can manipulate data-wings to put them overlapped together. This works as Parallel Coordinates. However, when the user wants to visualize a huge number of database records, he/she has to prepare exactly the same number of data-wings and practically it is impossible to manipulate them. To deal with this problem, we extended the functionality of data-wing as explained in the following.

Figure 7 shows Parallel Coordinates version of Time-tunnel. We extended data-wing to enable it to display more than one time-series numerical data records, i.e., multiple database records as multiple charts, in it like Parallel Coordinates. Even if there are a huge number of database records to be visualized, the user can divide them into several groups and assign each group to one of the multiple data-wings of the same Time-tunnel. For example, as shown in Figure 8, database records are divided into three groups. In addition, it is possible to display multiple Parallel Coordinates on different views by using already existing component called CameraBox of IntelligentBox. In Figure 8, an upper left, upper right, and lower left view are treated as three individual Parallel Coordinates of center Time-tunnel. In this way, the user can visualize a huge number of database records using PCTT. The user can select one record that he/she wants to analyze in each data-wing and the selected chart is soon highlighted in red color.

Since the user can rotate and put any data-wings overlapped together, he/she can compare his/her selected records by looking at highlighted charts. Furthermore, the radar chart for the selected charts can also be displayed similarly to original Time-tunnel. Such a radar chart is the same as shown in Figure 6.

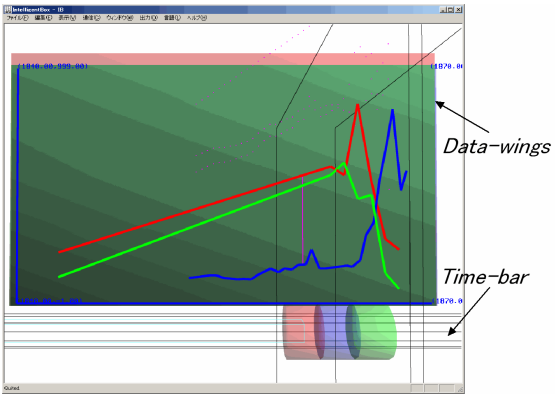


Fig. 5. Screen image of Time-tunnel

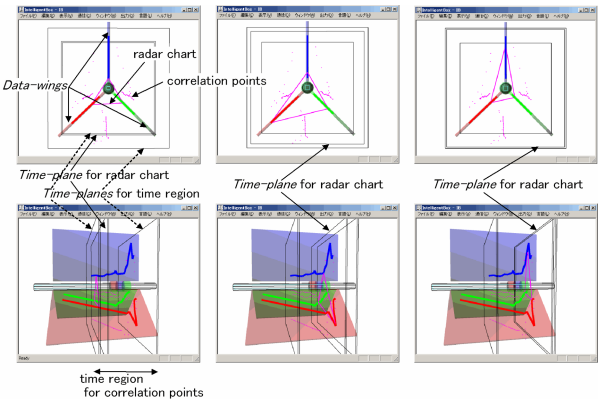


Fig. 6. Radar chart views of original Time-tunnel

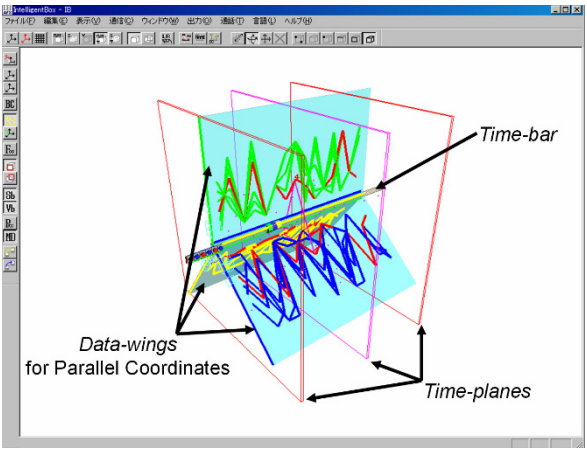


Fig. 7. Parallel Coordinates version of Time-tunnel

When database records have too many attributes, it is impossible to visualize them in one rectangular area as one Parallel Coordinates due to the width size limitation of a display screen. Using multiple data-wings of Time-tunnel, the user can divide attributes into several groups and assign each group to one of the multiple data-wings. In this case, the user can visualize database records with a huge number of attributes using multiple data-wings. For this case, we also extended radar chart visualization functionality to visualize relationships among different attributes of all multidimensional data as multiple radar charts as shown in Figure 9. This visualization is possible because the number of data in each data-wing is the same in this case. As Figure 9 shows, it is possible to understand relationships between the two attributes corresponding to any two adjacent data-wings about all data at a glance.

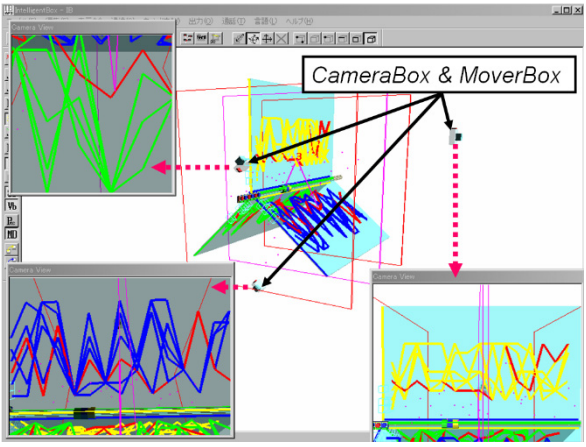


Fig. 8. Different views of multiple Parallel Coordinates using multiple CameraBoxes

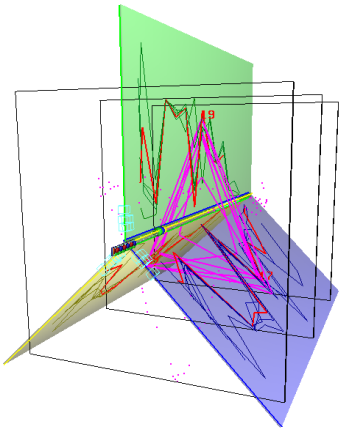


Fig. 9. Screen image of PCTT and its multiple radar charts

4.3 2Dto2D Visualization Functionality

In this subsection, we introduce 2Dto2D visualization functionality added to PCTT. Its conceptual image is as shown in Figure 10. 2Dto2D visualization functionality displays multiple lines those represent four-dimensional (four attributes) data drawn from one (2D, two attributes) plane to another (2D, two attributes) plane. Using 2Dto2D visualization for multiple attribute data, it is easy to understand relationships of four attributes of each data. Figure 11 shows a screen snapshot of actual PCTT with 2Dto2D visualization. In this case, there are three data-wings so that there are three 2Dto2D visualization areas as shown in the right figure of Figure 11.

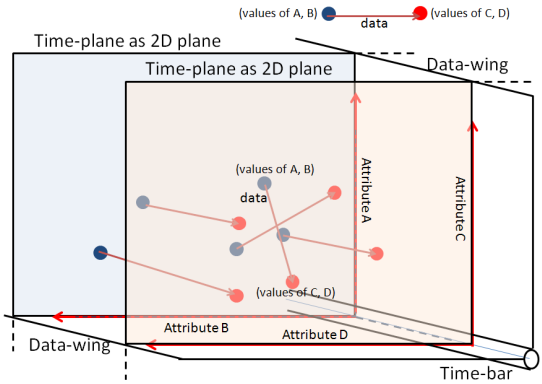


Fig. 10. Conceptual image of PCTT with 2Do2D visualization

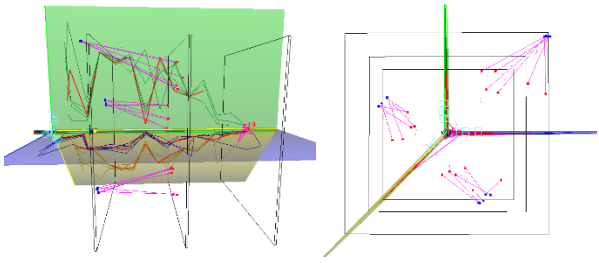


Fig. 11. Screen images of PCTT with 2Dto2D visualization

5 Network Data Visualization Using PCTT with 2Dto2D Visualization

5.1 IP Packet Data

Network data is considered as a set of IP packet data. IP packet has several attributes, mainly, source and destination IP, source and destination Port, Protocol type, and Packet size. Using Parallel Coordinates, it is possible to represent each IP packet as one polyline as shown in Figure 12. Individual axis corresponds to each of the attributes of IP packet data. Furthermore, Figure 13 shows 2Dto2D visualization image for IP packets. In this case, relationships among 2 attributes (source IP, source Port) to 2 attributes (destination IP, destination Port) can be visualized. To detect intrusion attacks, this visualization is very significant because intrusion attacks have a certain access pattern strongly related to the four attributes of IP packet data, i.e., source IP, destination IP, source Port, and destination Port.

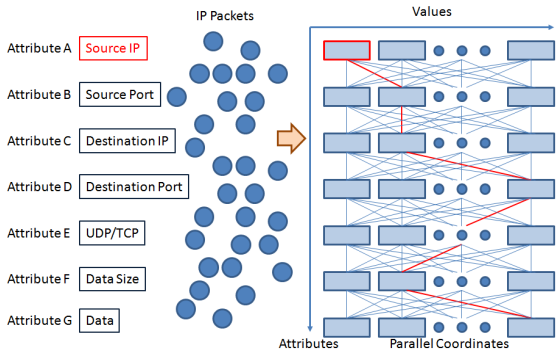


Fig. 12. Parallel Coordinates visualization for IP packets

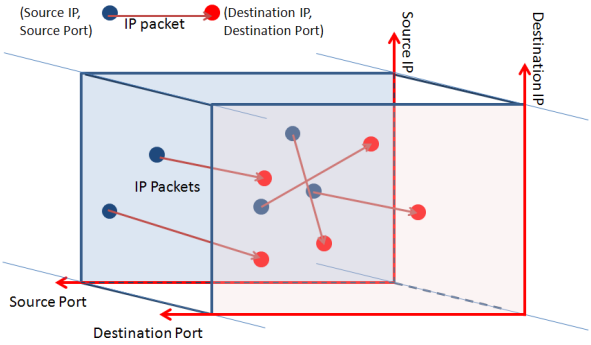


Fig. 13. 2Dto2D visualization for IP packets

5.2 PCTT with 2Dto2D Visualization for IP Packet Data

Figure 14 shows screen images of actual PCTT with 2Dto2D visualization for IP packet data. Here, we explain several components used for the visualization besides the main components of Time-tunnel. The top figure of Figure 14 shows the case that multiple radar charts and 2Dto2D visualization are both displayed.

The several components of the left part in this figure are dedicated for setting a begin time and an interval time of captured IP packet data, and for displaying such data, e.g., the total number of IP packets in the day, the number of IP packets in the current interval time, etc. The middle figure of Figure 14 shows the case that only multiple radar charts are displayed. In this case, it is possible to easily understand the relationships between any two attributes of the four attributes each of which corresponds to each of the four data-wings about all IP packet data. Finally, the bottom figure of Figure 14 shows the case that only 2Dto2D visualization is enabled. Since the four attribute set is the same as that of Figure 13, this case is suitable for the intrusion detection.

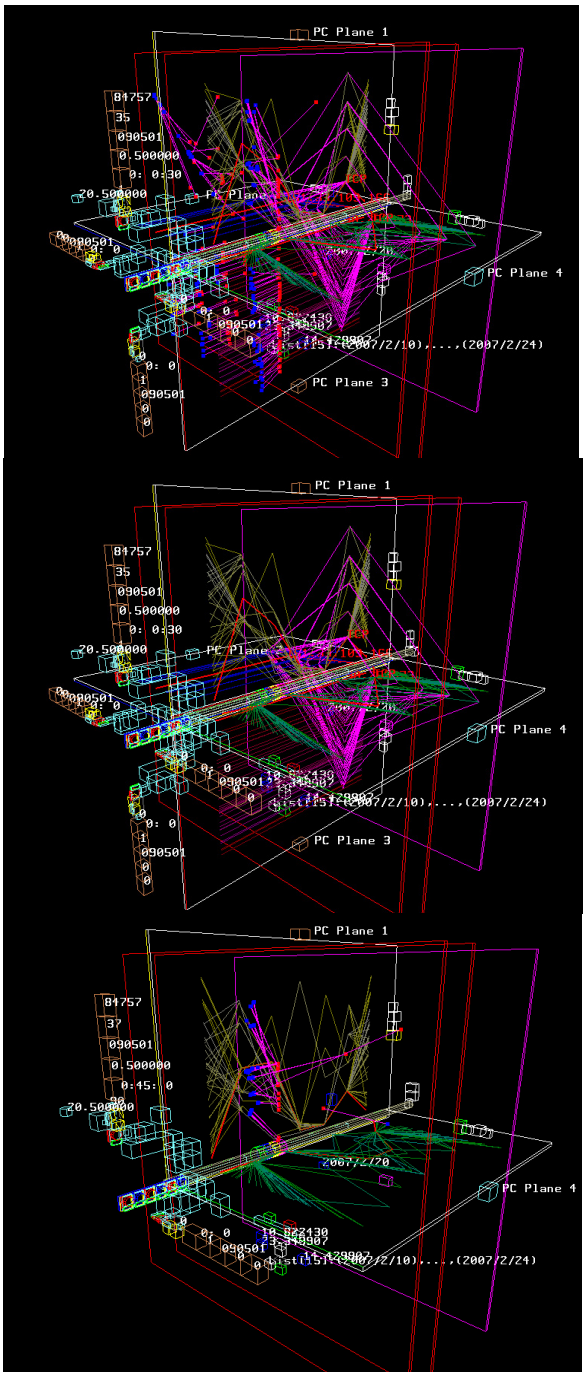


Fig. 14. Screen images of PCTT with 2Dto2D visualization for IP packet data

5.3 Intrusion Detection

We use darknet flow data of IP packets sent from the outside of our university and captured as pcap format files. Each file includes IP packet data in one hour and the average number of them in a file is around 3,500. PCTT can read 24 hours files at once so that it can visualize IP packet data of one day at maximum. Also, we can specify an interval time and its begin time for visualizing IP packet data using the GUI of PCTT as previously explained. There is an automatic change mode for the begin time. In this mode, visualization results are automatically changed according to the begin time. When the interval time is 30 seconds, a begin time will be shifted every 30 seconds, and one shift needs around 0.1 seconds as a real-execution time although several hundreds of IP packets are included in each of these intervals. So, even if you want to check visualization results of IP packets in a whole day, you need only 5 minutes. This value is reasonable although it dependents on the specification of the PC you use because we used a standard PC whose specification is as follows: CPU: Intel Core_i5, Memory: 4GB and no special graphics card. How many polylines are displayed atomically is regarded as the performance of PCTT. Its number is a few thousands. This number is enough for practical cases because it is difficult for a human to understand features of data if the data are represented as more than thousands of polylines. The followings are a couple of network attack patterns those are actually detected using PCTT with 2Dto2D visualization.

Port Scanning

Port Scanning is one of the most popular techniques attackers use to discover services that they can exploit to break into systems. During checking IP packet data of four days, we found only one case like port scanning as shown in Figure 15. In this case, a certain computer located outside of our university sequentially access to different source IP and source Port of our darknet in a very short period.

Security Holes Attacks

Security holes mean shortcoming of a computer program (software code) that allows unauthorized users (attackers) to gain access to a system or network, and to interfere with its operations and data. Figure 16 is regarded to show access patterns of security holds attacks because they indicate the cases that the computer of an attacker tried to check security holes of many target computers virtually located in our darknet.

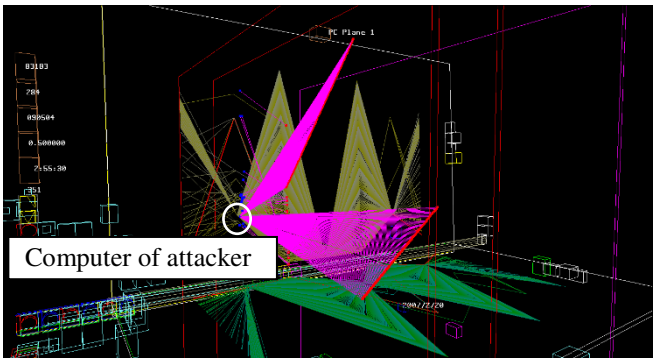


Fig. 15. Access patters of port scanning

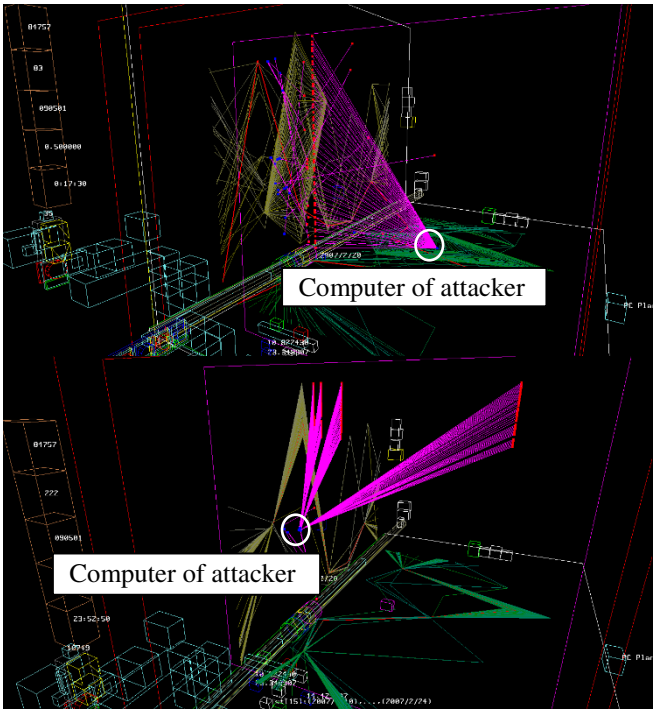
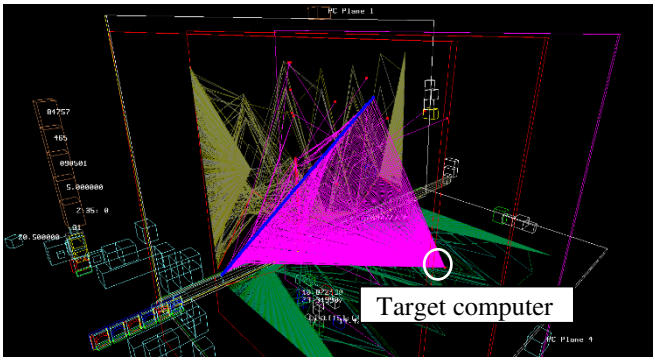


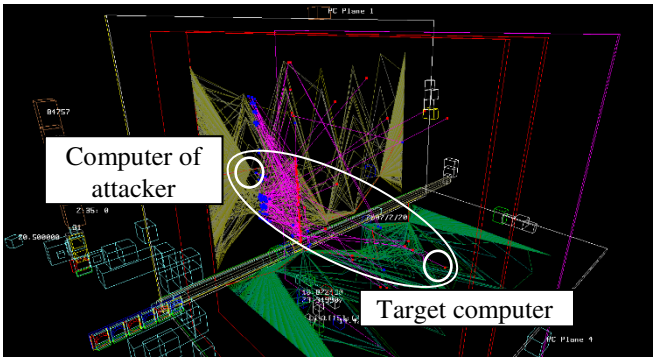
Fig. 16. Access patterns of security holes attackes

DoS Attacks

DoS attack means Denial of Service attack. There are several modes of the attack. The most popular access pattern is one computer of an attacker simultaneously accesses many times to his/her target computer in a very short period. As a result, the target computer will become disenable to provide the services that the computer originally provided. Sometime, the computer will become malfunctioned. Figure 17 is regarded to show access patterns of DoS attacks because they indicate such a case. Indeed, the upper figure of Figure 17 shows different 2Dto2D visualization, i.e., 2D (time, time) to 2D (destination IP, destination Port). Therefore, blue points located from the left lower to the right upper mean the transition of time about the corresponding IP packs those all tried to access to one target computer. As shown in the lower figure of Figure 17, their source IP and source Port are the same.



2D(time, time) to 2D(destination IP, destination Port) visualization



2D (src IP, src Port) to 2D (dest IP, dest Port) visualization

Fig. 17. Access patterns of DoS attacks

DDoS Attack

DDoS attack means Distributed Denial of Service attack. The most popular access pattern is more than one computers controlled by an attacker simultaneously accesses many times to his/her target computer in a very short period. As a result, the target computer will become disenable to provide the services that the computer originally provided. Sometime, the computer will become malfunctioned. Figure 18 shows such access patterns.

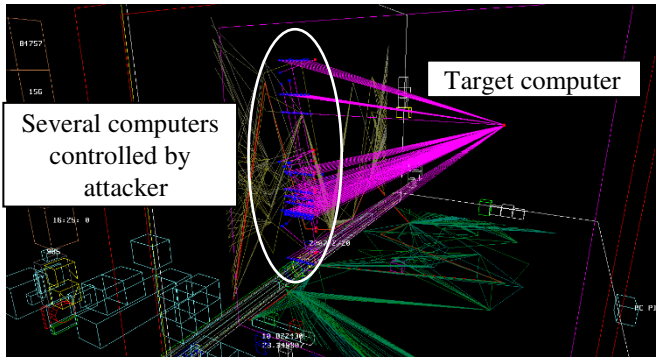


Fig. 18. Access patterns of DDoS attacks

6 Combinatorial Use of PCTT for Macro Level to Micro Level Visual Analytics

Through the way described in the previous subsection, you can find the characteristics of IP packets data using PCTT with 2Dto2D visualization at the attribute level, i.e., at the micro level. However, statistical analysis is also necessary to find out suspicious periods of time that seem to be attacked. This is regarded as the macro level visual analytics. Figure 19 shows statistics visualization results of IP packets data in several different interval times, 30 minutes to 0.5 minutes. As shown in the figures, the granularity of interval time is significant because the visualization results are strongly dependent on the interval time. The interval times of one minute or 30 seconds are suitable for the visualization of IP packets data of our darknet because the both results are almost the same. Besides the total number of data in a certain interval, the system provides several functionalities for calculating statistical values. For instance, those are the information entropy, the variance and the number of different kinds of data. The information entropy h is calculated using the following expression.

$$h_N = -\sum_{i=1}^{N} p_i \log_2 p_i. \quad (1)$$

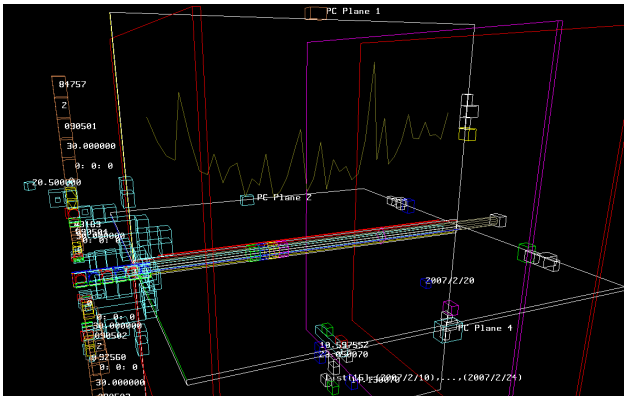
Here, $p_i$is the ratio of the number of the same kind of data to N, the total number of data, in a certain interval. Indeed, we use H as the normalized entropy because the maximum value of h_N depends on N. H is calculated by the following expression.

$$H = \frac{h_N}{\log_2 N} \times 100. \tag{2}$$

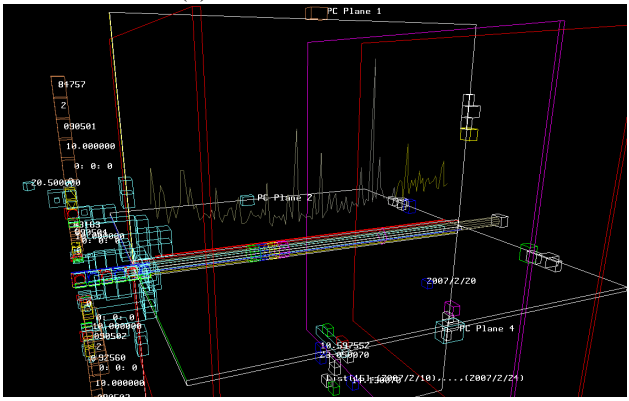
Also, the variance V is calculated using the following expression.

$$V = \frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2. \tag{3}$$

Here, $x_i$is the number of the same kind of data, $\overline{x}$ is the average of x_1 to x_N, and N is the number of different kinds of data in a certain interval.

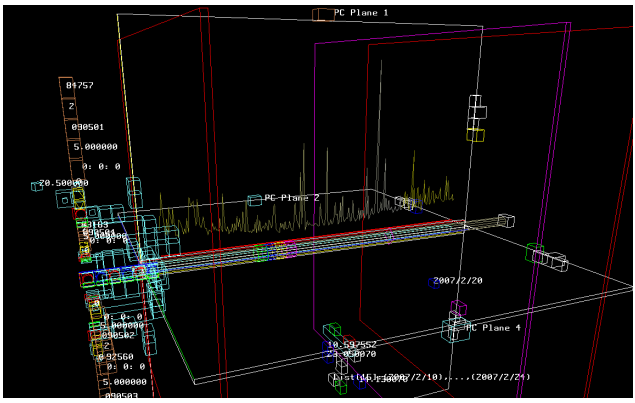


(a) Interval is 30 minutes

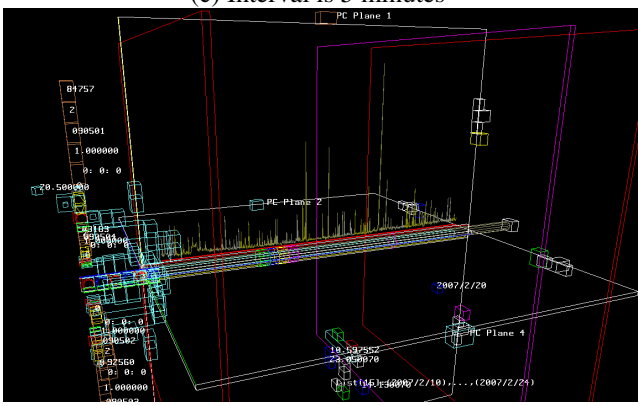


(b) Interval is 10 minutes

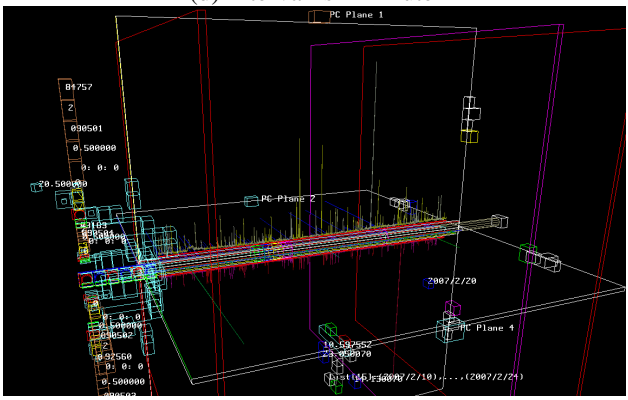
Fig. 19. Total number of IP packets in different interval times during a certain day



(c) Interval is 5 minutes

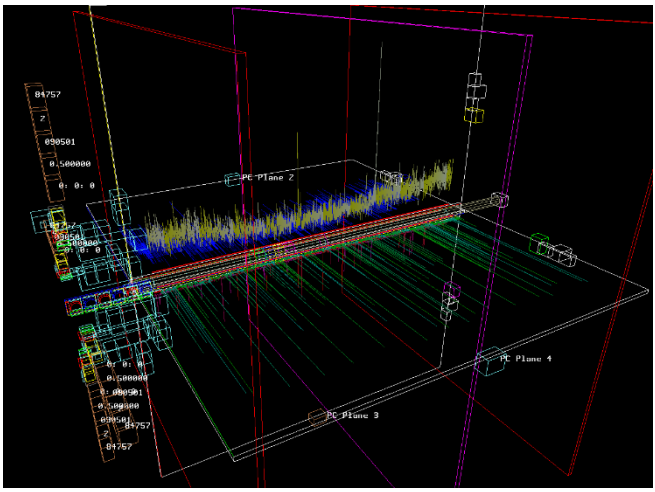


(d) Interval is 1 minute

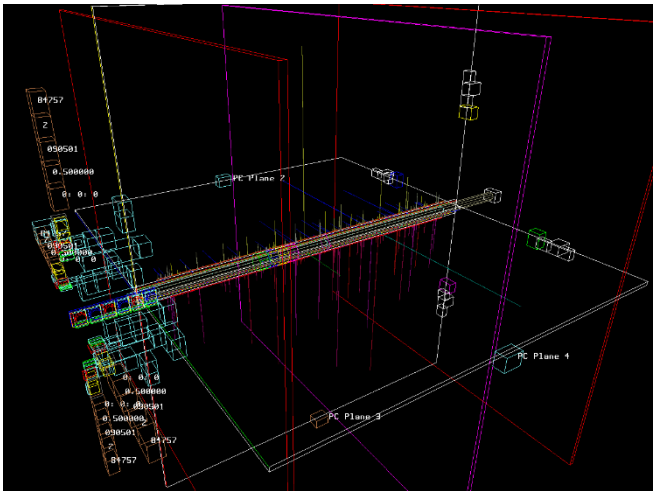


(e) Interval is 30 seconds

Fig. 19. (*continued*)

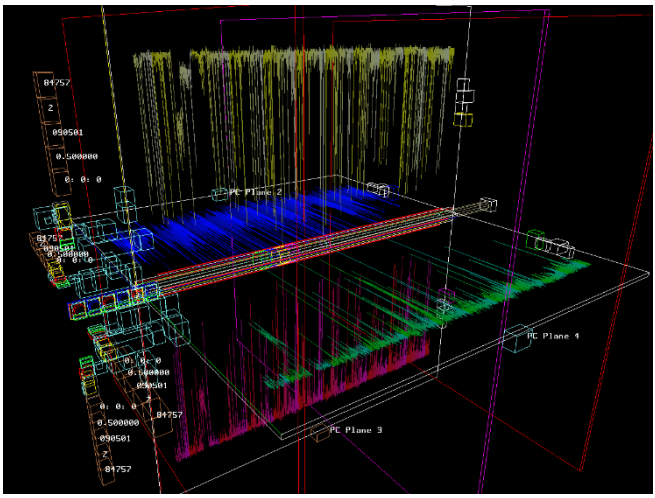


(a) Numbers of different kinds of source IPs, destination IPs, source Ports, and destination Ports in each 30 seconds during a certain day.



(b) Variances of numbers of different kinds of source IPs, destination IPs, source Ports, and destination Ports in each 30 seconds during a certain day.

Fig. 20. Different statistics results of source IPs, destination IPs, source Port, and destination Ports in each 30 seconds during a certain day

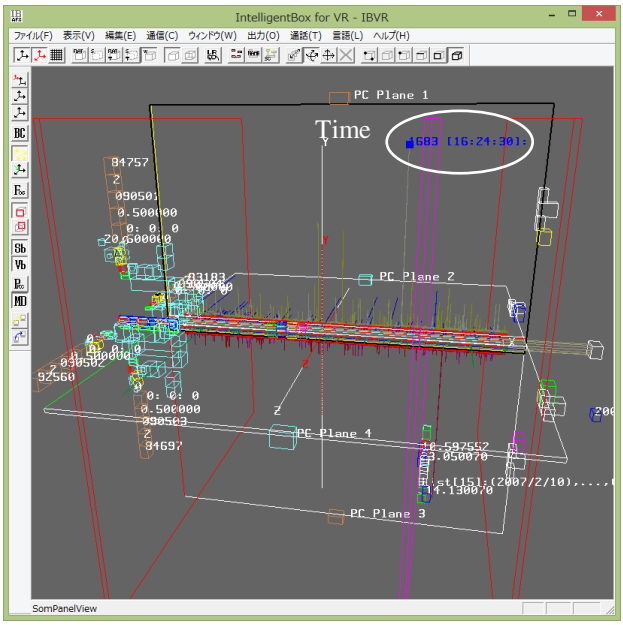


(c) Information Entropies of different kinds of source IPs, destination IPs, source Ports, and destination Ports in each 30 seconds during a certain day.

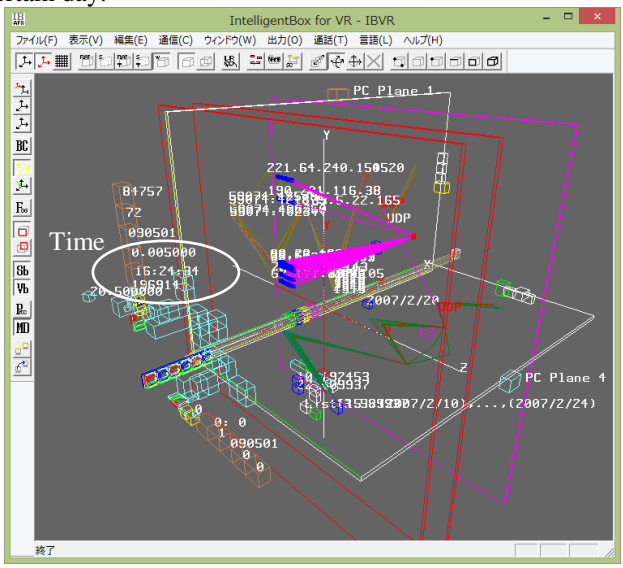
Fig. 20. (*continued*)

Figure 20 shows different statistics results of source IPs, destination IPs, source Ports, and destination Ports in each 30 seconds during a certain day using four data-wings. In this way, using multiple data-wings, it is possible to analyze statistics results of multiple attributes together at once. As for the intrusion detection, DoS and DDoS attacks indicate lower values of Information Entropy about both source IPs and destination IPs. Port scanning indicates lower value and higher value of Information Entropy about source IPs and destination Ports, respectively. Security hole attacks indicate lower values of Information Entropy about both source IPs and destination Ports.

Figure 21 shows an actual example of the combinatorial use of PCTT with 2Dto2D visualization at from macro level to micro level visual analytics. At the macro level, the time interval [16:24:30] indicates very high value rather than others as shown in the upper figure and it seems that any suspicious actions were occurred in this time period. So, if you check the same time period using PCTT with 2Dto2D visualization, you can find that DDoS attacks were occurred as shown in the lower figure of Figure 21.



(1) Macro level visualization: Total number of IP packets in each 30 seconds during a certain day.



(2) Micro level visualization: 2Dto2D visualization of PCTT at the same interval as the above.

Fig. 21. Combinatorial use of PCTT with 2Dto2D visualization from macro level to micro level visualization

7 Other Visualization Examples

We have one project about cyber-physical systems. One of the important topics in researches on cyber-physical systems is the analysis of big data to be collected from the physical world using various sensors. As one of the analysis methods, the information visualization is useful. Currently, we have been developing two types of visualization tools using IntelligentBox system for our cyber-physical system project. The first one is for the analysis of human movements because human movements are very important cues for the analysis of human activities. Figure 22 shows a screen image of such visualization tool. This is the building of our graduate school consisting of several floors. To simplify a 3D model for the building, we employ 2D floor map images and use the texture mapping mechanism called 2.5D inside building visualizer. It is very easy to visualize the building and display human movements. In this case, glyphs have a different color, a different size, and a different shape to specify several attribute values of their corresponding persons and move on a floor. So, we can understand persons' activities from glyphs' movements.

The other one is PCTT because our human activity data are consisting of several attributes collected by various sensors from physical world activities as shown in Figure 23. Although we have not yet realized the combinatorial use of PCTT and the 2.5D inside building visualizer of Figure 22, it is possible to combine them and realize it in the near future because all the visualization tools are implemented as composite components of IntelligentBox.

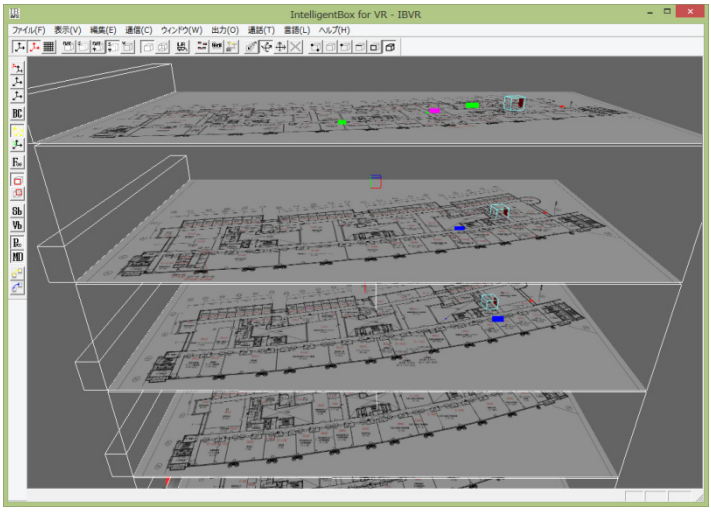


Fig. 22. 2.5D inside building visualizer

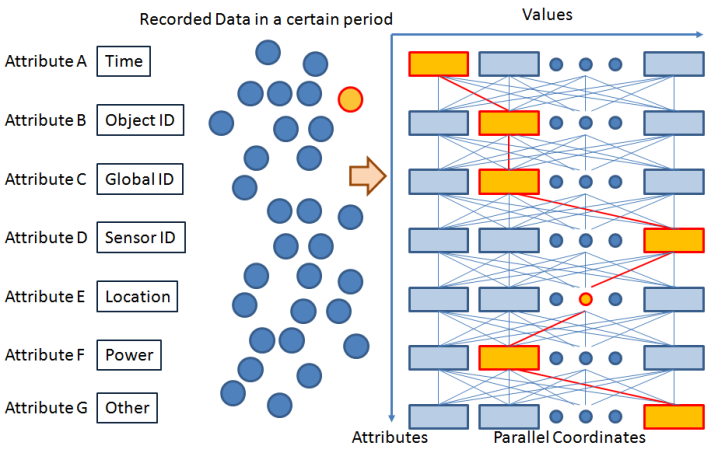


Fig. 23. Parallel Coordinates visualization for human activity data

8 Conclusion and Remarks

In this chapter, we treated Parallel Coordinates version of Time-tunnel (PCTT). Originally, Time-tunnel is a multidimensional data visualization tool and its Parallel Coordinates version provides the functionality of Parallel Coordinates visualization. It can be used for the visualization of network data because IP packet data have many attributes and such multiple attribute data can be visualized using Parallel Coordinates. In this chapter, we mainly proposed the combinatorial use of PCTT and 2Dto2D visualization functionality. 2Dto2D visualization functionality displays multiple lines represented as four-dimensional (four attributes) data drawn from one two-dimensional (two attributes) plane to the other two-dimensional (two attributes) plane in a 3D space. This 2Dto2D visualization functionality was introduced to PCTT. Network attacks have a certain access pattern strongly related to the four attributes of IP packet data, i.e., source IP, destination IP, source Port, and destination Port. So, 2Dto2D visualization is useful for detecting such access patterns. We showed several network attack patterns actually visualized using PCTT with 2Dto2D visualization as examples of the intrusion detection.

Using PCTT with 2Dto2D visualization functionality, it is possible to investigate access patterns of network attacks at the attributes level of IP packets. However, statistical analysis is also necessary to find out suspicious periods of time that seem to be attacked. This is regarded as the macro level visual analytics and the former is regarded as the micro level visual analytics. In this chapter, we also introduced such combinatorial use of PCTT for macro level to micro level visual analytics of network data as an example of multidimensional data. In addition, we introduced other visual analytics example to clarify the usefulness of PCTT about sensor data related to our cyber physical systems research project.

As future work, we will investigate more details about suspicious accesses of network data indicated as intrusion accesses by the proposed visualization tools. Also, we will try to use the proposed visualization tools for various types of data such as human activity data to clarify its usefulness.

Acknowledgments. This work was partially supported by Proactive Response Against Cyber-attacks Through International Collaborative Exchange (PRACTICE), Ministry of Internal Affairs and Communications, Japan.

References

1. Akaishi, M., Okada, Y.: Time-tunnel: Visual Analysis Tool for Time-series Numerical Data and Its Aspects as Multimedia Presentation Tool. In: Proc. of 8th Int. Conf. on Information Visualization (IV 2004), pp. 456–461. IEEE CS Press (2004)
2. Akaishi, M., Okada, Y.: Time-tunnel: Visual Analysis Tool for Time-series Numerical Data and Its Combinational Variation. In: Proc. of 1st Int. Conf. on Geometric Modeling, Visualization & Graphics (GMVAG 2005), Salt Lake, USA, July 21- 26 (2005)
3. Notsu, H., Okada, Y., Akaishi, M., Niijima, K.: Time-tunnel: Visual Analysis Tool for Time-series Numerical Data and Its Extension toward Par-allel Coordinates. In: Proc. of Int. Conf. on Computer Graphics, Imaging and Vision (CGIV 2005) (July 2005)
4. Inselberg, A., Dimsdale, B.: Parallel Coordinates: A Tool for Visualizing Multi-dimensional Geometry. In: Proc. IEEE Visualization 1990, pp. 361–378. IEEE CS Press (1990)
5. Okada, Y., Tanaka, Y.: IntelligentBox: A Constructive Visual Software Development System for Interactive 3D Graphic Applications. In: Proc. of Computer Animation 1995, pp. 114–125. IEEE CS Press (1995)
6. Martin, A., Ward, M.O.: High dimensional brushing for interactive explo-ration of multivariate data. In: Proc. IEEE Visualization 1995, pp. 271–278 (1995)
7. Fua, Y.-H., Ward, M.O., Rundensteiner, E.A.: Hierarchical Parallel Coor-dinates for Exploration of Large Datasets. In: Proc. IEEE Visualization 1999, pp. 43–50. IEEE CS Press (1999)
8. Hauser, H., Ledermann, F., Doleisch, H.: Angular Brushing of Extended Parallel Coordinates. In: IEEE Information Visualization (InfoVis 2002), pp. 127–130 (2002)
9. Graham, M., Kennedy, J.: Using Curves to Enhance Parallel Coordinate Visualizations. In: Proc. Information Visualization IV 2003, pp. 10–16. IEEE CS Press (2003)
10. Artero, A.O., Ferreira de Oliveira, M.C., Levkowitz, H.: Uncovering Clus-ters in Crowded Parallel Coordinates Visualizations. In: IEEE Information Visualization 2004 (InfoVis 2004), pp. 131–136 (2004)
11. Johansson, J., Cooper, M., Jern, M.: 3-Dimensional Display for Clustered Multi-Relational Parallel Coordinates. In: IEEE Information Visu-alization (InfoVis 2005), pp. 188–193 (2005)
12. `http://davis.wpi.edu/~xmdv/news.html`
13. Lanzenberger, M., Miksch, S.: The Stardinates - Visualizing Highly Structured Data. In: Proc. of Information Visualization IV 2003, pp. 47–52. IEEE CS Press (2003)
14. Fanea, E., Carpendale, S., Isenberg, T.: An Interactive 3D Integration of Parallel Coordinates and Star Glyphs. In: IEEE Information Visualization (InfoVis 2005), pp. 149–156 (2005)

15. Tominski, C., Abello, J., Schumann, H.: 3D Axes-Based Visualizations for Time Series Data, Poster Chapter. In: IEEE Information Visualization (InfoVis 2005) (2005)
16. McPherson, J., Ma, K.-L., Krystosk, P., Bartoletti, T., Christensen, M.: Portvis: A tool for port-based detection of security events. In: ACM VizSEC 2004 Workshop, pp. 73–81 (2004)
17. Muelder, C., Ma, K.-L., Bartoletti, T.: Interactive Visualization for Network and Port Scan Detection. In: Valdes, A., Zamboni, D. (eds.) RAID 2005. LNCS, vol. 3858, pp. 265–283. Springer, Heidelberg (2006)
18. Boschetti, A., Muelder, C., Salgarelli, L., Ma, K.-L.: TVi: A Visual Querying System for Network Monitoring and Anomaly Detection. In: The 8th Int. Symp. on Visualization for Cyber Security, VizSec 2011 (2011)
19. Kintzel, C., Fuchs, J., Mansmann, F.: Monitoring Large IP Spaces with ClockView. In: The 8th Int. Symp. on Visualization for Cyber Security, VizSec 2011 (2011)
20. Best, D.M., Bohn, S., Love, D., Wynne, A., Pike, W.A.: Real-Time Visualization of Network Behaviors for Situational Awareness. In: VizSec 2010, pp. 79–90 (2010)
21. Yin, X., Yurcik, W., Treaster, M., Li, Y., Lakkaraju, K.: VisFlowConnect: NetFlow Visualizations of Link Relationships for Security Situational Awareness. In: VizSEC/DMSEC 2004, pp. 26–34 (2004)
22. Axelsson, S.: Visualization for Intrusion Detection - Hooking the Worm. Understanding Intrusion Detection Through Visualization Advances in Information Security 24, 111–127 (2006)
23. Chu, M., Ingols, K., Lippmann, R., Webster, S., Boyer, S.: Visualizing Attack Graphs, Reachability, and Trust Relationships with NAVIGATOR. In: The 7th Int. Symp. on Visualization for Cyber Security, VizSec 2010, pp. 22–33 (2010)
24. Itoh, T., Takakura, H., Sawada, A., Koyamada, K.: Hierarchical Visualization of Network Intrusion Detection Data. IEEE Computer Graphics and Applications, 40–47 (March/April 2006)
25. Lau, S.: The Spinning Cube of Potential Doom. Communications of the ACM 47(6), 25–26 (2004)
26. Wang, W., Lu, A.: Visualization Assisted Detection of Sybli Attacks in Wireless Networks. In: VizSEC 2006, pp. 51–60 (2006)
27. Malecot, E.L., Kohara, M., Hori, Y., Sakurai, K.: Interactively Combining 2D and 3D Visualization for Network Traffic Monitoring. In: VizSEC 2006, pp. 123–127 (2006)
28. Oberheide, J., Karir, M., Blazakis, D.: VAST: Visualizing Autonomous System Topology. In: VizSec 2006, pp. 71–79 (2006)
29. Inoue, D., Suzuki, M., Eto, M., Yoshioka, K., Nakao, K.: DAEDALUS: Novel Application of Large-Scale Darknet Monitoring for Practical Protection of Live Networks (Extended Abstract). In: Kirda, E., Jha, S., Balzarotti, D. (eds.) RAID 2009. LNCS, vol. 5758, pp. 381–382. Springer, Heidelberg (2009)
30. Nicter Cube of nicter, `http://www.nicter.jp/nw_public/scripts/cube.php`

Towards a Big Data Analytics Framework for IoT and Smart City Applications

Martin Strohbach, Holger Ziekow, Vangelis Gazis, and Navot Akiva

AGT International
Hilpertstrasse 35, 64295 Darmstadt, Germany
{mstrohbach,hziekow,vgazis,nakiva}@agtinternational.com

Abstract. An increasing amount of valuable data sources, advances in Internet of Things and Big Data technologies as well as the availability of a wide range of machine learning algorithms offers new potential to deliver analytical services to citizens and urban decision makers. However, there is still a gap in combining the current state of the art in an integrated framework that would help reducing development costs and enable new kind of services. In this chapter, we show how such an integrated Big Data analytical framework for Internet of Things and Smart City application could look like. The contributions of this chapter are threefold: (1) we provide an overview of Big Data and Internet of Things technologies including a summary of their relationships, (2) we present a case study in the smart grid domain that illustrates the high-level requirements towards such an analytical Big Data framework, and (3) we present an initial version of such a framework mainly addressing the volume and velocity challenge. The findings presented in this chapter are extended results from the EU funded project BIG and the German funded project PEC.

1 Introduction

In times of increasing urbanization, local decision makers must be prepared to maintain and increase the quality of life of a growing urban population. For instance, there are major challenges related to minimizing pollution, managing traffic as well as making efficient use of scarce energy resources. For instance, in regard to congested traffic conditions, the Confederation of British Industries estimates that the cost of road congestion in the UK is GBP 20 billion (i.e., USD 38 billion) annually. In addition to challenges related to the efficient use of natural and manmade resources ensuring the health and safety of urban citizens, e.g., in the context of large events or supporting law enforcement are key concerns of a modern smart city.

In order to address these challenges urban decision makers as well as citizens will need the capacity to make the right assessment of urban situations based on correct data, and, more importantly, they will need the key information contained in the data to assist them in their decision processes.

As put by Neelie Kroes, EU commissioner for the Digital Agenda, data is the new gold, meaning that data is a valuable resource that can be mined for creating new values. In the context of smart cities, there is an abundance of data sources that can be

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_11

mined by applying data analytics techniques and generate value by offering innovative services that increase citizens' quality of life. Data may be provided by all stakeholders of a Smart City [10], i.e., the society represented by citizens and businesses and governments represented by policy makers and administrations.

On one hand data sources may include traditional information held by public bodies (Public Sector Information, PSI) including anonymous data such as cartography, meteorology, traffic, and any kind of statistics data as well as personal data, e.g., from public registries, inland revenues, health care, social services etc. [67].

On the other hand citizens themselves create a constant stream of data in and about cities by using their smartphones. By using apps like Twitter and Facebook, or apps provided by the city administration, they leave digital traces related to their activities in the physical city that has the potential to create valuable insights for urban planners.

With the advent of deployed sensor systems such as mobile phone networks, camera networks in the context of intelligent transportation systems (ITS) or smart meters for metering electricity usage, new data sources are emerging that are often discussed in the context on the Internet of Things (IoT), i.e., the extension of the internet to virtually every artifact of daily life by the use of identification and sensing technologies.

Thus, we can summarize that the key components required for Smart City application are available: 1) an abundance of data sources, 2) infrastructure, networks, interfaces and architectures are being defined in the IoT and M2M community, 3) a vast range of Big Data technologies are available that support the processing of large data volumes, and 4) there is ample and wide knowledge about algorithms as well as toolboxes [62] that can be used to mine the data.

Despite all the necessary conditions for a Smart City are met, there is still a lack of an analytical framework that pulls all these components together such that services for urban decision makers can easily be developed.

In this chapter, we address this need by proposing an initial version of such an analytical framework that we derived based on existing state of the art, initial findings from our participation in the publicly funded projects Big Data Public Private Forum (BIG) [7] and Peer Energy Cloud (PEC) [46] as well as our own experiences with analytical applications for Smart Cities.

The European Project BIG is a Coordinated Support Action (CSA) that seeks to build an industrial community around Big Data in Europe with the ultimate goal to develop a technology roadmap for Big Data in relevant industrial sectors. As a part of this effort the BIG project gathers requirements on Big Data technologies in industry driven working groups. Groups relevant for Smart Cities include health, public sector, energy, and the transport working group. The project has released both an initial version of requirements in these and other sectors [67] as well as a set of technical white papers providing an overview of the state of the art in Big Data technologies [31]. In this chapter, we draw from these results of the BIG project and complement it with the findings from a concrete use case in the energy sector as carried out in the PEC project.

The remainder of this chapter is structured as follows: Section 2 provides background about the technical challenges associated with the Big Data and Internet of Things topics. Section 3 summarizes the state of the art in Big Data technologies. In Section 4 we elaborate on the concrete Big Data challenges that need to be addressed

in the context of Smart City applications. Section 5 presents a case study from the smart grid domain that demonstrates how we applied big data analytics in a realistic setting. In Section 6, we extend the analytics presented in this case study towards an initial big data analytics framework. In section 7 we report on our lessons learned. In Section 8 we summarize further research directions required to extend the framework fully implement it. Finally, Section 9 concludes this chapter.

2 Background: Big Data and the Internet of Things

In this section we describe the technologies that are concerned with connecting everyday artefacts, i.e., the Internet of Things, and relate them to Big Data technologies that address the challenges of managing and processing large and complex data sets. For an extensive discussion and definition of the term Big Data we refer to the respective report of McKinsey [37].

2.1 The Various Faces of Big Data

Although managing and processing large data sets is not fundamentally new, during the past years a range of technologies have emerged that facilitate the efficient storage and processing of big data sets. While Big Data technologies such as Map Reduce [14] and Hadoop [63] are the result of big Internet companies such as Google and Yahoo!, the need to handle and process large data sets is quickly extending to other sectors. For instance, the amount of various patient data in the health sector offers new opportunities for better treatments [67] and the advent of smart meters, allows utility provider to better cope with the instabilities of the grid caused by renewable energy sources [33]. From a technological perspective Big Data challenges and technologies can best be described along the so-called 3 V's: Volume, Velocity, and Variety [32].

The Volume Challenge. The Volume challenge refers to storing, processing, and quickly accessing large amounts of data. While it is hard to quantify the boundary for a volume challenge, common data sets in the order of hundreds of Terabyte or more are considered to be big. In contrast to traditional storage technologies such as relational database management systems (RDBMS), new Big Data technologies such as Hadoop are designed to easily scale with the amount of data to be stored and processed. In its most basic form, the Hadoop system uses its Hadoop Distributed File System (HDFS), to store raw data. Parallel processing is facilitated by means of its Map Reduce framework that is highly suitable for solving any embarrassingly parallel processing problems. With Hadoop it is possible to scale by simply adding more processing nodes to the Hadoop cluster without the need to do any reprogramming as the framework takes care of using additional resources as they become available.

Summarizing the trends in the volume challenge one can observe a paradigm shift with respect to the way the data is handled. In traditional database management systems the database design is optimized for the specific usage requirements, i.e., data is preprocessed and only the information that is considered relevant is kept. In contrast,

in a truly data-driven enterprise that builds on Big Data technologies, there is awareness that data may contain value beyond the current use. Thus, a master data set of the raw data is kept that allows data scientists to discover further relationships in the data, relationships that may reside beyond the requirements of today. As a side effect it also reduce the costs of human error such as erroneous data extraction or transformation.

The Velocity Challenge. Velocity refers to the fact that data is streaming into the data infrastructure of the enterprise at a high rate and must be processed with minimal latency. To this end, different technologies are applicable, depending on the amount of state and complexity of analysis [57]. In cases where only little state is required (e.g., maintaining a time window of incoming values), but complex calculations need to be performed over a temporally scoped subset of the data, Complex Event Processing (CEP) engines (see section 3.3) offer efficient solutions for processing incoming data in a stream manner. In contrast, when each new incoming data set needs to be related to a large number of previous records, but only simple aggregations and value comparisons are required, noSQL databases offer the necessary write performance. The required processing performance can then be achieved by using streaming infrastructures such as Storm [56] or S4 [51].

The Variety Challenge. In a data-driven economy the objective is to maximize the business value by considering all available data. From a technical perspective one could formulate that problem by evaluating a function over all accessible data sets [38]. In practice, however, this approach must confront the challenge of heterogeneous data sources ranging from unstructured textual sources (e.g., social media data) to the wide disparity in the formats of sensor data. Traditionally this challenge is addressed by various forms of data integration. In the context of Big Data there is however a new dimensions to the integration challenge which is the amount of different data sources that need to be integrated. Social media, open (governmental) data sources [12], [21], and data platforms [64] and markets [11], result in a data ecosystem of significant variation. As the integration of new data sources requires manual work to understand the source schema, to define the proper transformations and to develop data adapters, existing approaches do not scale effectively.

Veracity. Apart from the original 3V's described above, an almost inexhaustible list of Big Data V's are discussed. For instance veracity relates to the trust and truthfulness of the data. Data may not fully be trusted, because of the way it has been acquired, for instance by unreliable sensors or imperfect natural language extraction algorithms, or because of human manipulation. Assessing and understanding data veracity is a key requirement when deriving any insights from data sets.

Visualization. Visualization of big data is particularly important for data scientists that try to discover new patterns in the data that can exploited for creating new business value, e.g., by creating new services by combining seemingly unrelated data sets.

2.2 Internet of Things

Over the last decade, there has been a growing research interest in the "Internet of Things" – a disruptive technology, according to the US National Intelligence Council [59]. Despite its recent popularity, the term "Internet of Things" was actually first heard of in the previous century. The original definition envisioned a world where computers would relieve humans of the Sisyphean burden of data entry by automatically recording, storing, and processing in a proper manner all the relevant information about the things involved in human activities [29]. Henceforth, and depending on the viewpoint, different understandings and definitions of what the "Internet of Things" is about have been reported in the literature [2][39][26]. The European Commission envisions it as an integrated part of the Future Internet where "Things having identities and virtual personalities operating in smart spaces using intelligent interfaces to connect and communicate within social, environmental, and user contexts [28]". The use of standard technologies in the World Wide Web to instrument the "Internet of Things" is frequently referred to as the "Web of Things."

Prominent players in all the major ICT markets (i.e., information technology, data networking, telecommunications, etc.) have publicly acknowledged the challenges brought on and the potential entailed by the Internet of Things (IoT) and Machine to Machine Communications (M2M). For instance, NEC expects that the M2M market will expand to approximately JPY 330 billion by 2015, while the Big Data market will expand to JPY 630 billion by 2017 and will exceed JPY 1 trillion by 2020. The 58.5% annual growth observed in 2010 in deployed M2M devices (as quantified via M2M SIM cards) in EU27 stands as market evidence in support of these estimates.

M2M Standardization. Making rapid progress over the last couple of years, the ETSI Technical Committee (TC) on Machine to Machine (M2M) communications has recently published its first version of M2M specifications. The objective is to define the end-to-end system architecture that enables integration of a diverse range of M2M devices (e.g., sensors, actuators, gateways, etc.) into a platform that exposes to applications a standardized interface for accessing and consuming the data and services rendered through these (typically last mile) devices [16]. To this end, ETSI M2M standards define the architecture, interfaces, protocols, and interaction rules that govern the communication between M2M compliant devices.

The M2M logical architecture under development in ETSI comprises two high-level domains:

1. The Network and Application (NA) domain, composed of the following elements:
   - M2M Access Network (AN) providing for communication between the Device Domain and the Core Network (CN).
   - M2M Core Network (CN) providing for IP connectivity and the associated control functions to accommodate roaming and network interconnection.
   - M2M Service Capabilities (SC) providing functions shared by M2M applications by exposing selected infrastructure functionalities through network interfaces while hiding realization details.
   - M2M Applications running the actual application logic.

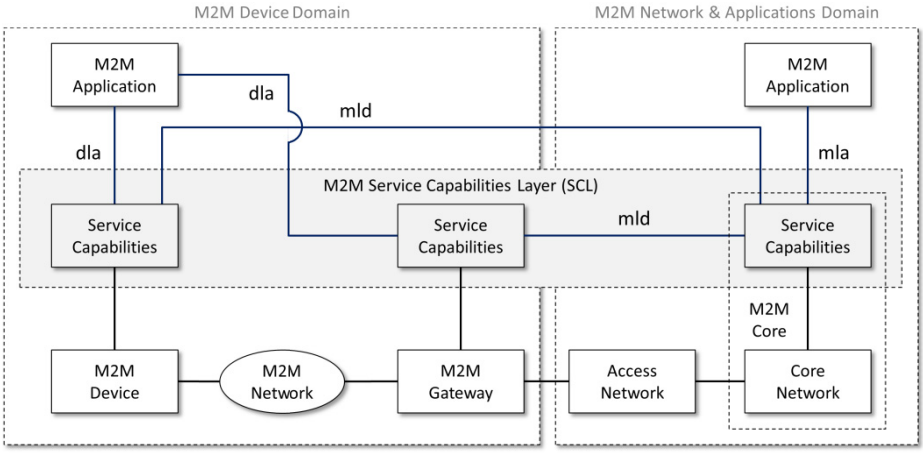


Fig. 1. The ETSI M2M architecture

2. The Device (D) domain, composed of the following elements:
   - M2M Gateway using M2M SC for interconnecting to the NA domain and interworking M2M Devices to it. The M2M Gateway may also run M2M applications.
   - M2M Device that runs M2M applications using M2M SC functions and connecting to the NA domain through any of the following modes:
   - Direct, where the AN provides connectivity.
   - Proxy, where the M2M Gateway provides connectivity by acting as a proxy.
   - M2M Area Network (ArN) interconnecting M2M Devices and M2M Gateways through a field-specific networking technology, e.g., Power Line Communications (PLC), KNX, M-BUS, etceteras.

Management plane functions include network management functions (i.e., functions for managing fault, configuration, accounting, performance, and security aspects) for the AN and CN domains, as well as management functions specific to M2M. An M2M application may use any combination of the Service Capabilities available in the D and NA domains. These Service Capabilities are accessed through the following reference points:

- mIa, for the NA domain, allowing access and use of Service Capabilities therein.
- dIa, for the D domain, allowing an M2M application residing in an M2M host (i.e., Device or Gateway) to access and use different Service Capabilities in the same M2M host. When the M2M host is an M2M Device, access and use of different Service Capabilities in an M2M Gateway is supported also.
- mId, for the communication between M2M Service Capabilities residing in different M2M domains.

Over these references, resource management procedures adopt the RESTful style for the exchange and update of data values on the basis of CRUD (Create, Read, Update, Delete) and NE (Notify, Execute) primitives.

The role of wireless technologies in ETSI M2M is that of connectivity with minimal infrastructure investment both in the Device domain and the Network and Applications domain.

On a global scale, the oneM2M Partnership Project, established by seven of the world's leading information and communications technology (ICT) Standards Development Organizations (SDOs) is chartered to the efficient deployment of M2M systems. To this end, existing ETSI M2M standards are to be transferred to oneM2M and ratified as global M2M standards.

Smart Cities. By amassing large numbers of people, urban environments have long exhibited high population densities and now account for more than 50% of the world's population [58]. With 60% of the world population projected to live in urban cities by 2025, the number of megacities (i.e., cities with a minimum population of 10 million people) is expected to increase also. It is estimated that, by 2023, there will be 30 megacities globally.

Considering that cities currently occupy 2% of global land area, consume 75% of global energy resources and produce 80% of global carbon emissions, the benefit of even marginally better efficiency in their operation will be substantial [58]. For instance, the Confederation of British Industries estimates that the cost of road congestion in the UK is GBP 20 billion (i.e., USD 38 billion) annually. In London alone, introduction of an integrated ICT solution for traffic management resulted in a 20% reduction of street traffic, 150 thousand tons of CO_2 less emissions per year and a 37% acceleration in traffic flow [18].

Being unprecedentedly dense venues for the interactions – economic, social and of other kind – between people, goods, and services, megacities also entail significant challenges. These relate to the efficient use of resources across multiple domains (e.g., energy supply and demand, building and site management, public and private transportation, healthcare, safety, and security, etc.). To address these challenges, a more intelligent approach in managing assets and coordinating the use of resources is envisioned, based on the pervasive embodiment of sensing and actuating technologies throughout the city fabric and supported by ubiquitous communication networks and the ample processing capacity of data centers. The umbrella term Smart City [68] refers to the application of this approach in any of six dimensions:

- Smart economy
- Smart mobility
- Smart environment
- Smart people
- Smart living
- Smart governance

By aggregating data feeds across these domains and applying data processing algorithms to surface the dominant relationships in the data, the situational awareness of

the Smart City at the executive level becomes possible. For instance, by leveraging its open data initiative, the city of London provides a dashboard application demonstrating the kind of high-level oversight achievable by cross-silo data integration and the use of innovative analytic applications [35].

The footprint of our current cities' impact is growing at 8% annually, which means it more than doubles every 10 years. Thus not surprisingly, NIKKEI estimates that USD 3.1 trillion will be invested globally in Smart City projects over the next 20 years [69].

Relationship to Big Data. The popularity of data mashup platforms, as evident today for human-to-machine and machine-to-human information, is expected to extend to machine-to-machine information [16]. Data generated in the context of machine-to-machine communication are typically not constrained by the processing capacities of human entities in terms of volume, velocity, and variety. Particularly in regard to velocity, the ongoing deployment of a large number of smart metering devices and their supporting infrastructures across urban areas increases the percentage of frequently updated small volume data in the overall data set of the Smart City. Thus M2M data exchanges in the context IoT applications for a Smart City impact upon the requirements of data handling through an increase in the volume, variety, and velocity of the data.

Considering the trinity of IoT, M2M, and Smart Cities from the standpoint of cloud technologies, it becomes apparent that the scalability to a large number of M2M devices (i.e., sensors, actuators, gateways) and data measurements will be a prime (non-functional) application requirement. It is, therefore, apparent, that IoT, M2M, and Smart Cities are, from a requirements perspective, right at the core of what Big Data technologies provides.

Increasing urbanization and M2M deployment bring on significant increases in the data generated by IoT applications deployed in the Smart City fabric. For instance, the London Oyster Card data set amounts to 7 million data records per day and a total of 160 million data records per month [4]. Given that a Smart City generates a wide spectrum of data sets of similar – and even larger – size, challenges characteristic of Big Data arise in collecting, processing, and storing Smart City data sets.

3 State of the Art

In this section we describe state of the art of Big Data according focusing on the volume and velocity challenge. For this section, we describe the state of the art mainly from an industrial perspective, i.e., we provide examples of available technologies that represent technologies that can also be used in a productive environment.

3.1 Big Data

As described in the previous section, the advances in IoT infrastructures, combined with the social demands for more efficient resource usage in densely populated urban

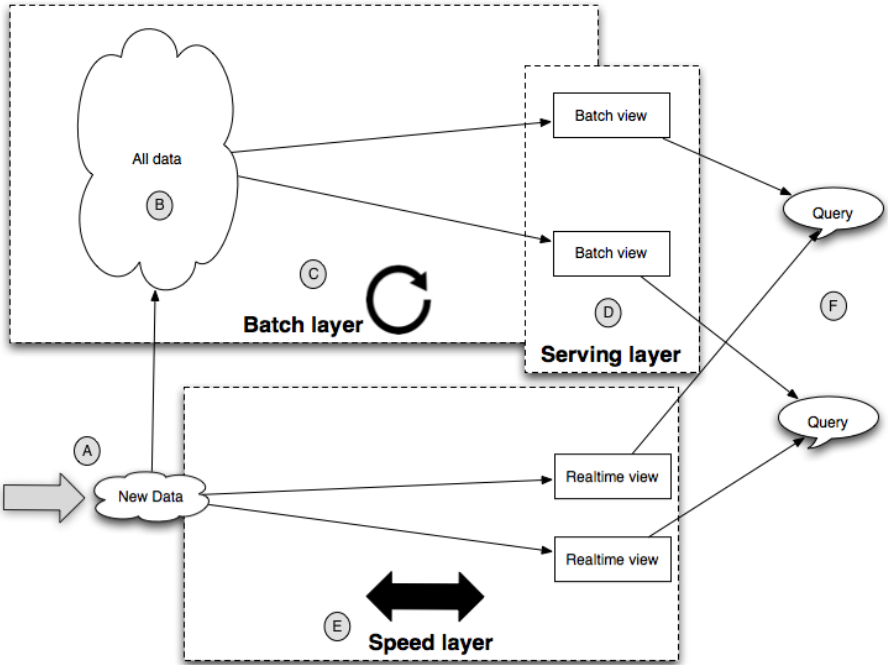


Fig. 2. Lambda Architecture (Source: Big Data – Principles and best practices of scalable realtime data systems, ISBN 9781617290343 [38])

environments, introduces data-related challenges that are at the focus of Big Data toolsets. The latter include technologies and tools that support solving the volume challenge. A range of noSQL databases are able to keep up with high update rates. Programming frameworks such as MapReduce [14] help processing large data in batches. And Stream processing infrastructure such as Storm [56] and S4 [51] provide support for scalable processing of high velocity data. In practice both technologies are required in order to design a low latency query system. Marz and Warren have devised the term Lambda architecture as an architectural pattern that defines the interplay between batch and speed layer in order to provide low latency queries [38].

New data (A) is provided both to the batch and speed layer as depicted in Fig. 2. The batch layer (C) stores all incoming data in its raw format as master data set (B). It is also responsible for running batch jobs that create batch views in a serving layer (D) that is optimized for efficient querying (F). The batch layer is optimized for processing large amount of data, e.g., by using the MapReduce framework and may require hours to process data. Consequently, the batch view will not be updated between repeated executions of the batch jobs and the corresponding data cannot be included in the result set of a query to the serving layer.

The speed layer (E) addresses this information gap by providing a real-time view on the data that has arrived since the last executed batch job. This way an application will always have up-to-date information by querying both the serving and speed layer.

3.2 Batch Processing

In order to cope with the volume challenge so-called noSQL databases are gaining in popularity. Those databases can be classified according to their scalability with respect to data size and suitability to model complex data relationships [43]. With decreasing ability to handle data size and increasing capability to handle complex relationships, we can distinguish between key-value stores, columnar stores, document databases, and graph databases.

Key-value stores are databases that scale easily, but are only able to capture simple relationships, i.e., a key and its value. Columnar stores in contrast store data in columns and are thus better suited for queries that access only small portions of high dimensional data. Document databases are able to store even more complex data, but typically do not scale as well. Graph databases in contrast can easily model any relationship, but do not scale as easily.

The actual processing of data uses different computational frameworks such as Hadoop's MapReduce [63], Hama's Bulk Synchronous Processing framework [53], or graph processing frameworks [24]. They are designed to process large volumes of data in batches, i.e., in regular intervals. As a consequence these technologies are not suitable to process data in real-time.

3.3 Real-Time Analytics

It is commonly acknowledged that, in a lot of economically significant application domains, the value of information decreases as it ages. That is, the more recently the data (and the respective information drawn from it) has been acquired, the more valuable it is. The challenge of processing high velocity data in real-time calls for dedicated solutions that can handle data in motion. Today a number of solutions exist that are designed for executing complex logic over continuous data flows with high performance. These solutions are typically referred to as stream processing or complex event processing systems. The terms stream processing and event processing developed independently and one may argue for some differences in the underlying philosophies. However, the terms are increasingly used interchangeably and the corresponding solutions follow similar principles. We will subsequently only use the term Complex Event Processing (CEP) and subsume stream processing under this term.

CEP engines are designed for implementing logic in the form of queries or rules over continuous data flows. Typically, they include a high level declarative language for the logic definition with explicit support for temporal constructs. The defined logic is executed by the engine using processing techniques that are optimized for continuous data flows. In contrast to batch-driven processing that is triggered on request, CEP engines process incoming data continuously in an event-driven manner. Another difference to batch systems is that CEP engines do not persist information. That is, they operate on temporal windows or synopsis of the incoming data in memory. Consequently the scope of this analytics is live data or data about the recent past as opposed to long-term analysis of recorded information.

The need for CEP technology is rooted in various domains that required fast analysis of incoming information. Examples can be found among others in the finance domain where CEP technologies are used for applications like algorithmic trading or the detection of credit card fraud [27]. In these cases CEP is very well suited due to applications requiring fast analysis of high volume data streams involving temporal patterns. For instance, a credit card fraud may be detected if multiple transactions are executed in short time from far apart locations. Other application domains for CEP include fields like logistics [60], business process management [61], and security [22]. Also, the IoT domain has sparked a range of applications that require event-driven processing and is one of the drivers for CEP technology. Specifically the field of sensor networks has early on let to development of systems that are designed for processing in an event-driven manner (e.g., TinyDB [36], Aurora/Borealis [1]).

Over the last years a number of CEP solutions have emerged in academia as well as in industry. Some of the known early academic projects are TelegraphCQ [8], TinyDB [36], STREAM [41], Aurora/Borealis [1], and Padres [30]. The research initiates were followed by (or directly led to) the emergence of several startups in this domain. For instance the company StreamBase is based on the Aurora/Borealis project and results from the STREAM project fueled the startup Coral8 and the CEP solution of Oracle. Other vendors software vendors like Microsoft and IBM have created CEP solution based on their own internal research projects (CEDR [3], System S [24]). In addition several major vendors have strengthened or established CEP capabilities through acquisitions in recent years. Some examples to name are the acquisition of Apama by Software AG, StreamBase Systems by TIBCO, and Sybase by SAP.

Next to purely commercial offerings the market includes offerings that are available as open source solutions. Example include engines like Esper [19], ruleCore [50], or Siddhi [55]. Noticeable additions to the open source domains are the solutions Storm [56] and S4 [51]. These solutions are not classical CEP engines in the sense that they do not provide a dedicated query language. In contrast, Storm and S4 provide event processing platforms that are focusing on support for distributing of logic to achieve scalability.

4 The Big Smart City Integration Challenges

In this section, we describe in brief the challenges related toward an integrated solution for scalable analysis of Smart City data sources. We consider these challenges mainly from an integration point of view along two dimensions. First there is a question how batch and stream processing should be integrated in a modern Smart City environment (Section 4.1). And second, there is the challenge how the variety of data source should be handled in order to efficiently deliver new services and analyze these data sets as a whole rather than in isolation (Section 4.2). As social media sources provide a potentially rich source of information, we describe them separately (Section 4.3).

4.1 Integration of Batch and Stream Processing

New kind of smart city application will greatly benefit from elaborate analytical algorithms. For instance in order to make prediction about traffic patterns, crime [47], diseases, or energy consumption, it is necessary to first learn patterns from the data, e.g., using statistical or machine learning algorithms. In a second step, the model is applied to new data making decisions based on the model such as the future energy consumption.

As the learned models do not need to be adapted with every incoming data set, state of the art big data batch processing approaches are suitable to learn models on large data sets. Vice versa stream processing approaches are suitable to evaluate new data in real-time.

Note that this is different from Marz' and Warren's main application of the Lambda architecture with applications in mind that require a scalable and robust system design, for querying large amounts of data with low latencies as for instance required for real-time Business Intelligence dashboard visualizations. These kind of applications require essentially the same logic in the batch and stream layer whereas in model-based analysis the logic differs.

The challenge is therefore to devise distributed version of known model learning algorithms and implement them on common distributed processing frameworks such as MapReduce [14]. As with any application of statistical or machine learning model, the main challenge is to find the right features. In particular, it is necessary to define interactions between the batch and speed layers required to access the model. However, the biggest challenge is finding the right combination of technologies that allow for a scalable and robust design. On top of that, the design must ensure that the interactions defined for accessing the model do not run counter to the realization of efficient and scalable processing.

4.2 Integration of Heterogeneous Data Sources

The spectrum of IoT data sources includes sensor data, product databases, or data extracted from the web, including social media. Different data sources model data in different ways and use different protocols and interfaces for communication. To avoid forcing each IoT application to understand a multitude of different data models, including the encoding and semantics of the consumed data as well as the protocol used to access it, the IoT platform must accommodate many different data models by supporting extensions, and a continuum in the evolution of data models and render those to the application in a standard, semantically enriched format. This will also enable better integration between data sources thus facilitating the analysis over disparate data sets.

It is understood that IoT applications vary in terms of the data sets they employ in their function and in the nonfunctional requirements (e.g., reliability, scalability, etc.) imposed upon their operation. Therefore, selecting the solution technology set that renders the intended function while meeting the nonfunctional requirements of the application domain will be an important concern. For instance, some IoT applications

will require real-time processing of frequently updated small volume structured data sets, while other will require complex analytic operations on large volumes of infrequently updated but semantically enriched unstructured data sets. The wide range of operational requirements entailed by this disparity, suggests that the proper instrumentation of the data processing stage will be a paramount concern for IoT applications in Smart Cities. Such instrumentation matters need to be addressed in conjunction with instrumentation options arising from section 4.1 above.

4.3 Natural Text and Social Media Analysis

Nowadays, social networks are widely accessible via mobile devices such as smart phones making them a rich source for monitoring citizen's behaviors and sentiments in real time. Social networks such as Twitter, Facebook, Foursquare, etc. provide access to various location-based information reported by their users, though usually in an unstructured format.

Integrating social sensors data with the IoT could be highly beneficial in several aspects:

1. Contextualization of physical phenomena by providing a subjective context signaling (i.e., explanation) on top of physical events detected by physical sensors
2. Calibration of noisy events detected by physical sensors, by providing an additional supportive signaling of the same events by the social sensors.
3. Detection of 'below the radar' events by combining both subjective and objective sensors data, which otherwise would not have been detected using each separately.

The initial challenge when considering the integration of social networks reports as sensorial insights is the extraction of such signals. Extracted sensorial data might include peoples' stated opinions and statements regarding a particular sentiment, topic of interests, reported facts about one's self (e.g., illness, vacation, event attendance) and various additional subjective reports. Natural language processing (NLP) along with Machine Learning algorithms are applied for extracting the relevant signals from the data posted by the users of the social medium.

An additional challenge is the reliability/credibility of the social sensors. Naturally, the model-based inferred signals are attached with a certainty level produced by the model, where handling the data uncertainty is not trivial. Moreover, the reporting user could also be attached with a credibility score which measures the overall worthiness of considering its data in general.

The sparsity of geo-tagged data in social network data is another challenge when considering its value for integration with the IoT. For example, only 1% of all Twitter messages are explicitly geo-tagged by the users. Recent work suggests several methodologies to overcome this challenge by inferring the users' location based on its context [13] [17].

Finally, there is a need for an architecture that combines both batch and stream processing over social data, for achieving a couple of goals:

1. Enabling the combination of offline-modeling over vast amounts of data and applying the resulting model over streaming data in real time. Most of the complex semantic analysis tasks, as for instance Sentiment Analysis [34] [45] require batch modeling. Feature extraction could be done in real time over a data stream by applying a sliding time window (e.g., 20 seconds) over measures as terms' frequency or TF-IDF [52]. For addressing the challenge of evaluating models in data streams where the data distribution changes constantly, a sliding window kappa-based measure was proposed [6].
2. Analyzing data of all social sensor types, either streaming (e.g., Twitter) or non-streaming (e.g., blog posts) using the same architecture.

5 Case Study: Smart Grid Analytics

In this section, we discuss a case study from the smart grid domain that illustrates the application of big data on smart home sensor data. The case is taken from Peer Energy Cloud (PEC) project [46] that runs a smart grid pilot in a German city. The pilot includes installations of smart home sensors in private homes that measure energy consumption and power quality such as power voltage and frequency at several power outlets in each home. Each sensor takes measurements every two seconds and streams the results into a cloud-based infrastructure that runs analytics for several different use cases. In this section, we discuss the technical details of three scenarios that we implemented in our labs using data from deployed sensors:

1. **Power quality analytics** – shows the benefits of big data batch processing technologies.
2. **Real-time grid monitoring –** shows the benefits of in-stream analytics.
3. **Forecasting energy demand –** shows the need to combine both batch and stream processing.

5.1 Power Quality Analytics

Power quality analytics addresses the identification of problem areas within the distribution grid. The fine-grained sensing allows for detecting power quality issues in the last mile of the grid, down to the household level. For instance, voltage fluctuations can be detected and compared between houses or streets. This enables to pinpoint hotspots of power quality problems and identify root causes. Fig. 3 below shows exemplary the statistics that identify problem hotspots (left) and an analysis of the root cause (right). Spatiotemporal clustering of power quality anomalies can provide operators with insights about the circumstances when power quality issues arise.

Clustering analysis may be further extended adding additional dimensions such as weather conditions, or periodic time attributes like season or time of the day. Together, these analysis support exploratory investigation of root causes and planning of counter measures.

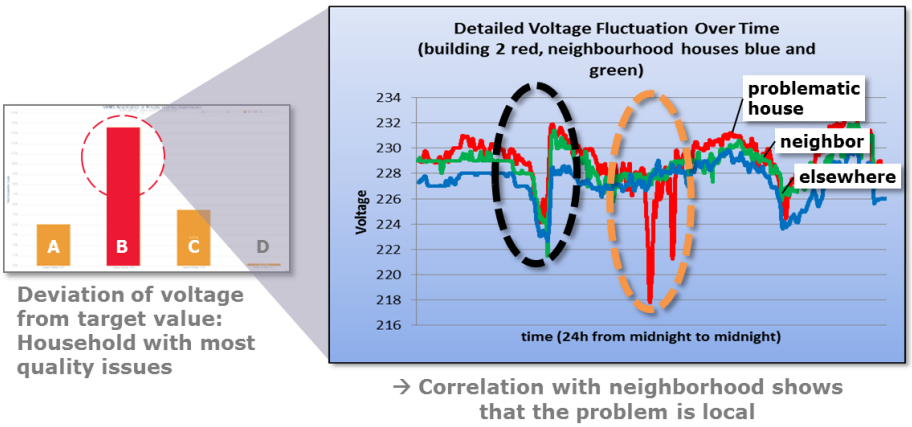


Fig. 3. Analysis of voltage deviations

The big data challenges in this use cases arises due to the high data volumes. Every household produces almost 2 million energy-related measurements a day. This number has to be multiplied by the number of households that use the technology and accumulates over time. For instance, about 200,000 million voltage measurements per month would be available in a full rollout in the small pilot city of Saarlouis.

The query logic for power quality analytics is relatively simple but poses challenges due the high data volumes. Experiments in the pilot project revealed operational challenges already when implementing the analytics for the first four pilot households. Already at this limited scope relational databases required tweaking to handle the queries. However, using the MapReduce programming model and the Hadoop framework it was straightforward to implement the power quality analytics in a scalable way. This is because the underlying analytics problem is of embarrassingly parallel nature and hence very well suited for parallel processing. For instance, it is trivial to partition the data for analyzing household specific voltage fluctuations by household and the Hadoop-based implementation achieved close to linear scalability.

5.2 Real-Time Grid Monitoring

The second use case – real-time grid monitoring – is about providing live insights into the current state of the electrical grid. Industrial control systems typically provide measures of the grid down to the level of secondary substations. With smart homes it now becomes possible to monitor the distribution grid on a level that is not covered by current infrastructures. For instance, power quality measures and the consumed power can be observed for each customer in real-time. This allows detecting critical states and aid responses on a local level. For instance, customers may selectively get demand response signals to temporally adapt their consumption in order to avoid overload situations.

The big data challenges in this use cases arises due to the high data volumes and velocity of the data. In the PEC pilot, every household produces about 18 sensor

measurements per second. A full rollout in the pilot city would result in about 360,000 new measurements every second. Continuously inferring an accurate live state of the grid poses challenges to the throughput and latency of the analytics system. The addressed analytics for power quality and consumption analysis are characterized by incremental updates as well as temporal aggregates. Specifically for such a setting, stream processing and CEP technologies provide an answer to these challenges. Inferring the live state only requires computations over latest sensor information. Thus, the state that is needed for processing is relatively small. CEP engines keep the state in memory and thereby enable high throughput. We found that CEP engines are suitable to (a) support the query logic for live analysis power quality measures and power consumption and (b) to provide the required throughput. Using the open source CEP engine Esper [19] we could run the required analytics for thousands of households in parallel on a single machine. The performance depends on implementation details of the specific analysis. However, the processing paradigm of CEP significantly eased the development of high throughput analytics over the pilot data.

5.3 Forecasting Energy Demand

The third use case – forecasting energy demand – is an important element in demand side management solutions that are under investigation within the PEC pilot. Demand side management takes the approach to balance the grid, not only by adapting the production but also the consumption. Being able to predict the consumption on a household level allows taking proactive measures to influence demand, e.g., by sending demand response signals that ask consumers to reduce load [44]. In the context of the PEC project we are investigating mechanisms to continuously predict load on household level. The underlying concept is to (a) build a prediction model based on recorded sensor data and (b) to apply the model in real-time based on using the latest sensor measurements.

The big data challenge in this use case is twofold. The first challenge arises for building prediction models based on high volumes of sensor records. The second challenge is to apply these models in the stream, using high velocity sensor measurements.

To build prediction models, a large spectrum of candidate algorithms may be applied and tuned to predict household-specific electricity demand. For instance, in [66] we used support vector machines and neural networks. In experiments we observed error reductions in load predictions between compared to persistence predictors of up to 33% (see [66] for details). A straightforward way to implement the model learning in a scalable way is to scale out by partitioning data and processing along households. This approach works to learn arbitrary prediction models in batch mode. To apply the model in real-time, one needs to continuously extract the model features from the incoming energy measurements and call the learned models. Partitioning can be done along households and supported by frameworks for stream processing. In [65] we describe an instantiation of the concept based on a combination of Esper [19] and Weka [62]. With this approach we are able to make low latency real-time forecast for 1000 households on a single machine (see [65] for details).

Suitable technologies exist for the challenges of model learning as well as real-time application of the prediction models. However, no off-the-self solutions to our knowledge directly meet the twofold challenges of this use case. Instead, a combination of big data technologies in required. We discuss such a combination in the following section.

5.4 Lessons Learned

Throughout the pilot project PEC we are gaining first-hand experience into operational aspects of IoT applications with big data. These experiences underpin many of the considerations described in the preceding sections. Specifically, we discuss (1) experience with practical use of Hadoop, (2) implications of using relational schemas, (3) operational challenges in an uncontrolled environment for sensor deployment, (4) and the challenge of combining batch and stream processing.

Using Hadoop Eased Development. By using the Hadoop framework, we found that the benefits of out-of-the box scalability materialized very early in the project. Even for rather simple analytics and after only a few month of data collection the development team struggled to make the corresponding queries scale sufficiently on relational databases. However, using Hadoop it was straightforward to achieve sufficient scalability and performance, without the need tune the implementation. This does not mean that solutions based on relational databases could not have achieved the required performance and scalability. Yet, the burden for the development teams was significantly lower using Hadoop.

Denormalization Helped with Operational Challenges. Regarding (2), the use of relational schemas, we found that denormalization helps with several operational challenges when handling time series of sensor data. Storing each sensors value with all related metadata (e.g., deployment location) makes it trivial to keep the metadata consistent with the measurement. This is especially helpful in an evolving system environment. Throughout the project we found that initially assumed functional dependencies between sensors and metadata entries did not hold anymore as the system use cases expanded. For instance, we initially assumed that sensor deployments reside within one household. However, the need to move the same hardware to multiple locations arose later in the project. By storing the metadata along with the sensor values, it is trivial to ensure that each sensor recording can always be analyzed in the context of the information data was correct during measurement time.

Uncontrolled Environments Require Robust Analytics. Regarding (3), operational challenges for sensor deployments in an uncontrolled environment, we found that this factor has major implications on developing analytics. The part of the system that is outside the realm of control is naturally exposed to unavoidable distortions and externally induced disruptions. This challenge arises in most IoT scenario and is therefore typical for this domain. In the PEC project the sensors are deployed in private households. Distortions due to power outages, accidental disconnection of the sensors, or

simply mishandling of the system are among incidents that must be expected. Therefore, analytics solution must cope with some degree of errors and uncertainty in the input data.

Missing Best Practices for Combining Batch and Stream Processing. Regarding (4), the challenge of combining batch and stream processing, we found that the application of existing big data technologies is straightforward when doing batch and stream processing in isolation. Both worlds have matured tool chains that work to a large degree out of the box. However, the design space for combining batch and stream processing is more open and best practices are less explored. While adapters for data exchange exist, the details of the interplay between batch and stream processing leaves significant effort to development. For instance, we found that the need for batch driven model learning and stream driven application of the models reoccurs in many use cases. However, the most suitable technologies for these tasks do not provide off-the-shelf support for this integrated scenario.

6 Unified Big Data Processing

As a first step toward addressing the Volume and Velocity challenge in the context of Smart Cities, we devised an Analytical Stream Processing framework for handling large quantities of IoT-related data. It extends the basic Lambda architecture by supporting statistical- and machine-based model learning in batches and its use in the streaming layer.

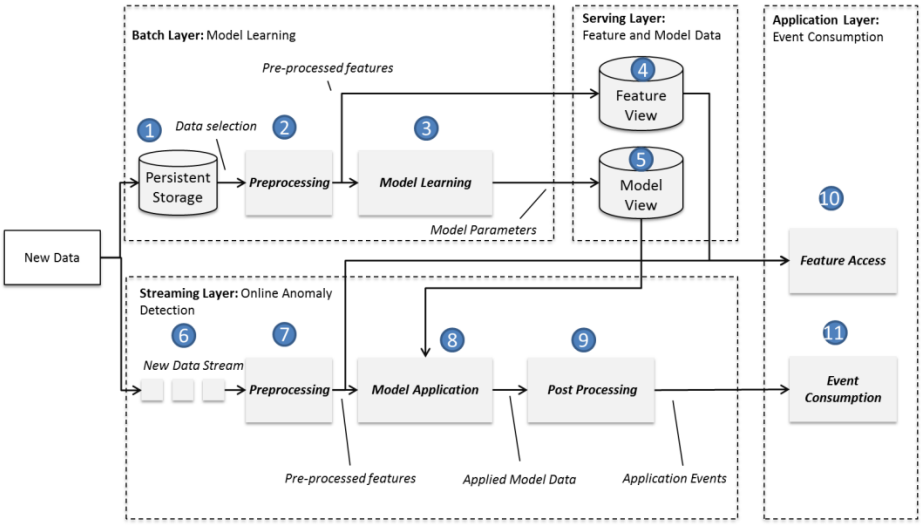


Fig. 4. Big Data Analytical Stream Processing Framework

Fig. 4 shows an initial draft of our framework. New data is being dispatched both to batch layer and stream processing layer. The main responsibility of the batch layer is to calculate models that characterize the incoming data, e.g., by describing re-occurring patterns creating a prediction model. For instance, the batch layer can realize the learning of models for households specific load prediction in the PEC project. The model is provided in the serving layer. Further to the basic lambda architecture, it is important to note that the serving layer must also serve the streaming layer that needs the model data in order to provide its analytical results to the application layer. Online load prediction in the PEC project is an example for such a situation. Here the speed layer extracts features (i.e., temporal aggregates) from the incoming load measurements and calls the previously learned prediction models to obtain a prediction value.

6.1 Model Learning in the Batch Layer

In the batch layer new data is collected in persistent storage such as the Hadoop File System (HDFS) or a NoSQL database (1). The model learning process consists of two main steps as commonly used by machine learning algorithms. They are executed in regular time intervals in order to adapt to changing patterns in the data. These three steps are preprocessing (2) and model learning (3). We briefly describe these processing steps and outline how they can be applied to large data sets.

PreProcessing. In the pre-processing step raw data is processed in order to obtain a list of features that is suitable for applying the respective model learning algorithms. For raw sensor data this typically includes further substages such as a sampling stage, data cleaning, feature extraction, and noise filtering. The sampling stage outputs a list of equally sampled measurements. This is particularly important for event-based sources. Data cleaning may include removal or substitution of erroneous sensor data. The feature extraction stage outputs a time series of one or multiple features that provide a suitable input to the model learning step. A feature can be any value that is calculated out of the raw sensor measurements including the raw measurement itself or applying a function over several types of raw measurements. Finally, the noise filtering stage smoothens the extracted features, e.g., by applying a moving average or median filter.

Model Learning. In the model learning step the extracted features are used to calculate the actual model. The model parameters representing the model depend on the used algorithms. In our use case above this would include temporal aggregates of household- and device-specific load measurements. For a statistical model the model parameters would represent the parameters of a statistical distribution function such as the mean and standard deviation for a normal distribution. The model would also include thresholds on the variance based on which an anomaly is considered to be detected.

Scalable Processing. In the context of processing large amounts of data it is important to parallelize the processing steps discussed above. The realization of the parallelization depends to a large degree on the actual data and algorithms used. Due to the simplicity of the MapReduce framework it is beneficial to describe the batch layer processing in terms of map and reduce steps.

In order to apply the MapReduce framework to a model learning problem, we first need to consider whether the problem is embarrassingly parallel, i.e., whether the input data can be split into independent parts for which the algorithms can be applied. If there is a large number of spatial or temporal groups of data records for which a model needs to be learned, applying the MapReduce framework is an obvious procedure. For instance for the load prediction use case discussed above, we can create our load model for each household independently mapping individual measurement to a household and calculating the model for each household in the reduce step.

In other cases it may be necessary to rewrite the algorithm in a distributed way. Chu et al. describe how a special class of machine learning algorithms can be rewritten so that their execution can be sped up by using the MapReduce framework [9]. For more complex problems, it may be necessary to redesign the algorithm, potentially sacrificing optimal solutions, or use other parallel programming frameworks such as Apache Hama [53] that provides an implementation of Bulk Synchronous Processing technique.

6.2 Viewing Data Patterns in the Serving Layer

Similar to the basic Lambda architecture the serving layer provides a view of the output data generated by the batch jobs. In the context of our analytical streaming framework such batch views includes a feature view (4) that contains the preprocessed feature values prepared for fast access by applications.

The model view (5) is an important specialization of the serving layer. It contains the model parameters as calculated in the model learning step. The size of the model view is typically considerably smaller than the original data sets. For instance, the size of a statistical model that characterizes load distributions of individual households aggregates data over the period that is used for model learning.

In contrast to the basic Lambda architecture, the model views are primarily used by the speed layer and not the application layer. Different design options exist for this integration. Two fundamental options are to (a) leave the model in the serving layer and (b) to load the model into the speed layer. In both options the speed layer extracts model inputs from the live data stream and calls the model. In option (a), the speed layer sends the extracted model input to the serving layer and gets the result of the model application in return. This option has the advantage that the model must not be managed in the memory of the speed layer. The drawback is that calls to the serving layer may increase latency and reduce throughput. In option (b), the model is loaded into the memory of the speed layer one the corresponding streaming logic is instantiated this option has the advantage that the model application avoids the overhead of making external calls but – dependent on the model size – can also cause resource problems with respect to the main memory.

6.3 In-Stream Analytics in the Streaming Layer

The streaming layer receives the same stream of new data (6) as the batch layer. Its main purpose is to analyze the incoming data stream in real-time. As a first step the data is preprocessed (7) in order to receive the features out of the raw data stream. This calculation is functionally equivalent to the preprocessing step in the batch layer (2). Thus it may be beneficial to reuse the logic in the batch layer, in particular if complex processing over time series data is being performed. In this case, the query languages offered by complex event processing engines may often be better suited to formulate the logic than a hand-crafted code or even SQL [20]. This can for instance be achieved by invoking the CEP queries in the reduce step of a MapReduce job [40].

In the next step the model in the serving layer is accessed in order to apply the model on the preprocessed features calculated in the previous step (8). In the case of statistical anomaly detection, this could for instance involve computing the deviation from the expected value as stored in the model for the matching time period and throwing an event if the threshold is exceeded. For instance, to detect power quality anomalies one may compare current voltage fluctuations against a statistical model of typically observed fluctuations.

As the processing in the speed layer is often time critical, it may be necessary to avoid further latencies introduced by accessing the stored model in the serving layer. This could for instance be achieved by providing an event source in the streaming layer that accesses the corresponding parts of the model or directly loading the model into the CEP engine and manually updating it. However, if the model is large and requires complex update strategy it is more beneficial to query an in memory database, e.g., using Redis [48].

The final step in the speed layer is the postprocessing of applied model data (8). This step may be necessary to reduce the number of false positives. In the case of anomaly detection for instance, it is often not desired to propagate a singular anomaly event that may result from measurement errors or other inaccuracies in the data or model. Thus, an anomaly may only be indicated if it is consecutively detected over a certain time.

Scalable Processing. In order to cope with a large number of events it is necessary to scale out the processing logic to multiple machines. This can for instance be achieved by creating a Storm [56] topology that defines how stream-based data is being processed. The partitioning of data can then be defined in a similar was in the batch processing layer. For instance if the data can easily spatially be separated it is possible to use Storm's field grouping capability to ensure that data of a spatial partition is always processed by the same task. The processing logic itself can then be implemented using modern complex event processing engines such as Esper [19] benefiting from their performance and query languages.

6.4 Accessing the Data in the Application Layer

As in the basic Lambda architecture the application layer is accessing the data both from the serving and speed layer. The application uses the data from the batch and streaming layer in two ways. First, the streaming data can be used to provide a real

time view of the features calculated in the batch layer (10). As in the basic lambda architecture this can be useful to provide a real-time dashboard view, for instance reflecting the current state of the power grid. Second, the application may only consume events generated in the stream layer that are based on applying incoming data to the model as described above (11). This way applications and human operators can receive events about detected anomalies or continuous predictions about energy consumption.

7 Further Research Directions

For further work we plan to extend our analytical framework and apply it to other domains in the context of Smart Cities. This includes in particular adding more machine learning algorithms for the batch layer and the corresponding logic for the streaming layer.

In this chapter we have focused on an analytical framework for processing large volumes of data in real-time, i.e., we addressed mainly the volume and velocity challenge. As we extend our work to different data sets and application domains, it will, however, be increasingly important to cope with the variety of data sources and their data.

It is, therefore, a key requirement for the proposed analytical framework that both the model learning algorithms as well as the stream logic can be applied uniformly across different data sets and application domains. On one hand this will maximize the value that can be extracted from available data sets and on the other hand the processing chain can then easily be applied to new data sets, thus saving effort, time, and costs during the development process.

A future research direction is therefore to extend the analytical framework with the necessary mechanisms to achieve such uniform processing. This could for instance be realized by a metadata model on which the corresponding logic operates. As we see the application of this analytical framework mainly in the context of Smart Cities and the Internet of Things, an entity-based framework that naturally models real-world entities such as sensors, people, buildings, etc. [23]. Such an information model along with a corresponding architecture has been defined by the IoT-A project [5].

8 Conclusions

In this chapter, we have proposed an initial draft of a Big Data analytical framework for IoT and Smart City applications. The framework is based on existing state of the art, initial findings from our participation in the publicly funded projects BIG [7] and PEC [46] as well as our own experiences with analytical applications for Smart Cities. Our work is motivated by the fact that key components such as data, sources, algorithms, IoT architectures, and Big Data technologies are available today, but still there is a lot of effort required to put them into operational value.

A significant part of this effort is due to missing standards (cf. for instance the SQL query language for relational databases) and wide variety of different technologies in

the Big Data domain as well as the required integration effort. But the application of advanced analytical computation at scale and speed also requires considerable design effort and experience. We believe that an analytical Big Data framework along with appropriate toolboxes can add significant value both to required development effort and insights that can be derived from the data. While there are Big Data machine learning libraries such as Mahout [42], as well as frameworks for model learning on top of Hadoop [48], we are not aware of fully integrated analytical frameworks that combine model learning and stream processing.

In order to fully benefit from the framework its overall design and associated toolboxes need to support a variety of data sources and algorithms. While this requirement does not change the high level architecture of the framework, such extensions do have significant impact on the interface level and overall design of the individual processing components. A carefully planned and sound conceptual design as well as pragmatic implementation decisions will be an important enabler to reduce development costs and create innovative services in the IoT and Smart City domain.

Acknowledgments. This work has partly been funded by the EU funded project Big Data Public Private Forum (BIG), grant agreement number 318062, and by the Peer Energy Cloud project which is part of the Trusted Cloud Program funded by the German Federal Ministry of Economics and Technology. We would also like to thank Max Walther, who implemented the batch-driven power quality analytics MapReduce jobs as well as Alexander Bauer and Melanie Hartmann who supported us with the design of the analytical models.

References

1. Abadi, D.J., et al.: The Design of the Borealis Stream Processing Engine. In: CIDR, vol. 5, pp. 277–289 (2005)
2. Atzori, L., Iera, A., Morabito, G.: The internet of things: A survey. Computer Networks 54(15), 2787–2805 (2010)
3. Barga, R.S., et al.: Consistent streaming through time: A vision for event stream processing. arXiv preprint cs/0612115 (2006)
4. Batty, M.: Smart Cities and Big Data, http://www.spatialcomplexity.info/
5. Bauer, M., Bui, N., Giacomin, P., Gruschka, N., Haller, S., Ho, E., Kernchen, R., Lischka, M., Loof, J.D., Magerkurth, C., Meissner, S., Meyer, S., Nettsträter, A., Lacalle, F.O., Segura, A.S., Serbanati, A., Strohbach, M., Toubiana, V., Walewski, J.W.: IoT-A Project Deliverable D1.2 – Initial Architectural Reference Model for IoT (2011), http://www.iot-a.eu/public/public-documents/d1.2/view (last accessed September 18, 2013)
6. Bifet, A., Frank, E.: Sentiment knowledge discovery in twitter streaming data. In: Pfahringer, B., Holmes, G., Hoffmann, A. (eds.) DS 2010. LNCS (LNAI), vol. 6332, pp. 1–15. Springer, Heidelberg (2010)
7. BIG Project Website, http://www.big-project.eu/ (last accessed September 19, 2013)
8. Chandrasekaran, S., et al.: TelegraphCQ: continuous dataflow processing. In: ACM SIGMOD International Conference on Management of Data, pp. 668–668. ACM (2003)

9. Chu, C.-T., Kim, S.K., Lin, Y.A., Yu, Y.Y., Bradski, G., Ng, A.Y., Olukotun, K.: Map-Reduce for Machine Learning on Multicore. In: Schölkopf, B., Platt, J.C., Hoffman, T. (eds.) Advances in Neural Information Processing Systems 19 (NIPS 2006), pp. 281–288. MIT Press, Cambridge (2007)
10. Correia, Z.P.: Toward a Stakeholder Model for the Co-Production of the Public Sector Information System. Information Research 10(3), paper 228 (2005), http://InformationR.net/ir/10-3/paper228.html (last accessed February 27, 2013)
11. DataMarket, http://datamarket.com/ (last accessed September 21, 2013)
12. Data.gov, http://www.data.gov/ (last accessed September 21, 2013)
13. Davis, J.R., Clodoveu, A., et al.: Inferring the Location of Twitter Messages based on User Relationships. Transactions in GIS 15(6), 735–751 (2011)
14. Dean, J., Ghemawat, S.: MapReduce: Simplified Data Processing on Large Clusters. Communications of the ACM 51(1), 1–13 (2008), doi:10.1145/1327452.1327492
15. Directive 2003/98/EC of the European Parliament and of the Council of 17 November 2003 on the re-use of public sector information, http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CONSLEG:2003L0098:20130717:EN:PDF (last accessed January 13, 2014)
16. Dohler, M.: Machine-to-Machine Technologies, Applications & Markets. In: 27th IEEE International Conference on Advanced Information Networking and Applications (AINA) (2013)
17. Dredze, M., Paul, M.J., Bergsma, S., Tran, H.: Carmen: A Twitter Geolocation System with Applications to Public Health (2013)
18. The Economist, Running out of road (November 2006)
19. EsperTech, http://esper.codehaus.org (last accessed September 22, 2013)
20. Etzion, O.: On Off-Line Event Processing. Event Processing Thinking Online Blog (2009), http://epthinking.blogspot.de/2009/02/on-off-line-event-processing.html (last accessed September 17, 2013)
21. European Open Data Portal, http://open-data.europa.eu/ (last accessed September 21, 2013)
22. Farroukh, A., Sadoghi, M., Jacobsen, H.-A.: Towards vulnerability-based intrusion detection with event processing. In: 5th ACM International Conference on Distributed Event-based System, pp. 171–182. ACM (2011)
23. Gazis, V., Strohbach, M., Akiva, N., Walther, M.: A Unified View on Data Path Aspects for Sensing Applications at a Smart City Scale. In: IEEE 27th International Conference onAdvanced Information Networking and Applications Workshops (WAINA 2013), pp. 1283–1288. IEEE Computer Society, Barcelona (2013), doi:10.1109/WAINA.2013.66
24. Gedik, B., Andrade, H., Wu, K.L., Yu, P.S., Doo, M.: SPADE: the system s declarative stream processing engine. In: ACM SIGMOD International Conference on Management of Data, pp. 1123–1134. ACM (2008)
25. Giraph Project, http://giraph.apache.org/ (last accessed September 21, 2013)
26. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems (2013)
27. Hinze, A., Sachs, K., Buchmann, A.: Event-based applications and enabling technologies. In: Third ACM International Conference on Distributed Event-Based Systems (2009)
28. INFSO D.4 Networked Enterprise & RFID INFSO G.2 Micro & Nanosystems, Internet of Things in 2020 – A roadmap for the Future (September 2008), report available at http://www.smart-systems-integration.org/public/internet-of-things
29. ITU, The Internet of Things (2005)

30. Fidler, E., Jacobsen, H.A., Li, G., Mankovski, S.: The PADRES Distributed Publish/Subscribe System. In: FIW, pp. 12–30 (2005)
31. van Kasteren, T., Ravkin, H., Strohbach, M., Lischka, M., Tinte, M., Pariente, T., Becker, T., Ngonga, A., Lyko, K., Hellmann, S., Morsey, M., Frischmuth, P., Ermilov, I., Martin, M., Zaveri, A., Capadisli, S., Curry, E., Freitas, A., Rakhmawati, N.A., Ul Hassan, U., Iqbal, A.: BIG Project Deliverable D2.2.1 – First Draft of Technical White Papers (2013), http://big-project.eu/deliverables (last accessed September 19, 2013)
32. Laney, D. 3D Data Management: Controlling Data Volume, Velocity and Variety. Meta Group Research Report (2001), http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf (last accessed September 21, 2013)
33. Leeds, D.J.: THE SOFT GRID 2013-2020:Big Data & Utility Analytics for Smart Grid. GTM Research Report (2012), http://www.greentechmedia.com/research/report/the-soft-grid-2013 (last accessed September 21, 2013)
34. Liu, B.: Sentiment analysis and opinion mining. Synthesis Lectures on Human Language Technologies 5(1), 1–167 (2012)
35. The London Dahsboard, http://data.london.gov.uk/london-dashboard (last accessed September 21, 2013)
36. Madden, S.R., Franklin, M.J., Hellerstein, J.M., Hong, W.: TinyDB: An acquisitional query processing system for sensor networks. ACM Transactions on Database Systems 30, 122–173 (2005)
37. Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., Hung Byers, A.: Big data: The next frontier for innovation, competition, and productivity. McKinsey Global Institute (2013), http://www.mckinsey.com/insights/business_technology/big_data_the_next_frontier_for_innovation (last accessed August 20, 2013)
38. Marz, N., Warren, J.: A new paradigm for Big Data. In: Big Data – Principles and Best Practices of Scalable Real-time Data Systems, ch. 1, Manning Publications Co. (to appear), http://www.manning.com/marz/ (last accessed August 16, 2013), ISBN 9781617290343
39. Miorandi, S., Sicari, F., Pellegrini, D., Chlamtac, I.: Internet of things: Vision, applications and research challenges. Ad Hoc Networks 10(7), 1497–1516 (2012)
40. Microsoft BI Team, Big Data, Hadoop and StreamInsight™, http://blogs.msdn.com/b/microsoft_business_intelligence1/archive/2012/02/22/big-data-hadoop-and-streaminsight.aspx (last accessed September 09, 2013)
41. Rajeev, M., Widom, J., Arasu, A., Babcock, B., Babu, S., Datar, M., Manku, G., Olston, C., Rosenstein, J., Varma, R.: Query processing, approximation, and resource management in a data stream management system. In: CIDR Conference, pp. 1–16 (2002)
42. Owen, S., Anil, R., Dunning, T., Friedman, E.: Mahout in Action. Manning Publications Co. (2011) ISBN 9781935182689
43. Neubauer, P.: Neo4j and some graph problems, http://www.slideshare.net/peterneubauer/neo4j-5-cool-graph-examples-4473985?from_search=2 (last accessed September 21, 2013)
44. Palensky, P., Dietrich, D.: Demand side management: Demand response, intelligent energy systems, and smart loads. IEEE Transactions on Industrial Informatics 7(3), 381–388 (2011)
45. Pang, B., Lee, L.: Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval 2(1-2), 1–135 (2008)

46. Peer Energy Cloud project website, http://www.peerenergycloud.de/ (last accessed September 19, 2013)
47. PredPol, http://www.predpol.com/ (September 21, 2013)
48. Radoop, http://www.radoop.eu/ (last accessed December 16, 2013)
49. Redis Project, http://redis.io/topics/faq (last accessed September 17, 2013)
50. ruleCore, http://www.rulecore.com (last accessed September 22, 2013)
51. S4 Project, http://incubator.apache.org/s4/ (last accessed August 16, 2013)
52. Salton, G., Buckley, C.: Term-Weighting Approaches in Automatic Text Retrieval. Information Processing and Management 24(5), 513–523 (1988)
53. Seo, S., Yoon, E.J., Kim, J., Jin, S., Kim, J.-S., Maeng, S.: HAMA: An Efficient Matrix Computation with the MapReduce Framework. In: 2nd IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2010), pp. 721–726. IEEE Computer Society (2013)
54. Shigeru, O.: M2M and Big Data to Realize the Smart City. NEC Technical Journal 7(2) (2012)
55. Siddhi CEP - The Complex Event Processing Engine, http://siddhi.sourceforge.net (last accessed September 22, 2013)
56. Storm Project, http://storm-project.net/ (last accessed August 16, 2013)
57. Stonebraker, M.: What Does 'Big Data' Mean? (Part 3). BLOG@ACM (2012), http://cacm.acm.org/blogs/blog-cacm/157589-what-does-big-data-mean-part-3/fulltext (last accessed August 16, 2013)
58. United Nations, World Urbanization Prospects 2011 Revision (2011)
59. US National Intelligence Council.: Disruptive Civil Technologies: Six Technologies with Potential Impacts on US Interests out to 2025, http://www.fas.org/irp/nic/disruptive.pdf (last accessed December 20, 2013)
60. Wang, F.-s., Liu, S., Liu, P., Bai, Y.: Bridging physical and virtual worlds: complex event processing for RFID data streams. In: Ioannidis, Y., et al. (eds.) EDBT 2006. LNCS, vol. 3896, pp. 588–607. Springer, Heidelberg (2006)
61. Weidlich, M., Ziekow, H., Mendling, J., Günther, O., Weske, M., Desai, N.: Event-based monitoring of process execution violations. In: Rinderle-Ma, S., Toumani, F., Wolf, K. (eds.) BPM 2011. LNCS, vol. 6896, pp. 182–198. Springer, Heidelberg (2011)
62. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update. SIGKDD Explorations 11, 10–18 (2009)
63. White, T.: Hadoop: The Definitive Guide. O'Reilly (2012)
64. Xively, https://xively.com/ (last accessed September 21, 2013)
65. Ziekow, H., Doblander, C., Goebel, C., Jacobsen, H.-A.: Forecasting Household Electricity Demand with Complex Event Processing: Insights from a Prototypical Solution. In: Middleware Conference, Beijing, China (2013)
66. Ziekow, H., Goebel, C., Strüker, J., Jacobsen, H.-A.: The Potential of Smart Home Sensors in Forecasting Household Electricity Demand. In: IEEE International Conference on Smart Grid Communications (SmartGridComm 2013), Vancouver, Canada (2013)
67. Zillner, S., Rusitschka, S., Munné, R., Lippell, H., Lobillo Vilela, F., Hussain, K., Becker, T., Jung, R., Paradowski, D., Huang, Y.: BIG Project Deliverable D2.3.1 – First Draft of Sector's Requisites (2013), http://big-project.eu/deliverables (last accessed September 19, 2013)
68. European Smart Cities project, http://www.smart-cities.eu/model.html
69. Smart City Week 2012 international conference and exhibition report, October 29-November 2 (2012), http://scw.nikkeibp.co.jp/2013/docs/SCW2012_Conference_Report_vol3.pdf

How the Big Data Is Leading the Evolution of ICT Technologies and Processes

Antonio Scarfò[1] and Francesco Palmieri[2]

[1] MaticMind SpA,
CDN Isola F4, Naples, Italy
ascarfo@maticmind.it
[2] Second University of Naples, Dept. of Industrial and Information Engineering,
Via Roma 29, I-81031, Aversa (CE), Italy
francesco.palmieri@unina.it

Abstract. This chapter has the main aim of providing an overview of the evolution process related to big data and its impact on the organization of ICT-related companies and enterprises. It starts from the severe scalability limits and performance issues introduced by the need of accessing massive amounts of distributed information, by highlighting the most important innovation trends, and developments characterizing this new architectural scenario both from the technological and the organizational perspectives. By trying to address the missing links in the ICT big picture, we also present the emerging data-driven reference models and solutions in order to give a clearer vision of the near future in the modern information-empowered society, where all the activities are more and more frequently conducted in very large collaborative partnerships involving multiple people and equipment scattered throughout the world.

Keywords: Big Data, Analytics, Modeling.

1 Introduction

Nowadays, the proliferation of the data sources available on the Internet and the widespread deployment of network-based applications are fostering the emergence of new architectures referred as "big data", characterized by the need of capturing, combining, and processing an always growing amount of heterogeneous and unstructured data coming from new mobile devices, emerging social media, and human/machine-to-machine communication. This implies managing data at volumes and rates that push the frontiers of current archival and processing technologies. For this reason, such architectures usually require the orchestrated usage of geographically sparse resources to satisfy their immense computational and storage requirements. Consequently, the distributed nature of these resources makes remote data access and movement, often involving petabytes of data, the major performance bottleneck for the involved end-to-end applications. However, big data processing architectures are now widely recognized as one of the most significant ICT innovations in the last decade, since they can

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_12

bring extraordinary and, maybe today unimaginable, opportunities to improve the social conditions as well as the quality of life (e.g., in mobility, healthcare etc.), the efficiency of resources utilization, the technology evolutions, and so on. These opportunities are tightly related to the challenges that technologies and organizations have to face in order to adopt new evolution models based on the big data processing paradigm. First of all, organizations should rethink their own structure, in terms of processes and workflows, as well as the management of their data assets and of the whole information lifecycle, in order to deal effectively with the data deluge facing modern ICT world, by embracing such a new evolution model led by the availability of massive information. The most critical issues in such reorganization process are associated to the definition of the right technologies, data sets, data interpretation, and timing to support decision making. The technologies currently adopted to support the *Information Life cycle Management* (ILM) are not appropriate to meet big data requirements. The legacy storage systems and the legacy database architectures do not ensure the efficiency, the performances, and the scalability required by the new evolution models and trends driven by big data architectures. In addition, the legacy front-end platforms are not still able to provide the enhanced analysis and the visualization features that would make big data helpful for modern organizations. Accordingly, new architectures and technologies are emerging, whose goal is to address the limitations characterizing the legacy ones in dealing with the big data scenario.

2 What Is the Big Data

Big data is usually defined as a huge collection of unstructured information stored in multiple data sets that due to their size and complexity cannot be processed by using traditional database management systems or data mining/analysis applications. The emerging big data phenomenon is the natural result of more and more digitalized world, where people and machines that produce, use, and share data are steadily increasing. At the base of this global "digitalization" trend there are several factors ranging from the increasing adoption of digital data representation/storage formats as well as the changing way to share and use information through flexible and powerful semantic/social networking organizations, to the astonishing evolution and deployment of the communication devices and their associated high-performance transport infrastructures. In particular, the "always connected" paradigm, fostered by the diffusion of ubiquitous and pervasive communication devices has greatly propelled the above phenomena.

2.1 Telecommunication Services

The evolution of telecommunication services is a fundamental element that contributes to big data. The amount of data shared and produced, is directly related to people reached by high-performance telecommunication services. In addition,

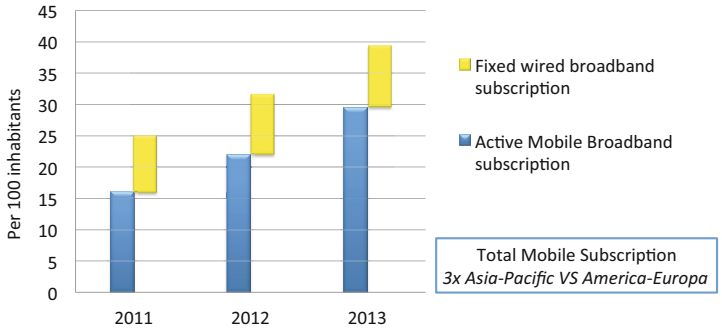


Fig. 1. ITU Global Statistics: telecom services subscription trends. Data from [1].

mobility is another great enabler for both data production and processing demand. This is directly associated to the considerable increase in mobile telecommunication services subscriptions experienced in the last years (see fig. 1). These services, also fostered by network pervasiveness and availability, resulted to be an unbelievable accelerator for data production in almost any IT sector.

2.2 The Social Wave

The constantly evolving social media are one of the most powerful catalysts of unstructured data, mainly produced by mobile devices. The potential of social networks is enormous. Their growth is driven by their widespread use not only in the consumer arena but also in enterprise scenarios, so that they are even now ready for marketing and social communication. In fact the future of unified collaboration technologies is to become social-like.

To give an idea of the amount of data flowing in social media, consider that every minute 100 hours of video are uploaded to YouTube [2], and considering that an hour of standard video takes up about 1GB in average, resulting in a production of 100GB/min, that is something as 50PB/year.

2.3 Science and Research

Science and research are other fields that are contributing significantly to big data. During last years, Research Centers have done large investments in technologies in order to perform experiments and study or simulate physics phenomena. Many scientific activities continuously gather and analyze huge amounts of unstructured data. As an example, we can consider the *Large Hadron Collider* (LHC) experiments, involving about 150 million sensors delivering data at the rate of 40 million samples/second. The information produced by all four main LHC experiments sums to about 700MB/sec (or 20.5 PB/year) before replication on geographically distant processing sites, reaching about 200 petabytes

after replication [3][4]. Whereas LHC is one of the most impressive experiments in term of data produced, there are a lot of initiatives that are sources of huge amount of data, encoding the human genome, or climate simulation or the Sloan Digital Sky Survey, and so on.

2.4 Government and Public Sector

Almost all nations in the world are implementing or are considering about programs focused on big data gathering and utilization. In fact, both public sector and government agencies collect huge amounts of data for various aims. These organizations have been the most active producers of documents to be digitalized and stored on IT supports, primarily in order to limit paper consumption. Accordingly, they are accumulating huge amounts of digital data and now are studying effective ways for analyzing and correlating them. To clearly appreciate the phenomenon, we can think about digital data coming from healthcare, bureaccuracy processes, supervision of districts, and so on. On March 29, 2012, US administration presented the "Big Data Research and Development Initiative": a huge investment program for big data R&D, related to Defense, Health, Energy, and Geological Survey. The basic idea is making available, and hence open, data collected by government to companies, individuals and nonprofit sector, in order to build useful "apps" and services, and promote democracy, participation, transparency, and accountability [5]. Also other initiatives by various governments around the word focus on the utilization of the large and continually growing amount of data produced by the public sector to introduce new benefits in society. In order to give a real example, according to a recent McKinsey report [6], there are three strategic areas in the public sector where big data can create added value for at least 150 Billion Euro in European Community.

2.5 Internet of Things

Machine-to-machine communication is another interesting cross-sources trend for which, again, mobile wireless connectivity is a fundamental enabling element. Modern machines such as cars, trains, power stations, planes, and so on are equipped with an increasing numbers of sensors constantly collecting massive data. Analogously very huge areas are covered by sensor equipment for monitoring/surveillance purposes, as well as many real-world object (aka things) are equipped with RFID devices providing plenty of useful digital information to their users (e.g., books in a library or products in a store). It is common to have thousands or even hundreds of thousands sensors providing information about the performance and activities of a specific machine or site and making them available through the Internet. Analogously, millions of independent RFID-tagged devices may make their data/characteristics available over the network. The connection between things together with their correlation and interworking, while complex, is one of the emerging technologies that have an enormous potential of changing our life. According to [7] at the state of the art 1 trillion of devices are potentially connectable resulting in 100 millions

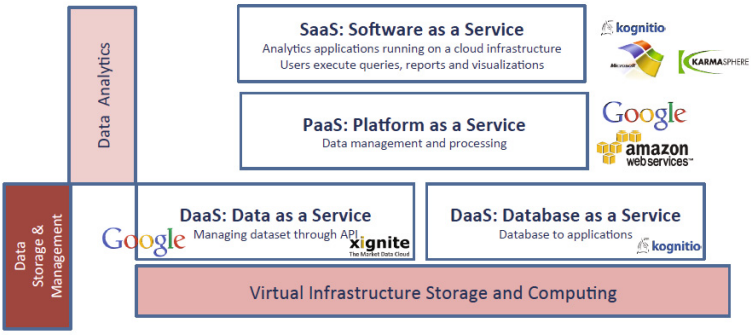


Fig. 2. Big data and cloud services

of cross-sectors machine-to-machine connections. More connections mean more automation and more control, then more process optimization and security. Enterprises are strongly investing in that sector resulting in a substantial growth of the data produced.

2.6 Clouds

Clouds are also strongly involved in big data generation simply because a lot of services relying on unstructured data are delivered in Cloud style. For instance, e-mail, instant messaging, or many social networks rely on cloud infrastructures because they provide the only scalable way allowing such a huge number of end-users to access these services through the web. On the other side there are enterprise applications and services, for which the Cloud paradigm (also in its private cloud vision) is an excellent accelerator because it makes extremely easy and affordable to deploy new large-scale data warehousing architectures, supporting data generation, processing, replication or low-cost storage. The *SaaS (Software as a Service)*, *PaaS (Platform as a Service)* and *DaaS (Data as a service)* cloud service models cover almost all the needs associated to big data processing and make big data opportunities more affordable. In particular DaaS can be seen as a form of managed service, similar to Software as a Service, or Infrastructure as a Service delivering data analysis facilities offered by an outside provider in order to help organizations in understanding and using the insights gained from large datasets with the goal of acquiring a competitive advantage. Big data as a service often relies upon cloud storage (see fig. 2) to provide effective and flexible data access to the organization that owns the information as well as to the provider working with it.

Cloud data services can also be the cheapest way to store data. The chart reported in fig. 3 shows that the really relevant data for organizations objectives may be a small part of the whole amount. Consequently, a tiering strategy based on relevance can be an affordable solution to store a huge amount of data, where the most part of data, characterized by a limited relevance is stored in the cloud.

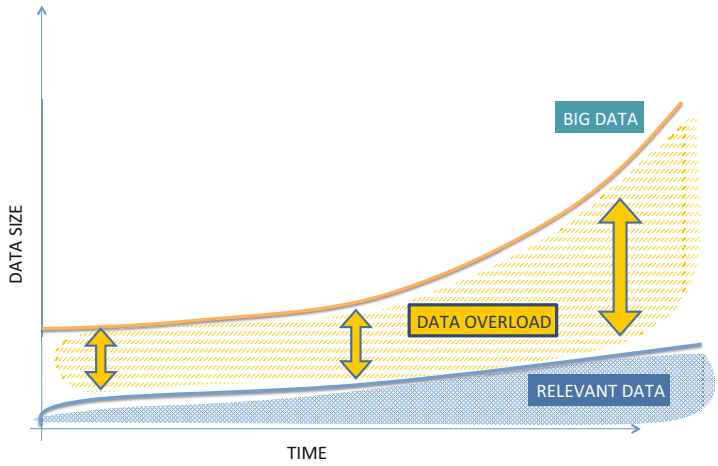


Fig. 3. From big data to valuable relevant data

By considering all the aforementioned big data sources together we can no longer speak of isolated phenomena that concur to big data generation, but of several factors that interact each other by creating a virtuous cycle. In other words, the production of big data is affected by Mobility, Social Networking, and the emergence of Data-Centric Cloud Services. However, all these factors, in turn are influenced by the availability of big data for their evolution and growth dynamics.

3 The Big Data Opportunity

Historically, data is one of the most important assets for firms and governments, and data analysis is the first element needed for creating strategies and measure their effects. So, big data processing and archival technologies may bring great opportunities with them, in order to improve performance and competitiveness of modern organizations, and resulting in significant benefits for the overall society. There is a significant difference between data and big data beyond the pure dimensional factors. Exploitation of big data potentialities creates the opportunities for a real "quantum jump" or a "sharp transition", that is, a significant and unusual level of improvement, in the traditional IT scenario. Mainly, it changes the way we manage and share our data by making available a huge amount of heterogeneous information coming from a wide variety of sources that can be correlated each other in order to create new added value.

3.1 Bringing Value

Several factors may concur in bringing value from big data:

- *Transparency*: a great value for communities comes from just making big data easily accessible through the most widespread media, to relevant stakeholders in useful time. For the Public Sector, this is a topic directly related to democracy as well as to the improvement of bureaucratic processes. For private firms it means more competitiveness.
- *Experimentation*: organizations can analyze, even in real time, the huge amount of data they have interest in (e.g., about people, machine operations, natural phenomena, etc.). This enables them to understand complex processes and appreciate evolution dynamics as well as to perform forecasting by arranging controlled mining or simulation experiments. By using the information pointed out from these experimentations it is possible to foresee future organizational needs or changes in specific behaviors/trends in order to perform capacity planning and ensure efficiency over time.
- *Customization*: the highly specific segmentation provided by big data allows fine tailoring of solutions, products, and services in order to precisely address the expectations of people. For instance, we can think about differentiating the same service delivered on the base of age, or lifestyle, or place of residence. This greatly improves the usability of services by taking care of end-users' specific needs.
- *Human decision*: the availability of huge amounts of observations together with the always more affordable computing capacity allows the utilization of powerful automated algorithms to support human decisions. Sophisticated analytics can strongly improve decision-making processes, minimize risks, and discover important understanding that would otherwise continue to be hidden. The difference with the past is mainly in the amount of inputs available as well as its variety of sources and formats. This is a very promising issue for the improvement of both quality of life and risk management.
- *Innovation*: The analysis of big data can drive innovation and evolution. The creation of new business models, products, services, as well as new kind of social relations and ways to communicate and keep in touch can be greatly improved by the clever utilization of the large volumes of data available in modern IT systems. In order to give an idea of that we can think about tracking services related to people or goods. The details of paths that most people use to reach their every day's destinations (e.g., offices, homes) are extremely useful in order to design successful urban plans. Regarding real-time location services, there are several opportunities offered by the available location and/or tracking solution. For instance, there are insurances based on where, and how, people drive their cars. Imagination is the only limitation we have, when thinking about innovations based on big data analysis. In order to give another example of innovation value bring by big data, in 2003 some scientists founded *Sense Networks*, a company using real time and historical personal location data for predictive analysis. The first user application built by Sense Networks was *CitySense* [8] analyzing and showing the activity level

of a city and sending alerts in case of unexpectedly high activities. *CabSense* followed CitySense with the aim of providing its users with a list of street corners ranked by the number of Taxi picking up passengers.

3.2 The Big Data Value across Sectors

Several sectors of modern society have a great potential in drawing value from big data. According to [6] both the electronic devices' production and the information management sector can considerably gain from big data processing technologies and produce added value even in the near term. In particular, they can rely on big data analytics to generate insights to improve their strategies, products and services. On the other hand, other strategic sectors, such as transportation, manufacturing, and healthcare are characterized by a slightly lower potential in gaining value from big data, essentially because of the fragmentation of initiatives that, until now, have not yet allowed the involved organization to collect, integrate, and analyze significant amounts of mission critical data. Also government and financial organizations could better use their data in order to refine their growth and evolution plans and forecast economic events. In addition, big data correlation could provide a multidimensional view of their ecosystem generating powerful advices that can be helpful in optimizing their operations.

Clearly, data availability is the most obvious enabler for all the above opportunities. However, the widespread availability of massive amounts of information introduces new challenges related to data security and privacy. In fact, when coping with big data, the source is a fundamental matter not only to recognize what or who produces the data but also to understand which data can be collected/archived and how and where storing it, as well as what kind of data can be provided to who. As digital data travels across organizational boundaries, several policy issues, like privacy, security, intellectual property, and liability become critical. In particular, privacy is a concern whose importance grows as the big data value becomes more evident. Healthcare and financial data could be the most significant examples in terms of introduced benefits while being extremely privacy sensitive. In these cases it is necessary to deal with the trade-off between privacy and utility. The recent and past history teaches that data breaches can expose personal consumer information, confidential corporate information, and even national secrets or classified information so that security issues become more and more important. With the emergence of social networking technologies and new media for information sharing also the importance of property right has significantly grown. Clarifying who "owns" information and what rights come attached with a dataset, becomes fundamental for a fair use of the involved data. Finally, another basic question is related to liability: who is responsible for data utilization? Generally speaking, as the economic importance of big data increases, it also raises a number of legal issues, making the resulting scenario very complex to manage because data, being immaterial, are fundamentally different from many other assets, and it is impossible to compare it with traditional physical assets from the legal point of view.

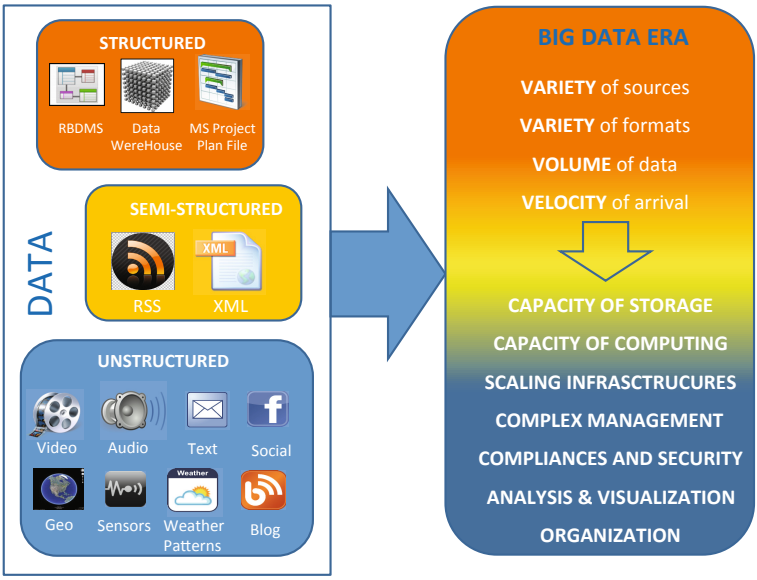


Fig. 4. Data typologies in big data

Professional skills are another fundamental factor since companies need new professions together with the related educational courses in order to generate value from big data technologies. Experts in data mining, machine learning, statistics, management, and analysis, can significantly help organization in the above task.

4 The Impact of Big Data on Technologies

In order to face the main technological challenges related to big data it is necessary to acquire a deep understanding of the fundamental features and dynamics characterizing the involved scenario. These features (see fig. 4) can be essentially individuated in the five "V"'s criteria:

- *Volume*: big data are huge in quantity and ever increasing. According to the recent IDC survey [9] the volume of data that will be managed by 2020 will increase more than 40 times over the current levels.
- *Velocity*: more data implies increased speed in accessing, transmitting, and processing them, so that, the proper technological and architectural solutions are to capture, understand, categorize, prioritize, and analyze big data at the maximum possible speed.
- *Variety*: big data may come from a large number of sources and may be archived by using a wide variety of formats, structured, unstructured, and

semi-structured. It is extremely important to integrate and design data management practices with these new types in mind, so that their diverse structured and unstructured formats can be quickly used to represent useful information
- *Verification*: with the volume of data the effort in governance, security, and compliance processes will also increase simply because the number of different typologies of data handled grows. As an immediate consequence, the related processes become heavier, by introducing the need of adopting automated strategies and tools in order to put in place new processes for verifying the quality and the compliance to rules. It is easy to think about data coming from credit card that have to be compliant with Payment Card Industry (PCI) roles and responsibilities.
- *Value*: it is the most important element and the main aim of any big data management and processing architecture. All the technologies, processes, and people involved in such architectures should be able to generate value from the available big data. Organizations should measure how their big data management frameworks affect the value of the insights, benefits, and business processes within their companies.

Starting from the above considerations it is clear to understand how technologies assume an always more crucial role in the success of almost all the initiatives related to big data processing. According to [10] they are the second most important factor that can potentially affect organizations after market-related ones.

4.1 Big Data and Technologies

Effectively managing big data means addressing all the technological challenges associated to the above five V's features, that are essentially the same characterizing the handling of legacy data, despite the new volume, and structural heterogeneity characteristics. The variety and the massive volume characterizing big data require significant infrastructure-level improvements. First of all, more scalable and high-performance computing and storage systems are needed. In addition, more flexible database management and archival services able to handle huge amounts of unstructured data become necessary. Also analytics and data mining practices may be subject to significant changes, since big data require the ability of operating in real time on several typologies of data, by going beyond the production of classical charts and graphs. Killer applications require massive parallelization of both processing and storage activities in order to scale out performance and capacities. Under the pressure of these new requirements, several technologies, and solutions are emerging or coming back in order to provide the needed flexibility and power characterizing new big data scenarios.

4.2 Storage Costs and Efficiency

Cost is one of the key parameters related to big data storage, since with the massive proliferation of structured, unstructured, and semi-structured data the

costs related to storage risk to grow uncontrolled. Fortunately, the cost for each kind of storage device, even SSDs, is decreasing as the amount of data to be stored grows, due the optimization of manufacturing processes and to the large diffusion of the involved devices. This introduces a partial rebalancing effect that, however, is not enough to compensate the data growth. In addition, increasing the quantity of storage available is not a final response to data proliferation, since more devices or higher capacity ones usually implies more space, but more power consumption, more risks, and less performance.

Improving the storage efficiency could be the key. There are several initiatives focused on storage features and aiming at improving their efficiency: *Deduplication, Compression, Scaling-out, Auto-tiering*, and *Thin Provisioning*. Each of them can improve substantially the efficiency in storage utilization and, accordingly, reduces power consumption, bandwidth demand, and management overhead. The effectiveness of each option depends on the specific application cases. At the same time each technique presents its tradeoffs, limiting its utilization scope.

Data Deduplication

Deduplication is an excellent way for optimizing data storage, mostly in multitenancy and centralized storage systems. While data volumes are continuously growing, a lot of data stored around the world are the same, or very similar. We can better appreciate this phenomenon by thinking about information published via social media, very often reproposed by several users. Deduplication avoids storing several times the same piece of data by partitioning an incoming data stream into several data chunks, and comparing such chunks with previously stored data. If a chunk is unique, then it can be stored safely. Otherwise, if an incoming chunk is a duplicate of some block of data that has already been stored, a reference to such block is associated to it and the data chunk is not stored again. In other words, deduplication algorithms analyze the data to be archived on disks and store only the unique blocks of a file or a set of files (e.g., present within a backup). This can reduce from 10 to 30 times the storage capacity needed, bringing significant economic benefits. The fundamental dynamics behind deduplication are sketched in fig. 5.

The choice of storing only unique chunks of data can be very effective in terms of storage space utilization, although it introduces the need for additional computing power in order to perform data deduplication-duplication in each single I/O operation. This can introduce nonnegligible latencies so that online deduplication can be performed only in presence of adequate performances and storage architectures. The deduplication ratio is the measure of deduplication efficiency. It can assume very different values case-by-case, depending on the kind of data stored as well as on the devices' occupation and hardware equipment. In a typical storage system deduplication efficiency can significantly vary over time. Initially, the storage occupation increases linearly since the device is not still able to perform deduplication. As the amount of available data grows the probability of finding replication of some pieces of data increases so that deduplication can start, by introducing significant savings in storage occupation. Several commercial deduplication solution are available such as *EMC Data*

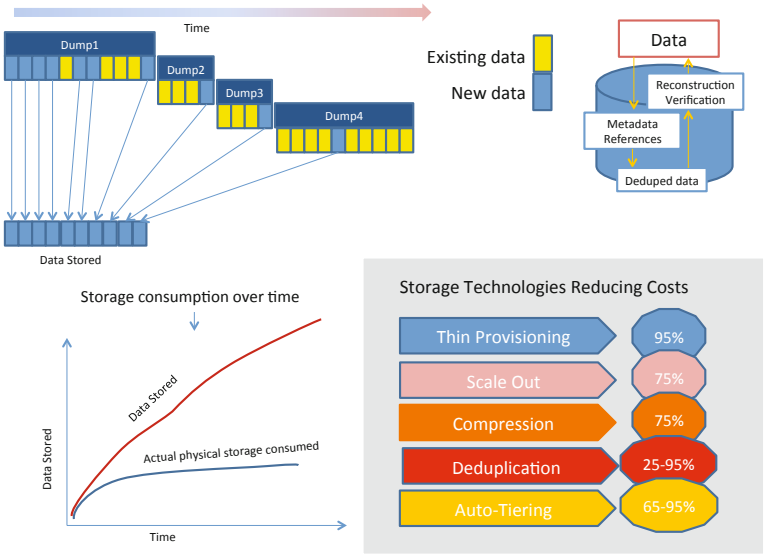


Fig. 5. The data deduplication process

Domain, HP StoreOnce, but deduplication facilities have also been added in the *ZFS* [11] open source implementation.

Data Compression

With the success of big data technologies, the demand for effective structured and unstructured data compression techniques is ever growing, in order to reduce storage space requirements and hence increment storage efficiency. Performance in compression and decompression function is achieved through a sophisticated balancing of hardware and software solutions, working on top of a sound data storage architecture. In particular, the right algorithms should be chosen for each kind of data, by considering a tradeoff between reduction in storage space occupied and efficiency in decompression/compression activities, that can take place online or offline depending on the chosen strategy. For example, lossless compression tools derived from the Lempel-Ziv-Welch [12][13] [14] scheme (such as Gzip [15]) or from the Burrows-Wheeler transform [16] (e.g. Bzip2 [17]) can be used for generic data, whereas specialized lossless or lossy solutions (e.g., those provided in JPEG [18] or PNG [19]) can be more effective to compress some specific kind of data, such as photo pictures. However, this may increase processing times by a factor 3-4, also introducing an additional overhead when data is inserted and updated.

Scale-Out Storage

Another strategic element for storage efficiency is scale-out storage, that is a storage architecture relying on a scaling methodology to buildup a dynamic

storage environment supporting balanced data growth on an on demand basis. Differently from legacy storage architectures based on a couple of controllers managing a fixed maximum number of disk arrays, and hence, limiting their scalability, scale-out storage architectures, are incrementally built by combining a large number of storage nodes, consisting of multiple low-cost servers and storage components, configured to create a manageable storage pool of virtually unlimited size. All the nodes in the systems behave like a single storage device and more capacity may be obtained on-demand by simply adding new nodes. While this scale-out approach can also be applied to direct-attached storage and *storage area networks* (SAN), it is usually associated with *network-attached storage* systems (NAS), that by using servers, disks, and management software to serve files over a network optimize both management and investment issues. A key component of scale-out systems are the Operating System-level networked storage services, enabling the nodes to be interconnected and referenced as a single object by storage administrators. So, the administrators have only the responsibility of managing data and not the underlying storage hardware. Also other services such as snapshots, replication, and data protection are provided at this level. In order to provide a scale out behavior, when starting from traditional solutions, clustering together multiple separate devices, it is necessary to add a software abstraction layer to simulate a single aggregated file system namespace. Anyway, the underlying limitations of separate volumes and multiple file systems still exist and must be managed. The EMC *Isilon* architecture, is a very good practical example describing the concept of scale out storage. It is based on the OneFs OS, managing storage nodes and supporting NFS, CIFS, HTTP/FTP, iSCSI, etc., access facilities via Ethernet links. Storage nodes are connected through a *Infiniband* back-end network supporting up to 144 (288 in the next Infiniband release) nodes, supporting multiple SSD, SAS or NL-SATA storage units. Fully automatic load balancing and autobalance of free space are provided at the OS-level. Other well-known proprietary solutions come from *Symantec FileStore*, and *IBM GPFS* [20].

On the other side, there are open solutions like *Scality*, that use industry standard hardware devices managed by a software glue that distributes information across storage nodes by granting performance and protection. The result is an object-based scale-out architecture able to reduce substantially both storage and operation costs and supporting advanced services such as storage auto-tiering, balancing, etc.

Scale-out solutions promise a near linear scalability, that is, the performance and the efficiency of the whole storage solutions scales linearly by adding storage node. Another important feature of these solutions is granting storage distribution locally or geographically.

Thin Provisioning

Thin provisioning is essentially a flexible storage allocation strategy achieving just-in-time storage space provisioning, that is allocating new room on disks only when it is strictly needed and not at the system setup time, in order to avoid any kind of storage over dimensioning at any time. In such a way storage

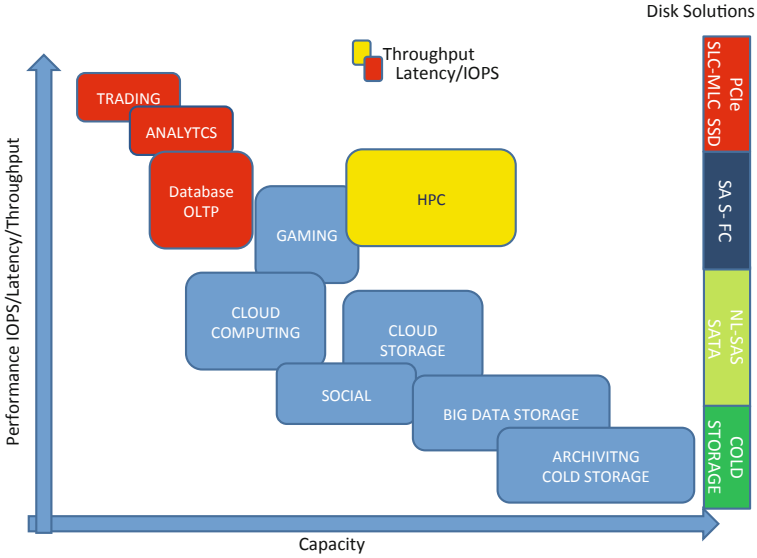


Fig. 6. Application performances characterizations

space is available on demand only when an application really needs to consume it. All the storage space modifications should take place without disrupting the applications using them. This requires flexible and automatic mapping of logical units and volume resize, by keeping the provisioning-related management overhead at minimum. Thin provisioning greatly benefits from capacity planning, predicting the storage upgrade times by analyzing dynamically storage usage patterns. Storage systems providing such facilities have to include tools constantly monitoring storage utilization and sending alerts when volume occupation goes over predefined thresholds. Thin provisioning also allows coupling multiple applications or servers to use a common storage pool in order to better share the available disk space. This saves a lot of money because an organization has to buy more storage only when needed, by maximizing it utilization since only a minimum part of storage space is left empty.

Storage Tiering

Storage Efficiency is also a concept related to the access performance needed by applications, that according to their mission, purposes, operating environment, and constraints (see fig. 6) may be characterized by specific demands usually expressed in terms of throughput, latency, and protection, through specific *Service Level Agreements* (SLA). In order to honor these SLAs, last generation of storage solutions use the concept of *Tiering*.

In detail, storage tiering manages the migration of active data on higher-performance storage devices (the topmost tiers) and inactive data to low-cost, high-capacity ones. The result is an increased performance, lower costs, and a denser footprint than conventional systems. By moving data on disk devices

suitable for ensuring a specific SLA such architectures can realize a direct link between data and their related SLAs. For instance, if a piece of database requires a 200 IOPS access performance it must be placed on devices (SSD or SAS disks) which are able to fulfill such SLA. A similar approach is very useful also to balance performances and costs, by taking the most from a limited amount of high performance storage devices (e.g., SSD units) while owning a virtually unlimited storage space built by using low-cost devices (e.g., NL-SATA disks). That is, through the concept of storage hierarchy, one can rely on the abstraction of a virtual storage space characterized by the capacity of the whole aggregate and the performance of the SSD devices.

Usually three tiers are enough, associated to data classification:

- *Tier 1*: Mission Critical data, when the highest degree of performance, reliability, and accessibility are needed
- *Tier 2*: Seldom used data, when mid-level performance is acceptable
- *Tier 3*: Archive data, when only retention required for compliance

The migration of data between storage tiers is based on fully automated or manual policies. *Auto-Tiering* or Dynamic Storage Tiering seems to be the most interesting way to manage data movement across disks. Clearly such a feature simplifies operations and therefore the related costs. A typical criterion on which basis moving data is the access frequency. Data mostly accessed is moved into higher performance storage layer and vice-versa. At the same time, applications and people can play a role in data moving across storage layers, deciding when pieces of data have to be moved or deciding that some piece of data must be stored in certain devices. That can happen, for instance, when access to data requires a particularly high performance or in the case when data are permanently archived.

SSD Devices and In-Memory Storage

The current HDD technology has the perspective of improving its density but is close to its maximum performance in terms of latency and IOPS. When using HDD devices the only way to increase performance is parallelizing workloads across more disks and, consequently, over dimensioning the storage capacity. The evolution of SSD technologies allows us to fill the gap between HDD and RAM, overtaking historical limitation in terms of performance sustainability and costs. Thus, when I/O performance and latency are the main requirements characterizing the storage architecture, SSD would be the right choice. In addition to performance, SSD technology also brings significant improvements in energy efficiency and heating, that indirectly reduces storage costs by containing the overall power consumption and cooling requirements (see fig. 7).

There are essentially three models for SSD utilization in storage architectures:

- *Server Attached*: when SSD cards are directly hosted by servers, this model offers the best performance but it requires an overlay architecture that moves and protects data, making data sharing, and storage optimization more difficult

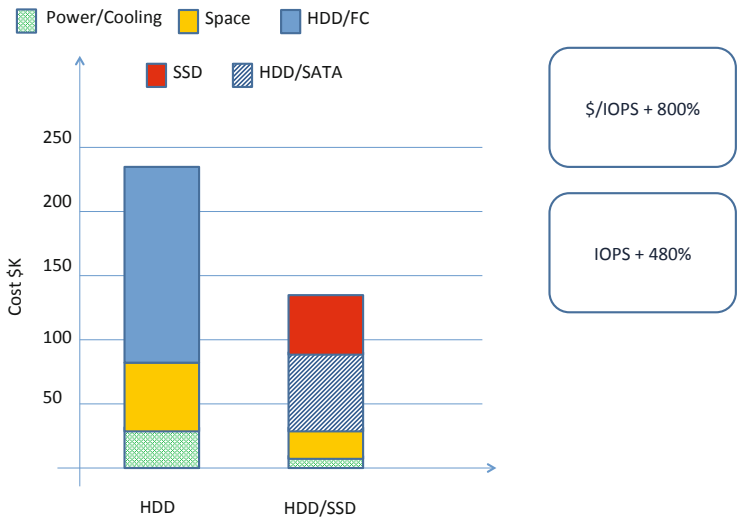


Fig. 7. HDD-based vs. HDD+SSD-based storage costs

- *SSD Appliances*: this is another device attached to the network, it easy to add in existing infrastructures and can not only provide high performance, but also requires data management
- *SSD in multipurpose storage devices*: in this case SSD is a storage layer, together with SAS HDDs and SATA HDDs, in a network storage machine (e.g., SANs). Adding SSD devices to existing infrastructures and leveraging existing data management features for protection and auto-tiering it is very easy. On the other hand legacy storage machine architectures can introduce limitations for SSD performance: putting drive form factor SSDs into these legacy arrays quickly shifts the bottleneck away from the back-end storage media onto the disk array controllers.

However, when the performance provided by SSD devices is not enough, another emerging architecture capable of providing the highest levels of performance is direct "in memory" storage. Such architecture implies having a database (for structured data) as well as a file system (for unstructured data) or a relevant part of it, entirely stored in RAM. This often avoids the need of developing access optimization techniques like indexes, aggregates, and designing special purpose database architectures like stars or cubes. The main drivers pushing such in-memory solutions are essentially:

- *Storage technology evolution*, mainly in terms of density increase and cost reduction of DRAM technology and performance improvement of solid state storage devices, makes in-memory storage both feasible and affordable;
- *Emerging database features*, like compression or column-centric databases are reducing the dimension of the storage required for them;

– *Real time reporting and access needs*, reducing drastically the expected ETL (extract, transform, load) processing times

In memory architectures have several advantages compared to legacy solutions mostly if related to business intelligence and data analytics. Real-time analytics is one of the most appealing features offered by in memory solutions. It allows very fast information navigation, correlation and extraction.

Furthermore, while developing a traditional business intelligence tool can take more of 15-17 months, in memory solutions allow reducing considerably this time since there is no more need of introducing sophisticated optimization techniques. On the other hand, the amount of RAM needed increase as users grow, affecting the costs of the solution. For this reason, several vendors are introducing high-performance solid state memory in place of DRAM.

Many market leader vendors are investing in such technologies and releasing new products based on in-memory solutions (e.g., *SAP Hana, WebDNA, H2, HazelCast, UnQLite, EhCache, Oracle TimesTen*) and several available benchmarks show that in-memory databases provide near-linear scalability. For instance, *ExtremeDB* by McObject claims linear scalability within a benchmark consisting of 160 64-bit processor cores and over 1 terabyte of data completely in memory. On the other hand, an open source MySQL Cluster tested on a 16-node cluster achieved 500,000 reads/second, and increasing the number of nodes to 32 triples the performance. However, performance is just the most visible aspect of in-memory storage, but it should be also considered that almost all the available solutions can provide replication to ensure availability of the data as well as load balancing.

In order to scale-out in-memory storage architectures the new concept of *in-memory data grid* (IMDG) has been proposed. It differs from traditional in-memory system in distributing and storing data in multiple memorization nodes scattered throughout the network. It is also based on a quite different data representation model, usually object-oriented (serialized) and nonrelational. At a first glance an IMDG is a distributed data base providing an interface similar to a concurrent hash map. It memorizes objects by associating them with keys according to the traditional key-value mapping schema. There are also some other features in IMDGs that distinguish them from other products, such as NoSQL and in-memory databases. One of the main differences would be truly scalable Data Partitioning across clusters. Essentially, IMDGs in their purest form can be viewed as distributed hash maps with every key cached on a particular cluster node – the bigger the cluster, the more data you can cache. In a well-designed IMDG, there is no, or minimal, data movement. The only movement, aimed to reapportioning data along the cluster, should be when nodes are added or removed. In this situation, processing should be performed only in the nodes where data is cached.

5 Big Data Management and Analytics

Generally speaking, the big data landscape is getting very complex due to the emerging needs and opportunities coming from such a rapidly evolving scenario. It essentially encompasses two main classes of technologies: real-time data management systems that provide operational capabilities for real-time, interactive workloads, and systems offering analytical capabilities that can be used to perform retrospective and complex analysis tasks that usually need to access most or even all the data of interest. Although these classes are complementary they are frequently deployed together.

New kind of database management systems have been designed to take advantage of new distributed computing and storage architectures (e.g., clouds and grids) in order to exploit massive computational power and archival capacity. This allows also managing real-time big data workloads in a much easier and cheaper way, making implementation efforts considerably faster. This is fundamental to allow some degree of direct interaction with the data as soon as they are produced, for example, in financial or environment monitoring applications. On the other hand, big data analytics, by allowing processing and correlation of huge amounts of data in reasonable time, can foster the emergence of previously hidden insights, and support effective forecasting by revealing unknown trends and evolution patterns, by removing the need for sampling and polls as well as promoting a new more investigative and deterministic approach to data analysis, leading to more reliable and precise results.

Due to the volumes of data involved, big data analytics workloads tend to be addressed by using parallel architectures like *massively parallel processing* (MPP) and *MapReduce*-based systems. These technologies are also a reaction to the limitations of traditional databases and their lack of ability to scale beyond the resources granted by a single cluster of servers. Furthermore, the MapReduce strategy provides a new way for analyzing data that is complementary to the capabilities provided by SQL.

5.1 Database Architectures for Big Data Processing

Traditional databases have substantial limitations when working with big data. By summarizing, the most critical ones are:

- they do not support neither mixtures of unstructured data nor nontabular data
- they are often based on quite old architectural concepts, not specifically conceived for parallel processing
- their operating speed, does not pace with network capacity and business needs
- they do not scale well, clustering beyond few servers is hard
- they do not address the need of combining data from unrelated sources in order to integrate them into a common schema
- they cannot handle information volumes over the petabyte.

In order to cope with the above problems, a higher degree of parallelism in both data storage and query processing is needed.

Parallel Architectures
Traditional parallel database systems are structured according to a *Symmetric Multiprocessor* (SMP) architecture, where multiple CPU are available to run the data retrieval and processing software and manage the storage resources and memory disks, but only a single CPU is used to perform database searches within the context of a single query.

The new massively parallel processing (MPP) architectures are designed to allow faster query operations on very large volumes of data. These architectures are built by combining multiple independent server/storage units working in parallel, in order to achieve linear increases in processing performance and scalability. Spreading data across many independent units in small slices results in more efficient database searches, whose performances increase roughly proportionally to the number of involved units.

All the interactions between the server units are accomplished through a high-performance network, so that, there is no disk-level sharing or any kind of contention to be handled. For this reason such architectures are also known as "shared-nothing" schemes.

The Data Models
When considering the underlying data models we can essentially consider two classes: relational and nonrelational solutions, introducing concepts addressing the specific operational requirements.

Due to their mostly unstructured nature big data are quite different from traditional structured data organized in tabular form by relational databases, so that, in lack of structured tables, standard query operations such as table join just do not make sense. Also, the relational model implies that well-defined schemas for data to be stored must be fixed in advance, before inserting any data into the database and that such schemas must remain static throughout all the data lifetime. This is a big limitation that does not fit with the agile development requirements often characterizing big data processing scenarios, since each time you dynamically introduce new data processing features, the database schema may be subject to changes, that in presence of static schemas will result in complex data transformations/migrations implying significant downtimes. In addition, due to their inherent architectural features, relational databases are characterized by a vertical scaling behavior, that is, a single machine or tightly coupled machine cluster is needed to run the entire database in order to provide effective data access as well as reliability and availability. This introduces the presence of single points of failure and severely limits the whole approach's scalability.

NoSQL Systems
For this reason, new kinds of databases, known as *NoSQL* systems [21], are emerging, based on architectures less constrained than the traditional relational

databases, in order to overcome some of their limitations in flexibility, complexity, performances, and costs. These solutions need to ensure horizontal scalability, so that, the capacity grows linearly by adding new server and storage elements instead of bringing more and more capacity in single servers or storage systems. However, in order to deliver such degree of scalability some of the so-called ACID (atomicity, consistency, isolation, and durability) properties, characterizing relational database transactions (in particular consistency and durability), have been relaxed or partially redefined in many NoSQL systems.

NoSQL databases support the insertion of new data without a statically predefined schema, allowing real-time application changes, that means agile development, better integration, and enhanced reliability. Most of them also provide automatic data replication mechanisms, in order to achieve data availability and support disaster recovery without the need of specific upper-layer applications devoted to these tasks.

In contrast with their name, these NoSQL databases do not disallow the use of a powerful structured query language such as SQL but only suppress some specific relational mechanisms such as all the operations that rely on fixed table schemas. More precisely, being natively nonrelational, instead of providing the table abstraction, they may present data as organized into objects or key/value pairs. This latter approach is known a *Key-Value Store DB* since each stored element is characterized by a primary key and a collection of values, also called bins. It essentially stores the data records one by one according to a row-wise policy since all of the elements within a single record are stored together, in a structure that can be conceptually seen as a row. Despite being extremely efficient in granting access to huge volumes of data, in a row-wise storage schema the blocks storing values for each single record are structured as a sequence of columns making up the entire row, so that internal fragmentation phenomena can occur if block size is greater than the size of a record, or alternatively, more than a block is needed for storing the record but the last block is not completely filled, resulting in inefficient disk space usage.

To cope with this problem, an alternative strategy, known as *columnar storage* [22][23] often combined with the aforementioned in-memory technologies, has become quite popular. By using columnar storage, each data block stores values of a single column for multiple rows, so that, data is physically grouped by column. In presence of very large numbers of columns and huge row counts, significant storage efficiency can be achieved [24] since the columnar approach allows homogeneous data to be stored all together, leading to the possibility for more effective compression, and hence, reduced disk space requirements. Since many data access queries only involve a limited number of columns at a time, it is possible to achieve substantial access performance improvements by only retrieving the blocks associated to the columns actually involved in the query. This means that reading the same number of column field values for the same number of records requires a much lower number of I/O operations compared to row-wise storage.

The above savings in disk space and access performance also affect the activities of retrieving and then storing data in memory so that larger amounts of data can be loaded entirely into RAM by combining columnar and in-memory technologies, so that both transactional and decision-support queries can achieve almost zero latency responses.

Finally, *Document-based* databases, store and organize data as collections of entire documents in the form of *JSON* objects [25], rather than as structured tables. These objects can be seen as nested key-value pairs that can be nested as much as you want. They can model arrays and understands different data types, such as strings, numbers, and Boolean values.

Currently, the most popular NoSQL solutions available are the open source Apache *Cassandra* DB [26] once used as the Facebook database, as well as Google *BigTable* [27], LinkedIn *Voldemort* [28], Twitter *FlockDB* [29], Amazon *Dynamo* [30], Yahoo! *PNUTS* [31], and *MongoDB* [32].

5.2 The MapReduce Paradigm

MapReduce is a method for distributing tasks across nodes in order to co-locate processes and related data [33]. It consists in two phases: Map and Reduce. Input data are decomposed in multiple independent records or chunks, processed individually and in parallel by the Mapper. The achieved results are then sorted and aggregated by the Reducer. MapReduce operates exclusively on $(key, value)$ pairs, so that, the input of each process can be seen as a set of $(key, value)$ pairs and the process results in another set of $(key, value)$ pairs, conceivably of different types.

In detail, the input data to a MapReduce process is structured in multiple key-value pairs (k_i^I, v_i^I). The map function processes all these pairs one by one, resulting a list of zero or more (k_j^O, v_j^O) pairs. Such (k_j^O, v_j^O) output pairs are collected and reorganized so that all the pairs with the same keys k_i^O are put together into more complex pairs represented as $(k_i^O, list(v_i^O))$, that are processed by the reduce function one by one. For each of them the reduce function emits a (k_j^O, v_j^R) pair. All these pairs together coalesce into the final result, so that, the whole process can be summarized by using the following equations.

$$map((k_i^I, v_i^I)) \rightarrow list((k_j^O, v_j^O)) \text{ with } i \in [1, N] \,|N > 0,\ j \in [1, M] \,|M \geq 0 \quad (1)$$

$$reduce((k_i^O, list(v_i^O))) \rightarrow (k_i^O, v_i^R) \text{ with } i \in [1, M] \mid M > 0 \quad (2)$$

The resulting distributed processing model is simple to understand and simultaneously very expressive. Many large-scale big data processing problems can be handled in parallel according to such model trough several Map and Reduce steps. Typically both the input and the output of each step are stored in a distributed file system. The map and reduce functions are implemented through appropriate interfaces and/or abstract-classes by starting from these inputs and outputs. The framework managing the MapReduce process also needs to take care of scheduling the needed tasks, monitoring, and reexecuting them in case of

failure. Where data processing algorithms offering more sophisticated features than simple aggregation are required, MapReduce emerges as the first choice for big data analytics. Some big data management databases provide native MapReduce facilities that allow in place analysis for the involved data. Alternately, data can be copied from these databases into more specific analysis platforms such as Apache Hadoop in order to perform Map & Reduce processing.

5.3 Hybrid Solutions

New technologies like NoSQL and MPP databases or Hadoop have emerged to address the new big data challenges and to enable new solutions and services to be delivered.

Several solutions are leveraging the capabilities of both systems by integrating a NoSQL database with Hadoop. The connection is easily made by existing APIs and allows analysts and data scientists to perform complex, retroactive queries for analysis, and insights while maintaining the efficiency and ease-of-use of a NoSQL database.

NoSQL, MPP databases, and Hadoop are complementary: NoSQL systems should be used to access big data and provide operational intelligence to users and MPP databases and Hadoop should be used to provide analytical insight for analysts and data scientists.

5.4 The Hadoop Processing Framework

Several big data processing frameworks emerged in the last years in order to support the most demanding analytics tasks, with the most famous and successful of them being the Hadoop platform [34]. Hadoop is one of the most important projects in the Apache Software Portfolio and the core of Apache big data solutions. In the Apache view Hadoop is the strategic engine for data management in any kind of organization managing big data processing tasks. It can be integrated with traditional data warehousing solution as well as emerging solutions more oriented to big data processing, becoming the cornerstone of data management architectures and bridging from traditional architectures to big data ones.

Hadoop is essentially a distributed fault-tolerant solution for data storage and processing. It is composed by two main components: the *Hadoop Distributed File System (HDFS)* [35] and the MapReduce engine. HDFS is the component that handles distributed data storage across clusters, by sitting on top of existing native file systems solutions (e.g., ext3fs, ufs, etc.) and managing high bandwidth data transfers between the involved nodes. It organizes data in files (that can reach the Terabyte in size) and directories, where the files are divided in blocks and then distributed across cluster nodes. Blocks and their associated processing activities are co-located on cluster nodes by Map-Reduce at runtime. Also, in order to provide data protection, blocks are replicated and their integrity controlled by using checksums. HDFS also support snapshot and replication facilities as well as undelete operations.

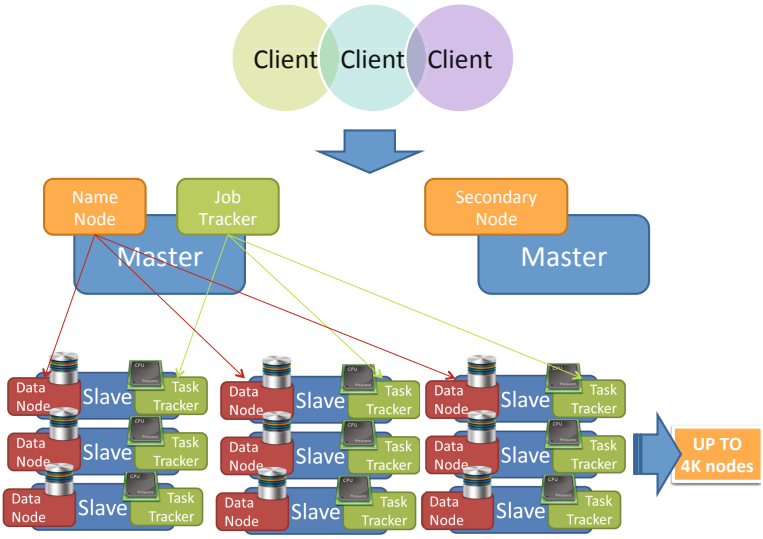


Fig. 8. The Hadoop architecture

The whole Hadoop architecture is sketched in fig.8, where the entire system is managed by a couple of Master servers running:

- a *Name node* that manages the HDFS hierarchical name space, by controlling the distributed Data Nodes running on a large number of machines. The Name node also manages blocks replica placement according to a rack-aware strategy. It also keeps block metadata in memory in order to provide faster access. Name nodes can be arranged in a federated way.
- A MapReduce engine *JobTracker*, governing the execution of data processing jobs and managing the Task scheduling decisions, to which client applications submit MapReduce jobs. The JobTracker pushes work out to the available *TaskTracker* nodes, distributed throughout the network or within a local cluster, striving to keep the runtime workload as close to the data as possible.

The Data Nodes, running on multiple machines equipped with their storage resources, are responsible for serving HDFS read and write requests by performing block creation, retrieval, deletion, and replication under the control of the Name Node. Thus, client applications achieve data access through Name Node, however, once the Name Node has provided the location information of the data, they can directly interface to one or more Data Nodes. The data blocks can be read in parallel from several nodes simultaneously.

The Task Trackers govern the execution of tasks and periodically reports the progress of such tasks by using a heartbeat message. TaskTracker instances should, be deployed on the same nodes that host Data Node instances, so that Map and Reduce operations are performed close to the data.

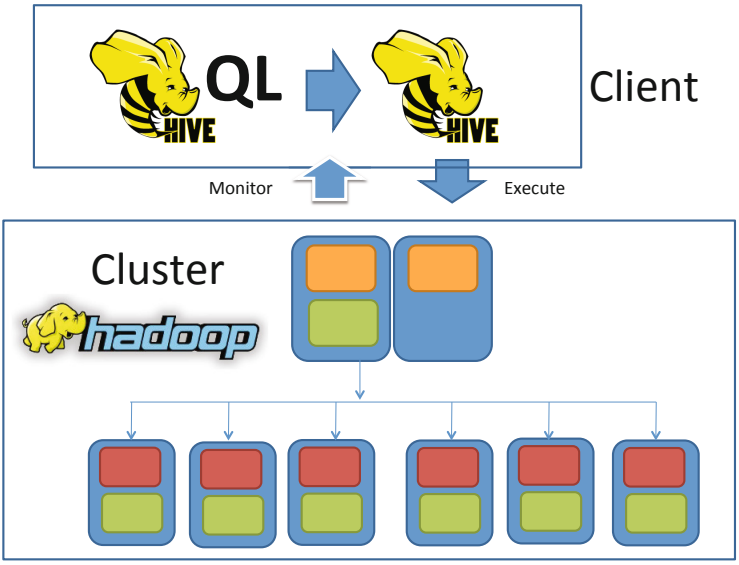


Fig. 9. The Hive – Hadoop relation

The Hadoop File System appears as a single disk and can address PBs of data running on native file systems, that can be extremely effective for storing large files, streaming data, managing write once, and read many operations, all implemented on commodity hardware. However, it is not so efficient in managing small files, low-latency access operations, and multiple writers.

Part of the Apache data management framework are also HBase and Hive. HBase [36] is a Column-Oriented data store known as Hadoop Database. It is fully distributed in order to serve very large tables. HBase is horizontally scalable and integrated with MapReduce and built on top of the HDFS file system. It also supports real-time CRUD (Create, Read, Update, Delete) operations unlike native HDFS. HBase is the platform of choice when there are lots of data and large amount of clients/requests to be handles. However, it is not so efficient when traditional relational database retrieval operations are needed or in presence of text-based searches. Hive [37] is a data warehousing solution built on top of Hadoop (see fig. 9). It provides a SQL-like query language called HiveQL and is able to structure various types of data formats and accesses data storage space from various solutions such as HDFS and HBase. It has been explicitly designed for ease of use and scalability and not for low latency or real time queries.

5.5 Big Data Analytics

Analytics reporting processes are fundamental elements within the big data framework, since they are the means which extract valuable information from

collected data. From analytics and visualization tools is possible to gain competitiveness from big data by:

- taking the right decisions from all the available information
- Predicting changes and behaviors and reacting preemptively to strategy mutations
- discovering new opportunities such as new business or market segments
- Increasing efficiency by changing, for instance, processes, or products characteristics.
- Quantifying current and potential risks associated to activities and processes

Analytics reporting is a classic IT function, but the ideas, and mostly the results, coming from big data analytics are very different form the past. The picture reported in fig. 10 gives an effective summarization of analytics reporting in the big data era. Clearly, what is totally different form the past is not only just the huge amount of data but also the level of complexity, speed, and accuracy required by the analysis process. In the past, analytics on transactional or structured data have helped organization in gaining competitiveness. Nowadays, data coming from social, sensor, video, image, and machine-to-machine sources represent a major opportunity for organizations and a big challenge for analytics tasks. In fact in-deep the examination of such data may support organization in better understanding their customers, partners, operations, and more generally their business or efficiency improvement opportunities. Emerging systems that deal with big data have to face with the enormous quantity of data available and with its variety. Also, the rate at which new data is created and stored is increasing more and more and, at the same time, the life of data is dropping from year and months to hours and even seconds. Hence, there is a need for analyzing data at the same speed in order to achieve up-to-date results.

If we take a closer look at the evolution of analytics, we perceive the emergence on new kind of analysis processes that are the result of new datasets. Figure 11 shows how the most significant analytics processes are evolving in in order to cope with big data.

Several old and new techniques can be employed for advanced analytics such as machine learning, data mining, artificial intelligence, natural language processing, genetic algorithms, and so on. The main difference between old analytics and big data analytics are basically two: data formats and structures as well as ability to predict the future. In fact, big data analytics processes provide a progressive view enabling organization to anticipate opportunities. They are able to perform complex correlations and cross-channel analysis, together with real-time forecasting. Nevertheless, they can largely use the new generation collaborative technologies, whereas classic analytics are only able to achieve a rear view on historical and structured data.

The availability of new kind of data allows to get more in depth, in understanding the associated phenomena. Thus, new generation of analytics allow exploring granular details and answer questions considered beyond reach in the past, like: why it did happen? When will it happen again? What caused it happen?

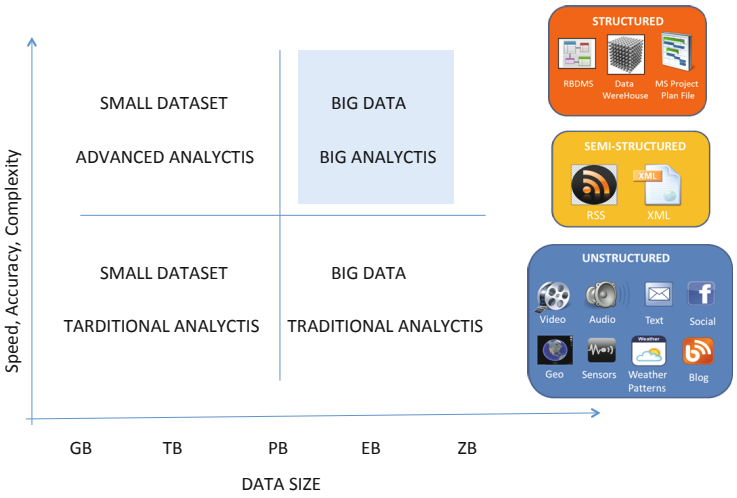


Fig. 10. Big data analytics

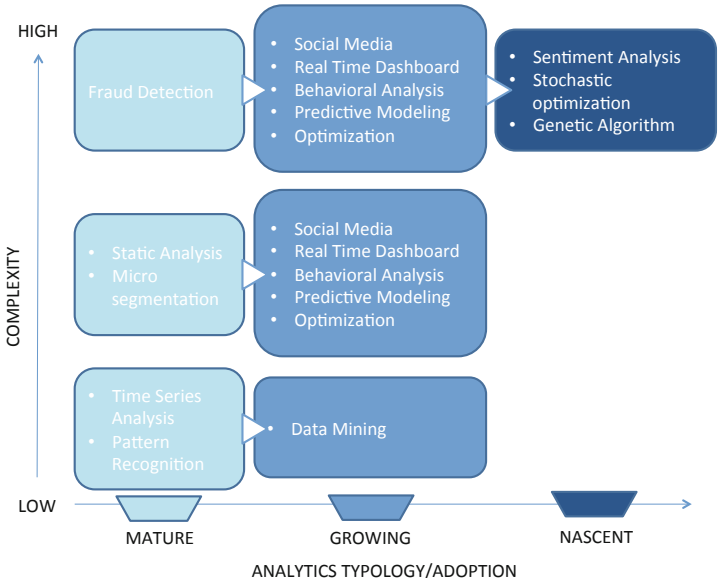


Fig. 11. Analytics adoption status

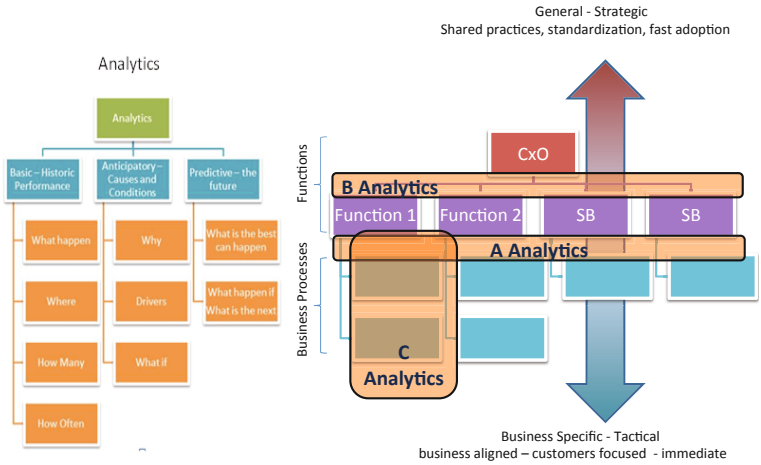


Fig. 12. Analytics deployment in a typical organization

What can be done to avoid it? Emerging analytics promise to be predictive and prescriptive, to improve decision making and efficiency, by discovering valuable insights that otherwise remain hidden.

Just to give an idea of the difference, big data analytics can detect and report in real time the consumer emotions during a service call on mentioning a competitor, that is a piece of information very useful in real time. Another example, regarding the personal business arena comes from the new offers of car insurance companies that, just by installing a little control unit on the user's vehicles allow tailoring an insurance contract on the specific user's habits, for a win-win business model where both the company and the user can take advantage of data produced by the sensors and analyzed by specific analytics tool.

It is always necessary to match the analytics solution with the current organization model in order to assign outcomes to the right organization elements. This responds to questions like: What outcomes can see who? What are priorities in using analytics? The example in fig. 12 shows that analytics can be deployed in a traditional organization according to three fundamental approaches indicated with A) B) and C):

- *A) Embedded shared mode*: analytics are deployed in a centralized mode and serve the entire organization. In embedded mode analytics support processes and increase efficiency. It is useful for standardization of processes and methodologies, and for sharing practices and services among functions. But it is not directly related to specific business issues and customers.
- *B) Stand-alone shared mode*: analytics are deployed as in the A) model but outside organizational functions. The outcomes of stand-alone mode are executive-level reports focused to the core competencies of the organization. It can enable the development of standardized processes and methodologies

at the cross-enterprise level, giving a high-level view of the organization, and losing the specific business view. It enables best practices and insight at the organization level. Of course, such a kind of approach lacks specific business visibility together with the related practical know-how.
- *C) decentralized mode*: analytics are deployed specifically for each function, that increases decision and execution speed, and is closer to business issues and customers. But this approach lacks of sharing practices and strategies, could introduce duplication, redundancies, and is unable to standardize at organization level.

The above examples of analytics deployment give an idea of pros and cons related to each approach. It is quite clear that each decisional level in the organizations needs its specific analytics though which is possible address specific business and customers' opportunities. Higher level of analytics are focused on comprehensive organization performance and efficiency, enabling directors to give shared best practices, and methodologies. In a perfect world, organizations should have one analytics system able to provide several views to address the needs of both specific business lines and directors. However, in the real world, organization have to face the availability of analytics products on the market and their costs. In addition, each approach described above needs its data in term of sources and typology, so that more analytics means more data, consequently, more infrastructures, compliances, and security.

Finally, the analytics processes need to be associated to proper visualization tools and decisional dashboards, in order to make their outcomes intelligible to humans and support their added value extraction.

5.6 Results Visualization

Implementing a big data analytics solution without valid means for communicating valuable information to humans is not useful. Systems that receive data and present them to humans are called *Visualization Systems*. These systems have to shape and present data in the best formats for humans' perception characteristics in order to allow them to understand complex analysis results and take advantages from them. Correlated to big data science, there is a tremendous amount of research and innovation about results visualization, mainly oriented to the creation of images, diagrams, animations, tables, with the aim of making data easily understandable. The most common data Visualization options available are:

- *Tagging*, is one of the most utilized means to communicate keywords. In web 2.0 tagging is a very common way to highlight information. The concept is underlining words that most frequently appear through larger typefaces and words that less frequently appear through smaller typefaces.
- *Clustergram*, is a visualization method that displays how members of a data set are assigned to clusters as the number of clusters increases (see fig. 13). This method is helpful in understanding how clustering changes with different amount of clusters.

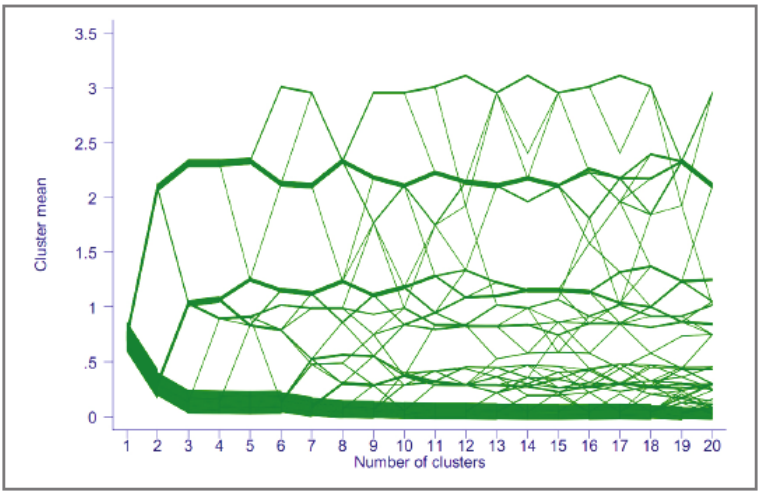


Fig. 13. A clustergram example [6]

- *History flow*, is a visualization technique depicting the evolution of a document over time as it is edited by multiple contributing authors. It reports the time on the horizontal axis and contributions on the vertical one with a different color code associated to each author. By using the color it is possible track the quantity of text written by authors through the length of a bar.
- *Spatial information flow*, is a visualization technique that depicts how information flow in their space of representation. It is a very powerful synoptic means to visualize information. Fig. 14 shows the connections between *LinkedIn* people that are part of different professional networks. It depicts how people of each network are strongly linked each other, so that the commercial networking group is connected to all the other as well as IP storage network group.
- *Correlation*, the picture in fig. 15 is an example of heat map correlation visualization technique, where values are represented by colors. In this case the colors are linked with geographical and time information.
- *Dashboards*, are visions at a glance of all the *key performance indicators* (KPI) of a process. The aim of dashboards is to depict the status or the health of a controlled process in a single view, reporting all its critical parameters. Usually dashboard tools allow performing a drill down or route cause analysis in order to understand why one or more of the parameters under control are negative compared with its normal behavior or with its target range of values. The fig. 16 example picture shows the dashboard for a Formula 1 grand prix, where teams can control the health of their cars.

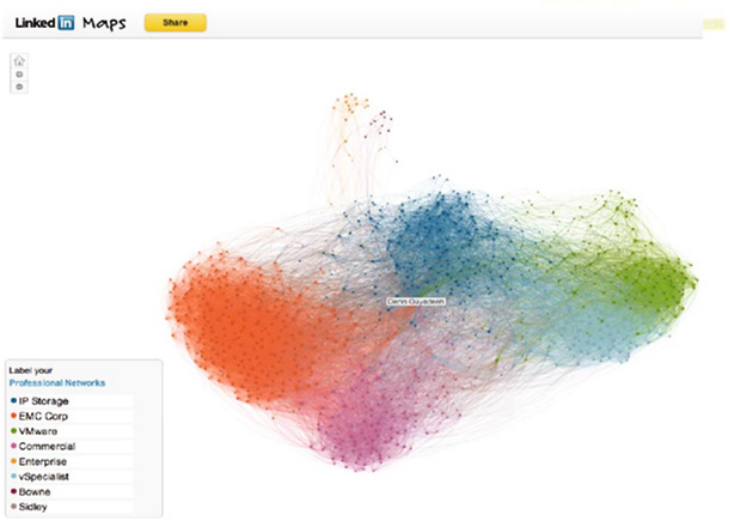


Fig. 14. LinkedIn adjacencies spatial information flow

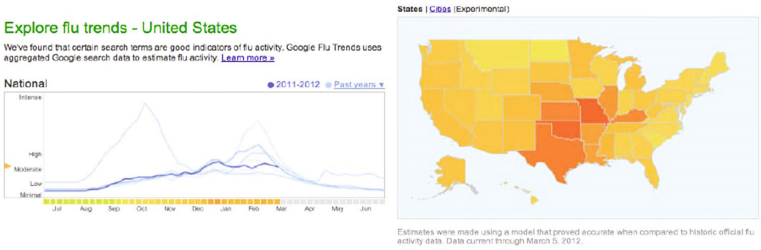


Fig. 15. Correlation map of historical Flu activity data

6 Big Data Models

When an organization decides to invest in big data, in order to adopt a new data-driven meta-business approach, it must define a properly tailored model to take advantage of all the opportunities offered. Such model can be seen as an abstract layer needed to represent and manage the data collected by the involved organization units. Its cost and complexity has significantly increased over the last years in order to meet the ever growing needs of big data. To construct a big data model, we must first define the data architecture by modeling the fundamental blocks based on storage needs, the involved data types, their relationships and access requirements, and so on. Then, we have to model applications needed to maintain analyze, display, and store the data of interest.

Fig. 16. Formula 1 dashboard

Most of the new models are based on a scaled-out, shared-nothing architecture, requiring fundamental choices for the interested organization in order to decide the technologies to be used as well as where and how to use them. Continuous evolution is the fundamental characteristic of a big data model, together with the modification of the organization that the big data outcomes can imply. In addition these outcomes together with an increasing understanding of the model could change the implemented model itself. For instance, an increasing awareness of a specific process could suggest new business models or new data to be collected and analyzed. When designing the big data model for an organization the fundamental issues to be considered are:

- *Understanding business goals and potential benefits*: the first pace in big data modeling approach is taking a step back and thinking about the key business drivers by focusing on which ones can take advantage of big data.
- *Defining data and outcomes*: organization must have a clear idea of what kind of information they need and consequently, they should be aware about what they already have, what is relevant, what is missing. It is important to define the data sources and the volumes of data involved. In large organizations, potentially valuable data often exists in multiple shims. All information about data should be clearly related to the outcomes useful to achieve the defined business goals.
- *Analyzing compliances, privacy, security, and liability requirements*: organization should be aware about consequences of data utilization, this is a critical factor to be addressed in terms of both processes and technologies.
- *Defining data management processes and practices*: it is very important, in the first phase of the model adoption, keeping the traditional production operational environment coexisting with a new environment where big data are sift in order to achieve the defined business goal. This approach is useful

to minimize the adoption risks and, to compare the models. Later, the new environment will be integrated with the original one. The new model has to cover, from sources to outcomes, the delivery of data across the organization
- *Choosing technological infrastructures*: organization have to define the infrastructures to be used in order to take advantage of big data and, at the same time, balance the costs/benefits, and protect the investment in mid time. The chosen technological infrastructures should be scalable, adaptable, able to fulfill performance request from business, and to allow customers consuming big data outcomes. The adaptability of the infrastructure could be a critical factor for a business data-driven organization since modifying the running business model is inherent in the data-driven model itself. The delivery of the outcomes is another key factor involving technological choices: once decided who can access outcomes and how, technological infrastructures have to deliver those services. Finally, technological infrastructures have to support the privacy and compliance requirements.
- *Knowing the people involved*: once the data management model has been defined, in addition to the technological infrastructure details it is necessary to address the skills missing, by identifying roles and profiles
- *Evolving the organization*: the organization who invests in a big data-based business model has to be ready to embrace a new culture driven by data flowing across its departments. In addition, it should be aware of the impacts of outcomes and be ready to react to them. For instance, if a predictive analysis forecasts the decay of a certain kind of business on a geographic area, the organization has to be ready to change its supply chain with the sake of being more cost-effective. In order to take the most advantages of big data, outcomes could be delivered to all people who are potentially interested in, allowing to power up an engine of ideas (self-service query model).
- *Managing outcomes*: once the model starts producing outcomes, they have to be consumed. Outcomes can imply modifications of processes, behaviors, products, communications, and so on. Each modification should be evaluated in terms of costs and potential benefits.
- *Performing evaluation and evolution*: once the model is deployed its results have to be evaluated continuously in order to understand items like:
  - if it is delivering the expected outcomes;
  - if outcomes are bringing truly values;
  - the cost of the big data model and the effectiveness of the infrastructures;
  - how the infrastructures are supporting the model.

The whole big data modeling framework, observed from the processes and technologies perspective is sketched in fig.17.

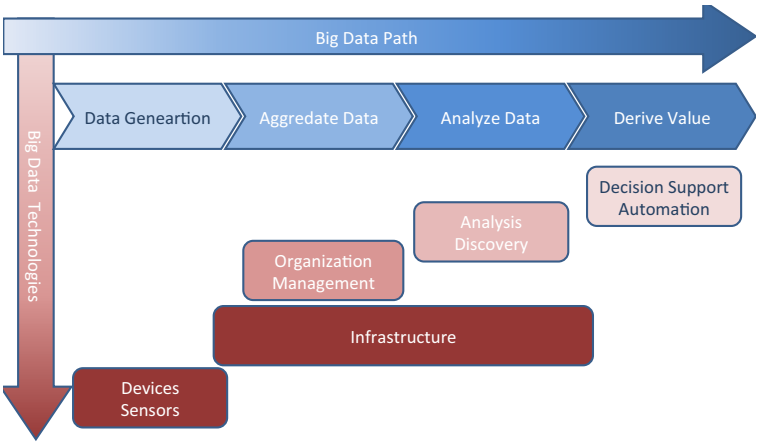


Fig. 17. Big data modeling framework: process and technologies

7 Conclusion

In the previous sections, we have discussed about big data and of their implications in terms of benefits and technological evolution. Now it appears clearer that big data is a great opportunity bringing plenty of challenges and even risks. Organizations could use big data in order to improve their competitiveness, improve their efficiency and manage systemic risks. In order to take those strategic advantages they should have to change their data management systems, their internal processes and even their culture. In term of IT infrastructure, who wants to take advantages of big data has to deal with a large amount of information and mostly with several kind of data sources and formats, to be carefully considered in a properly crafted big data model. Then he has to adopt the right analysis and visualizations tools, tightly coupled with the end-user applications, to properly consume the outcomes. Consequently, achieving a correct alignment between IT and business is a very critical factor in ensuring success. Several departments can be involved in the big data project but the initiative should pervade the whole organization according to a holistic approach.

References

1. International Telecommunication Union: World telecommunication/ict indicators database, 16th edn. (2012)
2. YouTube: Statistics (November 2013), `http://www.youtube.com/yt/press/statistics.html`
3. Brumfiel, G.: Down the petabyte highway. Nature 469(20), 282–283 (2011)
4. Lefevre, C.: Lhc: the guide (January 2008), `http://cds.cern.ch/record/1092437/files/CERN-Brochure-2008-001-Eng.pdf`
5. Open Government Initiative: Open government data, `http://opengovernmentdata.org/`

6. McKinsey Global Institute: Big data: The next frontier for innovation, competition, and productivity (2011), http://www.mckinsey.com/insights/business_technology/big_data_the_next_frontier_for_innovation
7. McKinsey Global Institute: Disruptive technologies: Advances that will transform life, business, and the global economy (2013), http://www.mckinsey.com/insights/business_technology/disruptive_technologies
8. Loecher, M., Jebara, T.: Citysense: Multiscale space time clustering of gps points and trajectories. In: Proceedings of the Joint Statistical Meeting (2009)
9. IDC iView: Big data, bigger digital shadows, and biggest growth in the far east (2012), http://www.emc.com/collateral/analyst-reports/idc-the-digital-universe-in-2020.pdf
10. Meeker, M., Wu, L.: Kpcb internet trends (2013), http://www.kpcb.com/insights
11. Bonwick, J., Ahrens, M., Henson, V., Maybee, M., Shellenbaum, M.: The zettabyte file system. In: Proc. of the 2nd Usenix Conference on File and Storage Technologies (2003)
12. Nelson, M.R.: Lzw data compression. Dr. Dobb's Journal 14(10), 29–36 (1989)
13. Welch, T.A.: A technique for high-performance data compression. Computer 17(6), 8–19 (1984)
14. Ziv, J., Lempel, A.: Compression of individual sequences via variable-rate coding. IEEE Transactions on Information Theory 24(5), 530–536 (1978)
15. Gailly, J.L., Adler, M.: The gzip compressor (1999), http://www.gzip.org/
16. Burrows, M., Wheeler, D.J.: A block-sorting lossless data compression algorithm (1994)
17. Seward, J.: The bzip2 algorithm (2000), http://sources.redhat.com/bzip2
18. Wallace, G.K.: The jpeg still picture compression standard. Communications of the ACM, 30–44 (1991)
19. Boutell, T.: Png (portable network graphics) specification version 1.0 (1997)
20. Schmuck, F.B., Haskin, R.L.: Gpfs: A shared-disk file system for large computing clusters. In: FAST, vol. 2, p. 19 (2002)
21. Leavitt, N.: Will nosql databases live up to their promise? Computer 43(2), 12–14 (2010)
22. Copeland, G.P., Khoshafian, S.N.: A decomposition storage model. ACM SIGMOD Record 14(4), 268–279 (1985)
23. Stonebraker, M., Abadi, D.J., Batkin, A., Chen, X., Cherniack, M., Ferreira, M., Lau, E., Lin, A., Madden, S., O'Neil, E., et al.: C-store: a column-oriented dbms. In: Proceedings of the 31st International Conference on Very Large Data Bases, pp. 553–564. VLDB Endowment (2005)
24. Abadi, D.J., Madden, S.R., Hachem, N.: Column-stores vs. row-stores: How different are they really? In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 967–980. ACM (2008)
25. Crockford, D.: The application/json media type for javascript object notation (json) (2006)
26. Lakshman, A., Malik, P.: Cassandra: a decentralized structured storage system. ACM SIGOPS Operating Systems Review 44(2), 35–40 (2010)
27. Chang, F., Dean, J., Ghemawat, S., Hsieh, W.C., Wallach, D.A., Burrows, M., Chandra, T., Fikes, A., Gruber, R.E.: Bigtable: A distributed storage system for structured data. ACM Transactions on Computer Systems (TOCS) 26(2) (2008)

28. Auradkar, A., Botev, C., Das, S., De Maagd, D., Feinberg, A., Ganti, P., Gao, L., Ghosh, B., Gopalakrishna, K., Harris, B., et al.: Data infrastructure at linkedin. In: 2012 IEEE 28th International Conference on Data Engineering (ICDE), pp. 1370–1381. IEEE (2012)
29. Gupta, P., Goel, A., Lin, J., Sharma, A., Wang, D., Zadeh, R.: Wtf: The who to follow service at twitter. In: Proceedings of the 22nd International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, pp. 505–514 (2013)
30. DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, P., Vogels, W.: Dynamo: amazon's highly available key-value store. In: SOSP, vol. 7, pp. 205–220 (2007)
31. Cooper, B.F., Ramakrishnan, R., Srivastava, U., Silberstein, A., Bohannon, P., Jacobsen, H.A., Puz, N., Weaver, D., Yerneni, R.: Pnuts: Yahoo!'s hosted data serving platform. Proceedings of the VLDB Endowment 1(2), 1277–1288 (2008)
32. Chodorow, K.: MongoDB: the definitive guide. O'Reilly (2013)
33. Dean, J., Ghemawat, S.: Mapreduce: simplified data processing on large clusters. Communications of the ACM 51(1), 107–113 (2008)
34. White, T.: Hadoop: the definitive guide. O'Reilly (2012)
35. Shvachko, K., Kuang, H., Radia, S., Chansler, R.: The hadoop distributed file system. In: 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), pp. 1–10. IEEE (2010)
36. George, L.: HBase: the definitive guide. O'Reilly Media, Inc. (2011)
37. Thusoo, A., Sarma, J.S., Jain, N., Shao, Z., Chakka, P., Anthony, S., Liu, H., Wyckoff, P., Murthy, R.: Hive: a warehousing solution over a map-reduce framework. Proceedings of the VLDB Endowment 2(2), 1626–1629 (2009)

Big Data, Unstructured Data, and the Cloud: Perspectives on Internal Controls

David Simms

Haute Ecole de Commerce, University of Lausanne, Switzerland
david.simms@unil.ch

Abstract. The concepts of cloud computing and the use of Big Data have become two of the hottest topics in the world of corporate information systems in recent years. Organizations are always interested in initiatives that allow them to pay less attention to the more mundane areas of information system management, such as maintenance, capacity management, and storage management, and free up time and resources to concentrate on more strategic and tactical issues that are commonly perceived as being of higher value. Being able to mine and manipulate large and disparate datasets, without necessarily needing to pay excessive attention to the storage and management of all the data that are being used, sounds in theory like an ideal situation. A moment's consideration reveals, however, that the use of cloud computing services, like the use of outsourcing facilities, is not necessarily a panacea. Management will always retain responsibility for the confidentiality, integrity, and availability of its applications and data, and being able to develop the confidence that these issues have been addressed. Similarly, the use of Big Data approaches offers many advantages to the creative and the visionary, but such activities do require an appropriate understanding of risk and control issues.

1 Introduction

This chapter will set out the risks related to the management of data, with particular reference to the traditional security criteria of confidentiality, integrity, and availability, in the contexts of the wider use of unstructured data for the creation of value to the organization and of the use of Big Data to gain greater insights into the behaviors of markets, individuals, and organizations. Of particular interest are the questions of identifying what data are held internally that could be of value and of identifying external data sources, be these formal datasets or collections of data obtained from sources, such as social networks, for example, and how these disparate data collections can be linked and interrogated while ensuring data consistency and quality.

Much of the current debate around big data technologies and applications concerns the opportunities that these technologies can provide and where issues of security and management are addressed; there is a tendency for these to be considered somewhat in isolation. This chapter will set out the risks, both to the owners and the subjects of the data, of the use of these technologies from the perspectives of security, consistency, and compliance. It will illustrate the areas of concern, ranging from internal

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_13

requirements for proper management to external requirements on the part of regulators, governments, and industry bodies. The chapter will discuss the requirements that will need to be met in respect of internal control mechanisms and identify means by which compliance with these requirements can be demonstrated, both for internal management purposes and to satisfy the demands of third parties such as auditors and regulators.

The chapter will draw upon the author's wide experience of auditing the IT infrastructures of organizations of all sizes in describing the processes for identifying relevant risks and designing appropriate control mechanisms. The chapter will contain discussion of the standard frameworks for implementing and assessing controls over IT activities, of information processing and security requirements, objectives and criteria, and of monitoring, testing evaluating, and testing control activities. The author will also apply his insights into the strategies being followed, and initiatives being considered, by a range of organizations and corporations in order to illustrate how risks are changing as technologies change and how control activities need to develop in response. This analysis will be based upon the results of a survey performed within Switzerland in 2011 and 2012 that set out to establish an overview of how organizations viewed and understood both cloud technologies and the nature and value of the data that they held, and what impact these concepts and opportunities would have on their strategies, policies and procedures.

2 Preliminary Concepts and Definitions

Many of the terms used in this chapter are reasonably recent coinages and definitions can still be flexible and varied.

For the purposes of this chapter, we will follow Bernard Marr [1] with the definition that "Big data refers to our ability to collect and analyze the vast amounts of data we are now generating in the world. The ability to harness the ever-expanding amounts of data is completely transforming our ability to understand the world and everything within it." The fundamental idea is of the accumulation of datasets from different sources and of different types that can be exploited to yield insights.

According to an article by Mario Bojilov in the ISACA Now journal [2], the origins of the term come from a 2001 paper by Doug Laney of Meta Group. In the paper, Laney defines big data as datasets where the three Vs—volume, velocity, and variety—present specific challenges in managing these data sets.

Unstructured data as a concept has been identified and discussed since the 1990s but finding a definitive and all-encompassing definition of what this might be is surprisingly difficult. Unstructured data have been defined vaguely positively by Manyika et al [3] as "data that do not reside in fixed fields. Examples include free-form text (e.g., books, articles, body of e-mail messages), untagged audio, image and video data. Contrast with structured data and semi-structured data" which provides a definition but one which is rather more in respect of characteristics that the data do not possess rather than what they are.

To complicate matters slightly, Blumberg and Atre [4] discussed the basic problems inherent in the use of unstructured data a decade ago. They wrote "The term unstructured data can mean different things in different contexts" and that "a more accurate term for many of these data types might be semi-structured data because, with the exception of text documents, the formats of these documents generally conform to a standard that offers the option of meta data" (p. 42).

Cloud computing is described by NIST as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" [5], while the Cloud, according to Manyika et al, is "a computing paradigm in which highly scalable computing resources, often configured as a distributed system, are provided as a service through a network." For corporate users, the immediate impact of using cloud services is not necessarily clearly distinguishable from that of using traditional outsourced services, with services and/or data being available through an external network connection. The key distinction is that unlike conventional outsourcing, where typically the service provider contracts to store, process, and manage data at a specific facility or group of facilities, in the context of the cloud the data could be stored anywhere, perhaps split into chunks, with no external visibility over how that was organized.

For the individual, private user, the cloud, represented by such services as Dropbox, iCloud, or Google Services, for example, is exactly as its name suggests, a virtual and distant and slightly opaque facility by which services are provided without significant identifying features or geographical links.

Internal Controls are the mechanisms – the policies, procedures, measures, and activities – employed by organizations to address the risks with which they are confronted. To define a little further, these activities fall into the framework of control objectives are the specific targets defined by management to evaluate the effectiveness of controls; a control objective for internal controls over a business activity or IT process will generally relate to a relevant and defined assertion and provide a criterion for evaluating whether the internal controls in place do actually provide reasonable assurance that an error or omission would be prevented or detected on a timely basis [6].

The traditional triad of information security objectives consists of Confidentiality, Integrity, and Availability [7].

- Confidentiality is the prevention of the unauthorized disclosure of information, providing the necessary level of security.
- Integrity is the prevention of the unauthorized modification of information, assuring the accuracy, and integrity of information.
- Availability is the prevention of the unauthorized withholding of information or resources, ensuring the reliable, and timely access to the information for authorized users.

To give a concrete example of how control objectives and internal controls fit together, an organization might be concerned about unauthorized access to its data. The control objective might be to ensure the confidentiality of the data, and one of many

possible control activities might be to perform regular reviews of the appropriateness of user access rights at the application level, to ensure that there are no redundant or unallocated accounts. Clearly in a well-controlled environment there will be a number of internal controls relating to each activity and each objective, and management will draw their comfort from the combined effectiveness of these controls.

At the same time there are a number of other objectives in respect of information security that may be of greater of lesser significance for different organizations, depending on the data they hold and process and the sectors in which they operate. These areas, which are included in standards such as those published by the ISO [8] and NIST [9], among others, include:

- Authenticity / authentication: having confidence as to the identity of users, senders, or receivers of information;
- Accountability: ensuring that the actions of an entity can be traced uniquely to that entity.
- Accuracy: having confidence in the contents of the data;
- Authority: the means of granting, maintaining and removing access rights to data;
- Non-repudiation: making it impossible for the person or entity who has initiated a transaction or a modification of data to deny responsibility after the event;
- Legality: knowing which measures are appropriate for the legal frameworks within which an organization is operating.

From a point of view of completeness, it should also be mentioned that there are a number of other classifications of information security criteria. In 2002, for example, Donn Parker proposed an extended model encompassing the classic CIA triad that he called the six atomic elements of information [10]. These elements are confidentiality, possession, integrity, authenticity, availability, and utility. Similarly, the OECD's Guidelines for the Security of Information Systems and Networks, first published in 1992 and revised in 2002 [11], set out nine generally accepted principles: Awareness, Responsibility, Response, Ethics, Democracy, Risk Assessment, Security Design and Implementation, Security Management, and Reassessment.

3 State of the Art

Cloud computing is a paradigm in computing that has emerged from the spread of high-speed networks, the steady increase in computing power, and the growth of the Internet. The interconnection of resources that may be separated by significant distances is allowing users access to what appears to be a single resource. As a result, there is an opportunity to outsource resource-intensive or resource-specific tasks to service providers who will deliver the service for a fee, often based on consumption, rather than developing and maintaining the infrastructure and competences in-house.

Neither of the central ideas in this model, distributed computing or outsourcing, is new. Distributing resources within and across networks has been performed since the days of mainframe computers, where data were entered on remote terminals and processed centrally, while outsourcing of computing activities and services has taken place for many years for strategic, operational, and financial reasons.

In technical literature reference is often made to cloud computing, grid computing, and utility computing; there is no real consensus over whether there is a distinction to be made between the three and, if so, whether the distinctions are clear or are rather subtle. In general, though, the models are identified by features, such as ubiquitous access, reliability, scalability, virtualization, the exchangeability of and independence from location considerations, and cost-effectiveness [12].

A key area of growth is that of the types of service that can be offered by providers. Historically providers typically offered remote data processing, storage, and systems management as major services, but the newer paradigm is to group offerings together as types of services: Software as a Service (SaaS), Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Service-Oriented Architecture (SOA). The underlying principle is that every facet of computing activities can be offered as services to consumers to be paid for on a usage basis. Computing is offered as a utility with billing based on consumption, with a corresponding shift toward IT spending being classified as expenditure on a service rather than infrastructure investment [13].

This paradigm has been likened to the electrical power grid, both in the impact it has on social and industrial development and because of its nature: elements of the grid can be owned and operated by different entities in different locations and might not share any physical characteristics, but the users, those who pay for the service, will only see a single interface or point of contact and will consume a consistent and homogenous service [14].

3.1 The Advantages of Cloud Services

The advantages of such services for certain users are, at least in principle, clear. The users need not worry about maintaining the application infrastructure, with all the operating system, database, and application patches and upgrades that are necessary. Nor need the users worry about data management practices such as backups or transfer between machines and environments. Both individuals using office-style applications and remote storage services and large corporations using large-scale applications remotely will see the benefits of avoiding questions of ongoing software compatibility and upgrade paths. Essentially these services are seen as an opportunity to avoid having to deal with certain aspects of the complexity involved in managing a technological infrastructure.

For corporate users there are also the advantages related to the use of scalable architectures. Instead of making potentially significant capital investments in hardware and software that might never be fully exploited and thus represent an unnecessary cost to the business, management will be tempted to subscribe to the model of being able to request and exploit additional resources when required, and for as long as required, on something like a Pay-As-You-Go basis. This has the potential to allow a much more accurate and dynamic management of costs while still permitting absolute flexibility in the access to and use of resources.

The advantages for service providers of operating in this sector also seem to be well established. Once a data center has been established and the reasonably fixed costs of operation have been established and incorporated into the business model, the

variable costs of service provision and marginal costs related to the acquisition of additional clients or providing additional services or additional capacity to existing clients should be straightforward to manage and reasonably simple to recover (and exceed, of course) through appropriate pricing. Thus the provision of such services can be viewed as a potentially lucrative business, with important initial investment but steady and permanent future revenue streams.

Conceptually, the techniques of cloud computing are not significantly different to those of traditional outsourcing of IT services. Many service providers offer outsourced data processing, system management, monitoring, and control services and these services are very popular with many organizations who either do not wish to have internal IT services and competences for organizational reasons or prefer to outsource for financial or logistical purposes.

Key elements of any outsourcing agreement are the contract terms and the service level agreements. With these terms, it is clear to both parties which services are being provided, what resources are being made available, how these resources are being managed, and how the quality of service can be evaluated and managed.

Many structured cloud services will provide such terms and conditions but might not be able to specify every element of interest to the customer, such as the precise location where data are stored or processed.

It is in the nature of any kind of outsourcing or service provision activity that both parties will wish to maximize their revenues and benefits from the agreement while at the same time accepting the minimum level of responsibility for addressing the risks involved and for handling any issues that arise. In the context of cloud computing, customers need to pay particular attention to the clear definition of roles and responsibilities in order to try to avoid situations of blame-shifting and cost avoiding should problems subsequently arise.

A key driver for the use of cloud facilities is costs: organizations might not wish to tie up capital in IT infrastructure that might not be used to capacity and which might only have a short life before obsolescence, when there might exist the possibility of renting services as a regular P&L charge. Along with the resource and competency questions, which of course also incur ongoing costs and can be expensive to update or replace, this has long been a prime mover for outsourcing services. Experience in the domain of outsourcing, however, reveals that the cost savings may not always be as significant as hoped for. As mentioned above, the prudent management team will look to ensure that its systems and data are secure and reliable by implementing additional internal controls to generate evidence that there are no weaknesses in the service provider's controls that can be or are being exploited. Typically, these internal controls will take the form of data and transaction analysis to identify exceptions, or the appointment of specialist staff that can manage the relationship with the service provider and ensure that trends are identified, that service levels are maintained, and that issues are identified, reported, and rectified. Very often the costs associated with implementing and operating such additional internal controls in an effective and robust manner are such that they can eat into the margins created by the whole outsourcing initiative.

3.2 The Disadvantages of Cloud Services

If the advantages for users of cloud services, as set out above, appear to be obvious, then the disadvantages are equally clear, particularly if viewed from a management and control perspective. The management of organizations always retains responsibility for the security and availability of their systems and data, in particular for the three key attributes of confidentiality, integrity, and availability. Ensuring that these attributes are understood, managed, and evaluated has traditionally been a significant challenge in systems management, even when systems and data are kept within a well defined and efficiently policed perimeter. Once this perimeter is extended outside the sphere of direct control of relevant management, the challenge becomes increasingly difficult.

In an article published in the Gartner blogs [15], Thomas Bittman argues that the widely used analogy of cloud computing to provision of electricity or water supplies is not particularly illustrative, for two reasons: "(1) Computing is a rapidly evolving technology, and (2) Service requirements vary widely for computing. Electricity production and distribution hasn't evolved much since the invention of AC that made distance distribution possible. How many forms of water are needed around the world? It's H2O – maybe it can be potable, purified, or come at a special temperature, but it's still pretty basic stuff."

His analogy is that of transmitted music: radio broadcasts of music began in 1916 and phonograph cylinders were used for storage, with the idea being that people would not wish to bear the expense of storing their own copies of music when it was available over the airwaves. But technologies have advanced, both for the broadcast and transmission of music and for the local storage and consumption. Individuals continue to be prepared to pay a premium, for infrastructure, material and content, for a quality and rapidity of service.

The choice between broadcast and local access to music is the same as the cloud computing question: it will depend on a number of factors and requirements that are constantly evolving and ultimately become a question of the best balance between costs, risks, and quality. A key element in the cloud computing choice will be assessing the development of requirements and adopting a strategy that will maximize the Return on Investment, however that is calculated.

Such analogies do reach their limits, however, because they do not consider the specific nature of the service provided. When plugging in a laptop in a foreign hotel room, the consumer broadly speaking does not care who has provided the electricity and the identity of the provider has no impact on the use of the service. Using cloud services is fundamentally different, though, for as soon as a provider is storing data or performing processing for a client, there emerge questions of the confidentiality and integrity of the systems and data as well as the availability of services.

A further aspect in which the analogy breaks down is in the area of the difficulty of changing approach once decisions have been taken. There is no question of not continuing to use electricity to power devices, but there will always be questions about the most efficient way to have access to and use IT infrastructure and resources. Moving from one cloud solution to another is likely to prove as complicated, if not more, than moving from one traditional outsourcing provider to another.

3.3 Data Storage and the Cloud

The opportunity to exploit the storage potential of the cloud can feasibly be viewed as an encouragement to organizations to place less priority on good practices in relation to data management. As storage space increases, so does the tendency simply to store all data rather than to classify, prioritize, archive, and delete them as appropriate, and there is nothing in the principles of cloud computing to reverse this tendency.

It is therefore reasonable to envisage situations in which individuals and organizations simply do not know what data they have placed in the cloud. There may be multiple copies of the same documents and datasets. There may be inconsistent and incoherent datasets. There may be quantities of unstructured data – data extracted from central systems and databases that are not in standard, easily recognized, and easily classified structures – that by their nature have not been classified and evaluated.

Of course, the question of unstructured and unmanaged data is not unique to the cloud: it exists in virtually all IT environments. Where it becomes particularly significant in the cloud environment, however, is as a consequence of the lack of visibility the data owners have over their data. Access to in-house, and properly secured outsourced data collections, can be defined, logged, and reviewed, at least in theory given sufficient resources and sufficient motivation on the part of the data owners. Once the data are in the cloud, however, the owners have very little visibility and control over the security of their data [16]. If this weakness is exacerbated by an absence of real knowledge of what data are out there, the risk of data loss and disclosure is increased.

The classical model for many businesses and other large organizations is to have one or several centralized key systems in which all of the organization's financial and operational activities are recorded. Typically, the data used in key applications and upon which users depend are stored in structured, centralized and at least theoretically well-controlled databases.

There is also, however, widespread use of noncentralized data repositories. Individual departments or users may have their own specific applications that do not fall into the overall organization-wide systems landscape and thus are subject to different standards and procedures for management and control. This is not to say that these systems and data are necessarily badly controlled, simply that management cannot necessarily be certain that standard policies and procedures are being applied.

3.4 The Use and Frequency of Unstructured Data

In the modern business environment great use is made of unstructured data in a variety of contexts. Typically users will extract data from central databases, often those underlying ERP systems, and then manipulate these data in a variety of ways as part of their business activities. Such data are thus often found in spreadsheets, user-built databases, and desktop or departmental server-based applications. These data can also be found in text files, pdfs, and even multimedia formats, depending on the use to which they have been put.

From an internal controls perspective the presence and use of unstructured data can pose numerous problems in respect of the confidentiality and integrity of data, two-thirds of the famous "CIA" triad of information security objectives (the third being availability). When data are located in a structured central database, they can, at least in theory, be controlled, managed, verified, and secured. Once extracted from the database, though, they can easily escape the internal control environment [17]. They can be used for decision taking without the assurance that they are still current, complete, or valid. They can also, depending on the security measures in place and the efficiency with which these are enforced, leave the organization easily, typically on the USB media that have become ubiquitous, on laptop hard drives, or even as attachments to emails.

4 Problems, Issues, and Challenges

There are several underlying problems related to the management of multiple datasets and of collections of unstructured data.

The fundamental problems related to the management and control of data stored in the Cloud are not dissimilar to those encountered when using outsourcing facilities or third-party service providers. Simply stated, as soon as data or applications are moved outside the perimeter of integrated and monitored internal controls, management have less control over their data.

Typically in an outsourced environment there are a number of ways for the organization purchasing the services to acquire at least a reasonable level of control and assurance. These can be described as obtaining comfort from audit procedures performed by the organization itself or by an audit entity specifically mandated to audit on its behalf; by obtaining a service auditor's report for the environment in question; or by implementing, performing, and monitoring the success of a series of internal control activities.

4.1 Self-performed Audit Procedures

The first of these methods is the most direct and is identical in form and process to the use of internal audit functions to evaluate and report upon business operations that are carried out in-house, as well as the use of the findings of external audits, where internal controls are often evaluated in the context of a controls-based audit. The audit team works with the organization's management to define scope and timeframe, documents the business procedures, identifies the key control activities, assesses these for both **design effectiveness** (whether as designed the control activities do address the risks to which they relate and the control objectives to which they correspond), and **operating effectiveness** (whether the control activities are actually working as designed, with necessary inputs being received, the control being performed, and appropriate conclusions being drawn and actions undertaken). Based on the results of these audit activities, management will be in a position to draw informed conclusions on whether or not internal controls are working as intended.

The right to perform such audits will need to be specified in the contract for service provision.

The difficulties of performing such reviews in an outsourced environment are many. First, the organization may not have an appropriately skilled, experienced, or available audit department – it is frequently the case that internal audit functions tend to have greater expertise in, and requirements to focus on, financial management issues, such as the use and disposal of assets, Value For Money, and purchase and inventory management. Detailed technical skills in the field of information systems are more likely to be found in larger and more IT-dependent organizations.

Secondly, it might be prohibitively expensive to mandate a third party to perform the necessary audit on their behalf.

Thirdly, it might be logistically difficult to identify all the areas in which processing or data storage or IT management tasks are performed at the service provider, especially if multiple sites in multiple locations are used. This will add to the complexity of scoping and scheduling such an audit.

Fourthly, but far from being the least significant issue, there is an impact of being audited on the service provider. In order to respond to audit questions, staff, and documentation need to be made available, and then resources need to be provided to assist with the testing of individual controls, extracting system information and reports, and explaining their contents. This can have multiple impacts on the service provider, taking up human and technical resources, distracting staff from their daily activities, and using office space. If a service provider were to allow all of its clients to pay audit visits independently and at times that suited them, it is not difficult to imagine that this could cause significant disruption to the provider's activities.

4.2 Third-Party (Service) Audit Procedures

Many service providers thus prefer the second method of allowing their clients to obtain comfort over internal controls: the service audit report. In this situation, the service provider itself mandates an external auditor to review its internal controls and provide an opinion on the effectiveness and efficiency of these controls [18]. This report is then made available to the service provider's customers who can incorporate its findings into their own evaluation of the control environment.

The advantages of such an approach for the service provider are clear. They meet their requirements to provide reliable information about their control environment, while avoiding the inefficiencies of having multiple audit teams visiting their premises and having to provide repeated briefings, explanations, and copies of documentation. If the audit partner chosen has a size and structure that allows consistency of team membership and a minimum of rotation, there will also be the advantage of familiarity year to year with business practices and documentation standards as applied by the service provider, which will also increase the efficiency of the audit process.

The report issued by the auditor can take many forms, but for many years the de facto international standard to be followed was the Statement on Auditing Standards No. 70: Service Organizations, commonly abbreviated as SAS 70.

This standard specifies two kinds of reports, named Type I and Type II. A Type I service auditor's report includes the service auditor's opinion on the fairness of the presentation of the description of controls that had been placed in operation by the service organization and on the suitability of the design of the controls to achieve the specified control objectives (thus corresponding to the concept of design effectiveness

as discussed above). A Type II service auditor's report includes the information contained in a Type I service auditor's report and includes the service auditor's opinion on whether the specific controls were operating effectively during the period under review. Because this opinion has to be supported by evidence of the operation of those controls, a Type II report therefore also includes a description of the service auditor's tests of operating effectiveness and the results of those tests.

SAS 70 was introduced in 1993 and effectively superseded in 2010 when the Auditing Standards Board of the American Institute of Certified Public Accountants (AICPA) restructured its guidance to service auditors, grouping it into Statements on Standards for Attestation Engagements (SSAE), and naming the new standard "Reporting on Controls at a Service Organization". The related guidance for User Auditors (that is, those auditors making use of service auditors' reports in the evaluation of the business practices or financial statements of organizations making use of the facilities provided by service providers) would remain in AU section 324 (codified location of SAS 70) but would be renamed Audit Considerations Relating to an Entity Using a Service Organization. The updated and restructured guidance for Service Auditors to the Statements on Standards for Attestation Engagements No. 16 (SSAE 16) was formally issued in June 2010 and became effective on 15 June 2011. SSAE 16 reports (also known as "SOC 1" reports) are produced in line with these standards, which retain the underlying principles and philosophy of the SAS 70 framework. One significant change is that management of the service organization must now provide a written assertion regarding the effectiveness of controls, which is now included in the final service auditor's report [19].

Internationally, the International Standard on Assurance Engagements (ISAE) No. 3402, *Assurance Reports on Controls at a Service Organization*, was issued in December 2009 by the International Auditing and Assurance Standards Board (IAASB), which is part of the International Federation of Accountants (IFAC). ISAE 3402 was developed to provide a first international assurance standard for allowing public accountants to issue a report for use by user organizations and their auditors (user auditors) on the controls at a service organization that are likely to impact or be a part of the user organization's system of internal control over financial reporting, and thus corresponds very closely to the old SAS 70 and the American SSAE 16. ISAE 3402 also became effective on 15 June, 2011.

The importance of understanding the related guidance for user auditors (which also applies, by extension, to user management) is critical. When a service audit report is received, the reader need to follow a number of careful steps in order to be certain that the report is both useful and valid before beginning to draw any conclusions from it.

These steps include:

1. Confirming that the report applies to the totality of the period in question. This is of particular importance when using a service audit report in the context of obtaining third-party audit comfort for a specific accounting period, but also applies to more general use of the report. If management's concern is over the effectiveness of internal controls for the period from 1 January to 31 December 2012, say, and the audit report covers the period from 1 April 2012 to 31 March 2013, how valid is it for management's purposes and how much use can they make of it? In the simplest of cases, a prior period report will also be available that will provide the

necessary coverage, but in other circumstances this may not be the case and management will have to turn to other methods to acquire the comfort that they need. These could include obtaining representations from the service manager that no changes had been made to the control environment during the period outside the coverage of the audit report and that no weaknesses in internal control effectiveness, either design or operational, had been identified during that period. Other methods could involve the performance and review of internal controls within the client organization, a subject to which we will return below.

2. Confirming that all the systems and environments of operational significance to the organization were included in the scope of the audit report. Management will need to have an understanding of the platforms and applications that are being used for their purposes at the service provider, including operating systems, databases, middleware, application systems, and network technologies, and be able to confirm that these were all appropriately evaluated. Very often in the case of large service providers a common control environment will exist, under which they apply identical internal control procedures to all of their environments: if the auditors have been able to confirm that this is the case, then it is not inappropriate for them to test the internal controls in operation around a sample of the operating environments, and management can accept the validity of their conclusions without needing the confirmation that their particular instance of the database, for example, had been tested. Should the coverage of the audit not meet management's requirements, again management would need to evaluate the size and significance of the gap and consider means by which they could obtain the missing assurance.
3. Understanding the results of the work done and the significance of any exceptions or weaknesses noted by the auditors. Generally speaking, if the work has been performed to appropriate standards and documented sufficiently, and if the conclusions drawn by the auditors are solidly based on the evidence, this step should be reasonably straightforward. Management should, however, guard against skipping to the conclusion and, if the report contains it, the service provider's management attestation, and blindly accepting the absence of negative conclusions as being sufficient for their purposes. Each weakness in the design or the operation of controls should be considered, both individually and cumulatively, to identify any possible causes for concern. This is because it is possible that weaknesses identified during the audit, considered to be insignificant within the overall framework of internal controls could potentially be significant in respect of the specific circumstances (use of systems and combination of technologies, for example) of one particular customer.

4.3 Other Procedures

Should such an independent audit report not be available, or only provide partial coverage of the control environment or the period in question, and should it in additional be impossible or impractical to the organization to perform their own audit procedures at the service provider's premises, a third way of obtaining comfort over operations is needed. This method can be summarized as consisting of two groups of activities: internal controls performed within the organization over the activities of the service provider, or audit procedures designed to evaluate the completeness, accuracy, and integrity of the service provider's processing and outputs.

First, internal controls can be designed that allow local management to monitor and evaluate the activities of a third party. These can take the form, for example, of procedures to track the responses of the service provider to requests for changes: if there is a process by which the customer asks the service provider to change access rights to an application or a datastore to correspond to the arrival or departure of a member of staff, management can track these change requests, and the responses of the service provider in order to ensure that the correct actions have been undertaken.

Very often the contract terms with the service provider will include regular meetings and reporting mechanisms through which the provider will present status updates, usually in the form of progress against KPIs and lists of open points, and management can ask questions and ensure that everything is under control. These meetings can form the central points of control activities for management, as a structured means of ensuring that they are monitoring the performance of the service provider in a regular and consistent way, observing long-term trends, and identifying anomalies.

Secondly, audit procedures can be designed along similar principles to the above, based on the expected results from the service provider's activities. An example of this would be extracting at the period end a list of user access rights to the organization's applications and comparing these to expectations, expectations based on management's understanding of the access requirements and on the instructions given during the year to create, modify, or delete access rights. If the rights correspond, this can provide a layer of assurance to management that both the service provider's internal procedures and controls are operational and that access to their applications and data is being appropriately managed.

With an appropriately selected range of internal controls and audit procedures, management can, therefore, obtain a certain level of comfort over the existence and quality of the control environment in place at the service provider.

4.4 Timescales and Logistical Concerns

It is important to recognize that the process of achieving the confidentiality, integrity, and availability of data is likely to be lengthy. Data growth is one of the most challenging tasks in IT and business management, with increasing quantities of data being generated in-house within organizations and being acquired from external sources. It is, therefore, essential to develop a structured plan for overall data management within with the steps necessary for data accumulation, security, and transfer can be carried out in a systematic and reliable manner.

In practical terms, management needs to take a number of operational decisions at an early stage in the planning process. Even before decisions can be taken about which data should be retained within an organization's infrastructure and which should be outsourced, more fundamental decisions need to be taken in respect of how, where, and by whom data will be cleaned, structured, collated, error checked, completed, or even clearly marked for deletion or a separate archiving process.

The "how" aspect will almost inevitably be addressed by a combination of automated tools and manual intervention. Software will be necessary be processing and transforming large quantities of data, but human intervention will be essential for defining and implementing parameters, reviewing output, and making ongoing decisions.

The "where" aspect throws up a question that anticipates, rather, the questions of data security that the move toward outsourcing and third-party storage also throws up.

Arguments could be made on the grounds of costs and logistics that the data preparation process should be performed offsite at a third-party site, on purpose-built infrastructure and away from the organization's internal networks. This would be in order, for example, to prevent excessive strain on resources caused by intensive processing and by large quantities of data passing across the network. Such a solution would, however, introduce the additional complication of ensuring adequate security over the data once the datasets have left the organization's security perimeter.

The "by whom" aspect will concern the use of internal resources, insofar as they can be spared, and external resources. Depending of the amount and the nature of the data to be processed, it may be appropriate to bring in resources from outside to perform aspects of the work. Internal resources will always be necessary, however, both from IT to provide technical input, and from the business side as users who understand where the data come from, what they represent, and how they relate to each other. A common failing in any data cleaning or migrating exercise is to view it as a purely technical procedure, whereas in practice the input from experienced and knowledgeable data owners and end-users is critical.

The timescale for such a project will depend on several factors, including the quantity and the nature of the data to be prepared, the availability of resources, and the priority set by senior management for the process. Experience of such projects would indicate that management should be setting their expectations in terms of months rather than weeks, however, and that if data quality and security are really expected, there is no scope for cutting corners.

5 Proposed Approach and Solutions

In order to gain an independent perspective on how the questions of cloud facilities and unstructured data were affecting organizations in Switzerland, the author performed a survey in 2011 and 2012.

5.1 Background to the Survey

50 questionnaires were sent out to contacts in 43 organizations across Switzerland, using the lead author's business experience to identify correspondents across a range of industries and sectors of activity who would be likely to respond. Completed questionnaires were received from 34 organizations, with three sending two responses and two sending three. The organizations that declined to respond did so on the grounds of confidentiality, not wishing to divulge details of their IT strategy or approach to security to a third party.

The organizations were selected in order to provide a wide cross-section of the range of IT environments and attitudes to the management of data and the use of new technologies. Of the organizations that did respond, five were publicly owned, eighteen privately, and the remaining eleven were public administrations or NGOs. The breakdown by organization size also shows variety: eight with less than fifty employees; ten with between fifty-one and one hundred; five with between one hundred and five hundred; and eleven with more than five hundred.

The rationale behind sending multiple questionnaires to the same organization was to attempt to discover whether there would be cases of poor communication of strategy or

developments within those organizations. It is the author's experience of large corporations in particular that there can be differences in the understanding of the overall strategy, the firm objectives, the initiatives undertaken, and the impact on users between top management, middle management, and IT management, for example.

5.2 The Survey

The questions in the survey were split into two sections, dealing with data security within the organization and with data storage in the cloud, as follows:

Table 1. Survey questions

Section A

Q1	Is there an overall policy concerning data management, security, and retention?
Q2	Is there awareness at the level of senior management and/or security management of the concept of "unstructured data"?
Q3	Has there been any assessment of whether the organization should be concerned, from security or efficiency perspectives, about the existence of unstructured data?
Q4	Has the organization established any guidelines on the classification of data according to their sensitivity, age, and relevance?
Q5	Has there been any structured attempt to identify and quantify the data stored around the organization outside centralized databases?
Q5A	If so, was this a manual process?
Q6	Does the organization have policies on the extraction, use, and storage of data by end-users?
Q6A	If so, is compliance with these policies monitored and enforced, and how?
Q7	Are there policies and/or restrictions in place of the use of removable storage media for file transfer or storage?
Q7A	If so, is compliance with these policies monitored and enforced and how?
Q8	Does the organization have any mechanisms for determining whether there have been breaches of security or confidentiality in respect of its sensitive data?

Section B

Q9	Is the organization using, or planning to use, cloud computing services for the storage of data?
Q10	If yes, have policies and guidelines been drawn up for the nature of data that can be stored in this way?
Q11	If cloud computing is being used, is the organization's approach based on the centralized management, monitoring, and retrieval of data, or do departments and/or individuals retain responsibility for their data?

5.3 Survey Results

Table 2. Survey results

Section A	Yes	No	Yes %	No %
Q1	32	9	78%	22%
Q2	22	19	54%	46%
Q3	17	24	41%	59%
Q4	9	32	22%	78%
Q5	6	35	15%	85%
Q5A	6	0	100%	0%
Q6	14	27	34%	66%
Q6A	5	9	36%	64%
Q7	12	29	29%	71%
Q7A	8	4	67%	33%
Q8	4	37	10%	90%
Section B				
Q9	39	2	95%	5%
Q10	6	33	15%	85%
Q11	7	32	18%	82%

5.4 Interpretation and Analysis of the Results

The results demonstrated two major trends in respect of unstructured data. The first was that although 78% of organizations reported having designed and implemented an overall policy concerning data management, security and retention, the exceptions being overwhelmingly small organizations with informal internal control structures, there was little systematic followup in terms of the management of unstructured data or in terms of managing and monitoring the use of data by users. 54% reported that senior management were aware of the issue of unstructured data; but only 41% reported that a risk analysis had been carried out to evaluate their exposure; 22% reported that guidelines for the classification of data had been developed and published; and only 15% reported having carried out a structured attempt to identify and quantify the data stored outside centralized databases. In each case it was a large multinational company that had undertaken such an initiative; interestingly, each one reported that the process had been largely manual.

From the perspectives of internal controls and good corporate management, many of these numbers are worryingly low. That more than three quarters of organizations report the existence of an overall policy in respect of data management is a reasonable starting point, but the lower numbers in respect of detailed data management indicate that few organizations had progressed further than the big picture, high-level aspects of data management at the time this survey was performed.

In respect of the management of user activities related to data handling, 34% reported having policies on the extraction, use, and storage of data by end users, and only 36% of these were able to report that compliance with these policies was monitored and enforced. Only 29% of organizations reported having policies or procedures in place concerning the use of removable storage media for file transfer or storage, while only 67% of this small subset were able to describe compliance mechanisms. Finally, only 10% of organizations were able to describe the existence of mechanisms for determining whether breaches of security or confidentiality in respect of sensitive data had occurred.

These rather low numbers suggest that organizations will have a significant amount of work to do in collating, cleaning, and preparing data. The responses suggest an overwhelming absence of effective internal control to date over data usage, and a lack of certainty over the nature, quality, and integrity of the datasets in use around organizations, factors that will automatically increase both the amount of scoping work that needs to be carried out and the quantity of detailed cleaning and tidying of data.

In the cases where more than one response was received from an organization, it was noted that end users were less aware of the existence of strategies and policies than senior management, a situation that is consistent with the author's long experience of auditing large organizations. A key difficulty in the implementation of internal controls within an organizations relates to ensuring that details of the control objectives and activities, their importance, their relevance, their nature and their follow-up, permeate sufficiently through the organization so that controls are operated effectively and efficiently, improving processes rather than hindering them, and with evidence of their performance and output being available for timely review and, if necessary, corrective actions.

In respect of the use of cloud computing facilities for the storage of data, 95% of the responses reported that the organization was using or was planning to use such facilities. The reasons most frequently given were cost management, flexibility, and a desire to streamline IT activities to concentrate on more value-added activities in-house. Of these organizations, however, only 15% had drawn up policies and guidelines concerning the nature of data that could be stored in this way, and only 18% were adopting an approach based on the centralized management, monitoring, and retrieval of data rather than leaving it to departments or individuals to manage.

Once again, the wide absence of overall policies and guidelines and the tendency as reported to adopt potentially uncoordinated approaches to project scope and management runs counter to established good practice in respect of internal controls. Without clear and enforced structure, inconsistency is likely to become a significant barrier to success. In addition, delegating down to departments or individuals increases the risks of decisions being taken without sufficient skills, experience, or perspective.

Overall it was possible to draw a clear distinction between large and small organizations and publicly and privately owned ones in respect of their approach to structured and formalized control environments. It was also possible to identify with high accuracy the responses received from publicly owned corporations and those from organizations subject to other strict and demanding controls requirements such as Sarbanes-Oxley or local industry or environment-specific regimes. It was also possible to identify organizations that had significant internal audit or internal controls functions, or that had been alerted to the risks involved in going through a process of discovery in the context of a legal dispute.

6 Summary Evaluations and Lessons Learned

The tentative conclusions that can be drawn from this very specific and targeted survey are the following.

First, the importance of having a structured and documented approach to data management and security is widely understood among the organizations surveyed, with a particularly positive attitude toward risk management and compliance from larger organizations and those subject to definite compliance regimes because of their ownership or industry. How this understanding actually translates into positive measures designed to ensure compliance is another question, however, and IT security managers in particular reported seeing greater enthusiasm for establishing policies within their organizations than for implementing and complying with the necessary procedures. There was also a question of priorities and resources raised by smaller organizations that did not feel that such policies corresponded to or were a part of their core daily activities.

Secondly, the concept of unstructured data and the particular challenges posed by such data is reasonably widely understood, but there has been little activity outside large publicly owned corporations to address the issue in any systematic way. In general IT departments had a good grasp of the nature and impact of the matter, and the

subject was frequently raised by internal and external auditors and by legal advisers, but it was rarely considered to be a subject of great priority by senior management.

Thirdly, even if thought has been given within some organizations to managing employee access to and extractions of sensitive data, in the majority of cases monitoring is weak and compliance cannot be ensured. Generally speaking, users who have access to data stored in centralized databases tend to have the ability and the opportunity, and frequently the encouragement, to extract those data and use them for analytical or reporting purposes. Once the data have been extracted, access controls around them are usually weaker, often being restricted to network or workstation access controls, and even these restrictions reach their limits once data are copied onto portable devices such as USB keys.

Fourthly, very few organizations are in a position of being able to detect reliably whether their security has been breached or their data compromised, even when all their systems and data are hosted and managed internally. Indeed, in respect of both security breaches and employee misuse of data, it was reported that incidents were typically identified either by chance or on the basis of information received, rather than on the basis of regular and reliable compliance measures. Of course, in practice being able to design, implement and monitor the operation of such control activities is frequently nontrivial and requires competence, resources, and careful planning. Whether organizations would be able to apply such controls to outsourced data, or be satisfied by the monitoring and reporting services provided by their cloud service provider, is the same question with an added layer of complexity.

Fifthly, the idea of cloud computing as a financially attractive option for outsourcing a number of traditional IT activities including data storage is widespread and there is a great deal of enthusiasm for it across a wide range of organizations, but this enthusiasm is not yet being widely and systematically backed up by detailed risk assessments and careful consideration of the approaches needed to identify, classify, manage, and monitor the data being transferred into the cloud.

The comments provided alongside the answers also provided useful information. In particular, several respondents referred to the different types of cloud that are beginning to exist: in certain industries such as financial services it would be unthinkable to use cloud services in which confidential data might be stored outside Switzerland or for which it would be difficult to obtain adequate audit comfort over key concerns, but the use of some kind of industry-specific Swiss cloud, perhaps setup as a joint venture, with appropriate controls and safeguards in place, might be conceivable.

Several respondents also flagged up the importance of the proper management of backup media, which of course need to be subject to the same policies and procedures for data management and security as live datasets. From the perspective of the management who will retain the responsibility for ensuring the availability and reliability of system and data backups for good practice and going concern reasons, and for the auditors who will be verifying this, it will be a challenge to identify exactly which data are backed up where, how this is managed from a security and availability perspective, and what the timelines, sequences, and interdependencies would be for restoring part or all of a missing dataset.

Within all businesses there is constant pressure to reduce costs and cloud computing could be seen as an effective method of managing and reducing costs, particularly in the short term. If there are no significant in-house IT systems, a case will always be made for reducing to a bare minimum, or even eliminating entirely, the IT function, thereby, reducing staff costs alongside the operational costs of monitoring and maintaining systems.

The decision to choose cloud computing services is not one to be taken lightly. For individuals, the use of personal services in the cloud (such as Facebook, gmail, and Dropbox) is, or rather should be, a matter of a calculated assessment of risks and benefits, for the potential negative impacts of breaches in security, for example, can be significant. For businesses and other organizations, the same concerns apply but on a larger scale. For all organizations that have a responsibility to keep their, and others', data secure and confidential, the decision can only be taken after a detailed analysis of how they will obtain, and continue to obtain, the necessary comfort that this is the case. If they cannot build into their own procedures, into enforceable contract terms, and into audit plans, the means of confirming the confidentiality, integrity, and availability of their systems and data, they should not consider externalizing it.

In addition, organizations should not forget the immediate costs and efforts involved in moving onto the cloud. Datastores need to be identified, classified, cleaned up, and archived, and serious technical and operational decisions are needed to determine what data will sit where. This will frequently be a project of a significant size requiring expertise, resources and input from a number of people across the business who understand the business, the systems, the data, and their use.

Experience of traditional outsourcing suggests that it is very easy for organizations to overestimate the cost savings generated by a move toward service providers and to underestimate the amount of internal competence and dedicated management required to make a success of such initiatives. As long as the organization relies on its data and retains responsibility for all aspects of its business from a regulatory perspective, it will need to ensure that its management of the relationship with its service providers and its access to critical operational information are both adequate and appropriate. Typically this will require retaining or recruiting skilled, experienced, and reasonably senior staff to liaise with and monitor the performance of the service provider.

The long-term consequences of opting for a cloud solution also need to be examined. Once on the cloud, systems and data are likely to stay there, and feasibly with the same provider. What begins as a simple and cost-effective solution to a small problem could develop into a long-term strategic commitment with little scope for alteration.

7 Further Research Issues

It will be necessary to monitor the development of the use of cloud services for data storage from a controls perspective in order to see how such use develops, whether it becomes widespread and whether it does become a factor in assessing the quality of internal controls. Perhaps practical and effective operating procedures will be developed and become standard very rapidly, so that the subject does not become a major

concern for controls managers, or perhaps the take up of the technologies will not be great, because of perceived controls issues or questions of cost or access.

It will also be interesting to study the impact of the uptake of such services on audit opinions and compliance reports. This will surely be a major driver in the development and the use of these services: if organizations find themselves subject to adverse comments from regulators or external auditors, that will necessarily cause a slow-down in adoption. On the other hand, if clean reports are issued and no concerns are raised, take up will only be encouraged.

8 Conclusion

In common with all IT-related projects, initiatives in respect of data collation, and accumulation and in respect of data migration and transfer require a great deal of planning, strategic awareness and effective controls in order to ensure that the key security objectives defined by the organization continue to be met. Both managing unstructured data and managing the migration onto, and ongoing monitoring of, cloud services, present countless opportunities for the loss of the confidentiality, integrity and availability of data, to cite once again just the three most famous security objectives.

The use of large datasets for competitive advantage is highly tempting for many organizations from a tactical perspective, while the use of cloud services is attractive for a number of reasons, financial, operational, and strategic. The senior management of organizations tempted by such initiatives should be aware, however, that neither type of project can be successfully completed overnight, and that they should be prepared to provide the necessary resources, guidance, oversight, and supervision to ensure that the advantages obtained through the initiatives are not outweighed by decreased security, increased costs, or reduced comfort from internal controls.

References

[1] Marr, B.: Is This the Ultimate Definition of “Big Data” ?, http://smartdatacollective.com/node/128486 (accessed September 2, 2013)

[2] Bojilov, M.: Big Data Defined. In: ISACA Now (2013), http://www.isaca.org/Knowledge-Center/Blog/Lists/Posts/Post.aspx?ID=299 (accessed September 2, 2013)

[3] Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., Hung Byers, A.: Big data: the next frontier for innovation, competition and productivity. McKinsey Global Institute, Washington DC (2011)

[4] Blumberg, R., Atre, S.: The Problem with Unstructured Data. DM Review (2003)

[5] National Institute of Standards and Technology. The NIST Definition of Cloud Computing. Special Publication 800-145, NIST, Gaithersburg (2011)

[6] Committee of Sponsoring Organizations of the Treadway Commission. Guidance on Monitoring Internal Control Systems. AICPA, New York (2009)

[7] Krutz, R., Vines, D.: Cloud Security: A Comprehensive Guide to Secure Cloud Computing. Wiley Publishing Inc., Hoboken (2010)
[8] Internal Standards Organization. ISO/IEC 27001 Information Technology - Security Techniques – Information Security Management Systems. ISO/IEC, Geneva (2005)
[9] National Institute of Standards and Technology, Information Security. Special Publication 800-100. NIST, Gaithersburg (2006)
[10] Parker, D.: Toward a New Framework for Information Security. In: Bosworth, S., Kabay, M. (eds.) Computer Security Handbook, 4th edn. John Wiley & Sons, New York (2002)
[11] Organisation for Economic Co-operation and Development (2002) OECD Guidelines for the Security of Information Systems and Networks: Towards a Culture of Security, http://www.oecd.org/sti/ieconomy/15582260.pdf (accessed January 8, 2014)
[12] Adolph, M.: Distributed Computing: Utilities, Grids and Clouds. ITU-T Technology Watch Report 9, ITU, Geneva (2009)
[13] Gantz, J., Reinsel, D.: Extracting Value from Chaos. IDC iView (2011)
[14] Information Systems Audit and Control Association, Big Data: Impacts and Benefits. ISACA, Chicago (2013)
[15] Bittman, T.: A Better Cloud Computing Analogy. Gartner Blogs (2009), http://blogs.gartner.com/thomas_bittman/2009/09/22/a-better-cloud-computing-analogy/ (accessed January 8, 2014)
[16] Information Systems Audit and Control Association, Security Considerations for Cloud Computing. ISACA, Chicago (2012)
[17] Ghernaouti-Hélie, S., Tashi, I., Simms, D.: Optimizing security efficiency through effective risk management. Paper presented at the 25th IEEE International Conference on Advanced Information Networking and Applications (AINA 2011), Biopolis, Singapore, March 22-25 (2011)
[18] American Institute of Certified Public Accountants, Quick Reference Guide to Service Organizations: Control Reports. AICPA, New York (2012)
[19] American Institute of Certified Public Accountants, Service Organizations: Reporting on Controls at a Service Organization Relevant to User Entities' Internal Control over Financial Reporting Guide. AICPA, New York (2011)

Future Human-Centric Smart Environments

María V. Moreno-Cano, José Santa, Miguel A. Zamora-Izquierdo, and Antonio F. Skarmeta

University of Murcia, Department of Information and Communications Engineering, Facultad de Informática, Campus de Espinardo, 30100 Murcia, Spain
{mvmoreno,josesanta,mzamora,skarmeta}@um.es

Abstract. Internet of Things (IoT) is already a reality, with a vast number of Internet connected objects and devices that has exceeded the number of humans on Earth. Nowadays, there is a novel IoT paradigm that is rapidly gaining ground, this is the scenario of modern human-centric smart environments, where people are not passively affected by technology, but actively shape its use and influence. However, for achieving user-centric aware IoT that brings together people and their devices into a sustainable ecosystem, first, it is necessary to deal with the integration of disparate technologies, ensuring trusted communications, managing the huge amount of data and services, and bringing users to an active involvement. In this chapter, we describe such challenges and present the interesting user-centric perspective of IoT. Furthermore, a management platform for smart environments is presented as a proposal to cover these needs, based on a layered architecture using artificial intelligent capabilities to transform raw data into semantically meaningful information used by services. Two real use cases framed in the smart buildings field exemplify the usefulness of this proposal through a real-system implementation called *City Explorer*. City Explorer is already deployed in several installations of the University of Murcia, where services such as energy efficiency, appliance management, and analysis of the impact of user involvement in the system are being provided at the moment.

Keywords: User-Centric, IoT, Smart Buildings, Energy Efficiency, Context Awareness.

1 Introduction

The smart concept is flooding our life. Currently, everybody speak about smart cities, smart companies, smart transport, etc., whose main enablers are the last advances in Information and Communication Technologies (ICT) as well as proposals for the integration of sensors, actuators, and control processes. The world is being transformed at such speed that by 2015 is expected that over 50 billion devices are interconnected into a full ecosystem known as Internet of Things (IoT) [1].

IoT represents a key enabler for smart environments, enabling the interaction between smart things and the effective integration of real-world information and

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*, Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_14

knowledge into the digital world. Smart things, instrumented with sensing and interaction capabilities or identification technologies, will provide the means to capture information about the real world in much more detail than ever before, which means it will be possible to influence real-world entities and other actors in real time.

The initial roll out of IoT devices has been fueled primarily by industrial and enterprise centric cases. For instance, a set of IoT application scenarios have been identified for their expected high impact on business and social benefits. These scenarios are showed in Fig. 1. Since some knowledge and services of different IoT scenarios can be shared and used in the other scenarios, all of them can be considered like linked, as it is reflected in Fig. 1.

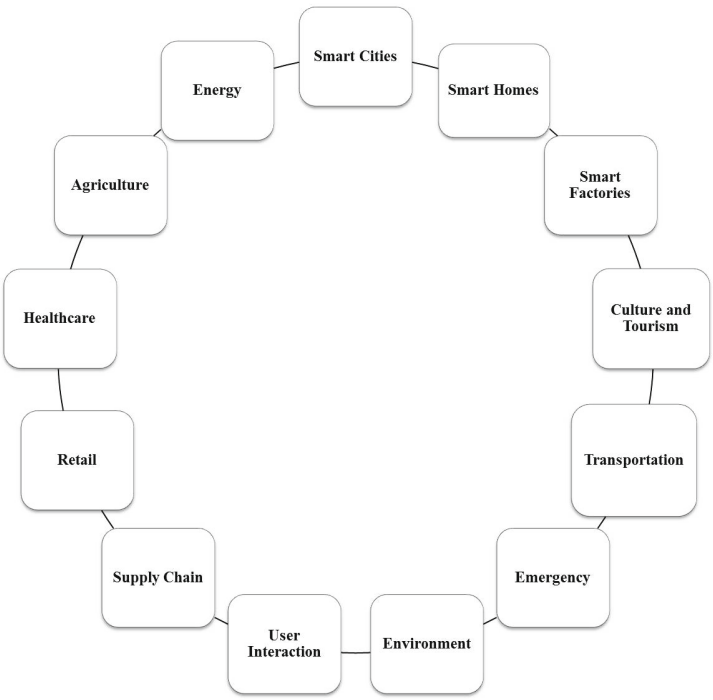


Fig. 1. IoT application scenarios with high expected business and societal impact

However, the exploitation potential of IoT for smart services that address the needs of individual users, user communities, or society at large, is limited at this stage and not obvious to many people. Unleashing the full potential of IoT means going beyond the enterprise centric systems and moving toward a user inclusive IoT, in which IoT devices and contributed information flows provided by people are encouraged. This will allow us to unlock a wealth of new user-centric IoT information, and a new generation of services of high value for society will be

built. The main strength of this idea is the high impact on many aspects of everyday-life, affecting the behavior of humans.

To achieve smart and sustainable environments it is fundamental that, first, people understand and participate actively within this ecosystem to ensure the expected goals considered during the system design. Additionally, encouraging people to contribute with their active participation (through their agreement for sharing the information sensed by their devices, interacting with the system, etc.) would be possible to build the technosocial foundations to unlock billions of new user-centric information streams. Therefore, the human perception and understanding is a key requirement for a successful uptake of ICT and IoT in all society areas.

Furthermore, as individuals produce the majority of content on Internet today, we can expect that users and their smart devices (like their smartphones) will be responsible for the generation of the majority of IoT content as well. Crowdsourcing [2] is a very good example of this trend. Through such collective effort, continuous capturing of detailed snapshots of the physical world around us would be possible, and thus providing a vast amount of data that can be leveraged for the improvement of the population quality of life.

From the point of view of individuals, one of the most obvious impacts of IoT applications will be present indoors, making smart buildings a reality. A smart building provides occupants with customized services, thanks to their intelligent capabilities, in offices, houses, industrial plants, or leisure environments. The smart buildings field is currently undergoing a rapid transformation toward a technology-driven sector with rising productivity. This paradigm will ultimately create a solid foundation for continuous innovation in the building sector, fostering an innovation ecosystem as the foundation stone for smart cities.

Bearing all these aspects in mind, in this chapter we present our proposal of user-centric smart system based on the optimal integration and use of the information provided by, among others, the users themselves. This system is applicable to different smart environments such as transportation, security, health assistance, etc. As an example of smart environment, in this work we focus on the smart buildings field, where the system makes decisions and uses behavior-based techniques to determine appropriate control actions, (such as control of appliances, lights, power energy, air conditioning, control access, security, energy-aware comfort services, etc.), and where it is promoted the user intervention and participation in real-time.

The content of this chapter is organized as follows: since the emerging horizon of IoT still presents a variety of technological and socioeconomic barriers that have to be overcome, in Section 2 we enumerate the main challenges for becoming this ideal concept into a livable and sustainable reality. Section 3 reviews the potential of IoT technologies applied to user-centric systems, where the central role of the user is reflected on all aspects of the ecosystem. Section 4 presents our IoT-based architecture designed and developed at University of Murcia, which is able to offer user-centric services in different smart environments. Later on we focus on the instantiation of this system in two real use cases within the smart

buildings area in Section 5. A set of different user-centric services are provided in these scenarios, such as user comfort, appliances management, security, or energy-efficiency. Finally, Section 6 reports conclusions extracted from our technological analysis, system design, and the real deployments introduced in this work.

2 Main Challenges for Achieving Livable Smart Cities

The IoT enables a broad range of applications in the context of smart cities for very diverse aspects of modern live, including: smart energy provisioning (e.g., monitoring and controlling of city-wide power distribution and generation), intelligent transport systems (e.g., reactive traffic management), smart home deployments (e.g., the control of household devices and entertainment systems), and assisted living (e.g., remote monitoring of aged and/or ill people), etc. For each of these applications it is important to understand the user requirements to design an optimal interaction with humans, to consider assurance needs (reliability, confidentiality, privacy, auditability, authentication, and safe operation), and to finally understand dependencies between the implemented systems.

From the above observations it can be formulated four main challenges to enable real-life smart city services by means of IoT technologies, while considering security and privacy requirements: *IoT technologies*, *trusted IoT*, *smart city management and services*, and *user involvement*. These are further described next.

2.1 IoT Technologies

End-to-end reliable and secure communication channels for enabling IoT applications and infrastructure management are two of the most relevant technological concerns to explore for establishing the different technologies to integrate in infrastructure deployments like smart cities.

Another particular aspect that requires attention refers to the **resource differences** between resource-constrained IoT devices and powerful back-end services. This issue render today's IP security solutions infeasible for a number of IoT devices. Hence, new lightweight (IP) protocols are needed for the IoT and smart cities. These protocols have to explicitly consider the resource heterogeneity in their design in order to ensure good performance and quality of service for smart city services.

Additionally, beyond the already recognized communication technologies (like 6LoWPAN - IPv6 over Low Powered Wireless Personal Area Networks- and ZigBee) and application protocols (like CoAP [15]), it is of paramount importance to think about **sensing and acting technologies** in the IoT. Sensors in smartphones have the capability of providing lots of information exploitable in many research areas.

So far approaches for opportunistic mobile sensing focus primarily on supporting single applications and provide resource optimizations only for a particular use case and underlying algorithm(s). But a more generic approach to resource

optimization for opportunistic sensing on smartphones is required, which should take holistically into consideration the sensing context and demands of multiple applications while performing optimizations across the entire processing chain. This field is at a very early research stage, especially if we consider that this source of information will be further complemented by a vast amount of sensing devices in a smart city deployment.

2.2 Trusted IoT

With IoT becoming a reality, smart devices, services, sensors and actuators are able to interact with users and among themselves to provide data about different aspects of live. However, the next questions arise: **How a user can configure and communicate securely with such devices? How he/she can control the information flow among smart devices and services? How the user's trust can be ensured with respect to the legitimacy of a smart object?**

So far, research and standardization has mainly focused on IP connectivity of smart devices provided by IPv6 and its related with emerging standards such as the IETF ROLL[1], 6LoWPAN[2], and CORE[3] for implementations such as Contiki[4], and supports the mentioned standards from IETF (ROLL, 6LoWPAN, and CoAP). This is supported by the majority of the 6LoWPAN nodes of the market.

However, security aspects at the different layers in the network stack have recently shifted into the research and standardization focus. Securely bootstrap mechanisms [15] and security mechanisms at the link layer are currently under development. For end-to-end security, variants of existing IP-based security protocols are already being designed [16,17,18]. Still, the currently proposed approaches do not give a sound answer to the questions asked above, in particular taking into account the **resource heterogeneity of Internet and IoT devices or the protocol translations** (e.g., HTTP to CoAP), that breaks end-to-end security.

Furthermore, sensor networks in the IoT are expected to support multiple applications at once, which may be manufactured by different vendors and installed by 3rd parties. Hence, secure software execution, data management and transactions between smart devices need to be defined in order to support trust and interoperability across IoT applications. In addition to these challenges, smart city applications will generate a huge volume of personal data and will need third-party data. Hence, **preserving the privacy of the user's data** in such complex systems remains an important challenge.

[1] The ROLL working group is focused on routing for different use cases and situations

[2] The 6LoWPAN working group specifies IPv6 adaptation mechanisms that afford IPv6 connectivity over low-power, lossy links.

[3] The CORE working group (constrained RESTful environments) standard is IPv6 and 6LoWPAN oriented.

[4] Contiki is an IoT-capable operating system for small devices from SICS, which is used in European projects such as Sensei [3].

2.3 Smart City Management and Services

This is a completely open area to be analyzed in deep with the aim of designing optimum strategies that face the previous challenges. Beyond the facts that networking/application protocols are going to enable the collection of huge amount of information, and these operations should be performed in a secure and privacy-aware manner, it is still necessary to use this information to enable smart services in cities.

Currently, the first smart city services are being already designed[4], but it is still unclear how services can be enabled in the best way to accomplish with the expected goals. For instance, if we want to allow for smart parking, **should we put sensors on the road, or is it better to use cameras?** Of course, we can rely on protocols to collect the data and do it securely, but **which of the approaches allow for the fastest and more economical deployment?**. In the same manner, we have to further explore how these services are to be managed. **Should they be managed by the city? Or by the users? How are the interactions between parties?**

For these reasons, it is still necessary to develop some control/monitoring schemes and decision-making systems based on intelligent information processing techniques to support the extensibility toward different kinds of smart city services, such as transport management, water distribution, smart buildings, environmental monitoring, energy efficiency, social requirements, etc. The IoT infrastructure, which is formed by many heterogeneous sensors and actuators networks embedded in the city environment (buildings, garbage bins, bicycles, public transport, lampposts, traffic lights, public parks and green areas, children playgrounds etc.), and connected wirelessly or via wired networks, will provide the basics to allow for smart city services.

Currently, there are no real solutions yet for such smart city services giving an answer to how they can be managed by relying on the IoT infrastructure. Therefore, IoT management frameworks have to be designed taking into account the specifics of various IoT devices, the contexts in which they are running, as well as stakeholders' policies and requirements.

Summarizing, efforts are required to integrate and fine tune the technologies proposed together with the standards proposed in IETF, IEEE, ETSI, or ISO. Core technology modules must be identified and advanced technology features can be dynamically loaded and configured based on the application requirements, the architectural, network, and device constraints for smart cities. Even more important it is to analyze how services can be actually used and enabled in smart cities by relying on the underlying IoT infrastructure and how these services can be efficiently managed.

2.4 User Involvement

User involvement in IoT smart systems refers to both passive and active participation of users with the system. An example of passive involvement is the case of only considering user presence, where any noninvasive IoT smart space can provide users with pervasive services according some goals, for instance alzheimer or

dementia patients who are sensed by the system and provided with customized assisted-living services). In contrast, active involvement of individuals in producing content for Internet and social networks has become prominent only with the advent of the so called Web 2.0, i.e., when supported by ubiquitous availability of Internet connectivity and personal devices connected to Internet **simple and easy to use tools** for content generation, publishing, and sharing. In the IoT case, the threshold for active participation of users in a joint IoT system is much higher due to lack of adequate tools, mechanisms, and incentives as well as fear of disclosure of private information that could potentially be abused in yet unknown ways.

The move toward a **citizen inclusive IoT** in which citizens provided IoT devices and contributed information flows are encouraged will have significant impact on the people and the societies in general. Thus, a variety of technological socioeconomic barriers will have to be overcome to enable such inclusive IoT solutions. In particular, the **human perception** of IoT is critical for a successful uptake of IoT in all areas of our society.

Perceived levels of **trust and confidence** in the technology are crucial for forming a public opinion on IoT. This is a real challenge for IoT solutions, which are expected to behave seamless and act in the background, invisible to their users.

In order to ensure a wide scale uptake of IoT in all areas of society, architectures and the protocols of an inclusive IoT ecosystem must be simple and provide **motivation for every citizen to contribute** an increasing number of IoT devices and information flows in their households, this way making them available to their immediate community and to the IoT at large. In addition to the simplicity in terms of how the system is used and the **immediate and clear benefits** provided by the system to each individual user, implementation must be done in such a way that to ensure adequate **control and transparency** in order to increase confidence and to allow better understanding of what is happening with the information and devices contributed. If transparency and user control are not treated adequately in such a community grown IoT system, there is a real danger for the systems to be perceived with suspicion and mistrust by the users, which may result in opposition and refusal of such technology, thus hindering its wide spread deployment.

Bearing in mind this last challenge to achieve user-centric smart city services, next section presents a complete description of the issue from different points of view.

3 User Centric IoT Systems

Thanks to last pervasive computing advances, the integration and development of systems based on IoT support and enhance the cooperation between humans and smart devices. For instance, some of these cooperation facilities can be found in the next areas:

- Facilitating communication between things and people, and between things, by means of a collective network intelligence context.
- People's ability to exploit the benefits of this communication with their increasing familiarity with ICT.
- The vision where, in certain respects, people and things are homogeneous agents endowed with fixed computational tools to compose smart and sustainable ecosystems.

The motivation for such cooperation between humans and technology can be seen from different perspectives. Fig.2 shows a schema of different aspects affecting user-centric smart environments.

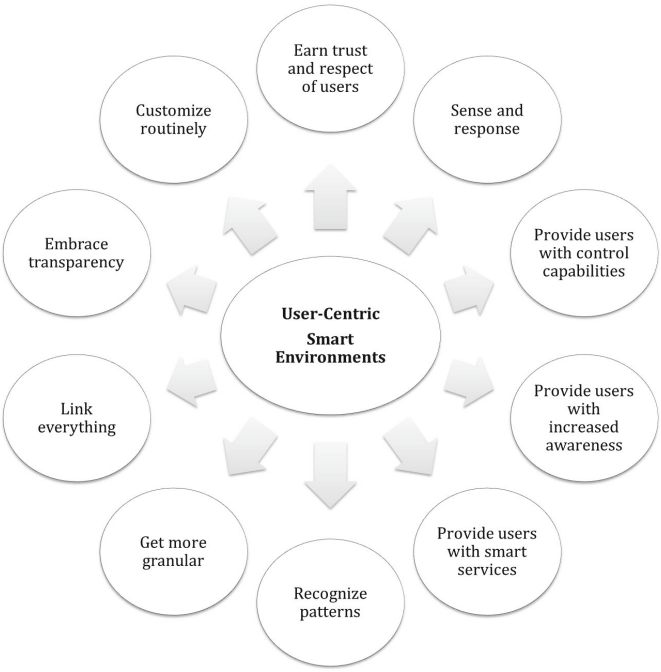


Fig. 2. Features of smart environments from a user-centric perspective

Next we focus on analyzing the human, social, and computational points of view of such user-centric IoT approach.

3.1 Human Centered Perspective

From a human-centric perspective [5], users should be both the final deciders and the system co-designers in terms of feedback conditioning future goals (i.e., the services provided by the IoT systems) and contributions to the software issuing these goals.

Nowadays one can already talk about smart people, smart computer systems and, during last years, about smart objects too. Following this approach, in a near future the "intelligence" will be so disseminated around us - integrated in the objects used daily by people - that it may be possible that people become foolish living in a reality where "smart entities" are in charge of taking all decisions for them. Computer systems have already similar capacities as humans, like memory and making decision processes. At this speed, in few years, a huge amount of our personal memory could be stored out of us, as well as our capability of making decisions. Some experts have already stated that the human role in a near future could be reduced to a big sensed system to the service of computer systems.

Computer science is one of the most revolutionary scientific creation of the mankind, but whose advances should not be fully dehumanizing, by considering the human-centric perspective of them. Allen Newell, one of the fathers of the artificial intelligence, published a book titled *Unified Theories of Cognition* [6], which was fully congratulated by the scientific community. Newell described in this book, that the goal of the intelligence is to relate two independent systems: the knowledge and the goals. According to this approach, when a problem is solved, the intelligence uses whole knowledge available to get a concrete goal, i.e., the solution of such problem. But in this theory, two essential functions of the human intelligence are excluded: create information and invent goals. Therefore, the Newell's definition of intelligence is not valid for humans. The main capability of human intelligence consists of selecting its own information, focus on its reality, and establish its own goals. Therefore, human intelligence is genuine thanks to its capability of addressing the mental activity to adjust ourselves to the reality - and even being able to extend it.

For all these reasons, computer systems providing people with human-centric services should include the active participation of humans to be properly recognized like *intelligent*, and thus to make possible sustainable and livable ecosystems to the service of the society.

Therefore, since the intelligence of computer systems is dependent on human intelligence of users interacting with them, it is necessary to propose an effective involvement process of humans during the whole lifecycle of this type of systems. In this sense, from the beginning to the end of such system operation, it is necessary to:

1. Include and take into account the data sensed and handed by user devices.
2. Provide users with all sensed data and predictions carried out according to their context.
3. Consider the information provided from the user interactions with the system. In this sense, people should be able to:
   - Accept, change, or reject the solutions provided by the system.
   - Communicate what are their preferences related with a specific service.
   - Indicate what are their satisfaction/dissatisfaction levels regarding to the provided solutions.
   - Inform about their habits and goals.
   - Specify new services that cover new needs or desires.

The general trend is that the goals considered during the system design can be questioned by users at various level of detail, and automatically updated for an optimum operation of the system. Consequently, this gives rise to a knowledge milieu where all the members, more or less experienced, of a social network may proactively share and refine their knowledge - in terms of goals generation - with other people who own the same or different devices under similar contexts. Below we describe in more details such social point of view of this emerging user-centric IoT paradigm.

3.2 Social Networks Perspective

According to the definition made by Wasserman & Faust in 1994 [7], a Social Network (SN) is generally a system with a set of social actors and a collection of social relations that specify how these actors are relationally tied together. In more recent years, due to the advent of new web technologies and platforms (blogs, wikis, Facebook, and Twitter, for instance) - which allow the users to take an active part in content creation - the definition of SNs has been updated including *content production and sharing aspects*[8]. Thanks to this information generation, smart systems can use this to provide more efficient services according to the user requirements and the context conditions.

The actions and relationships of users in a proactive SN provide a rich source of observable user behavior (for example, related to what, and with whom, content is shared) that modeling approaches can leverage. In order to exploit this additional information, a user model for SNs, in addition to the attributes of classic and context-aware user models, must also include attributes modeling the user's social behavior in terms of user relationships and content production and sharing (or limitations to it) [9].

More in detail, user relationships in SNs are more commonly expressed as friendship (or like, or trust) statements between users. Taken together, the trust sets of the users in the SN can be seen like a user graph (i.e., a social graph), in which the nodes are the users and the edges are the trust connections [10].

Since in an online community people are motivated to participate by different things in different ways, it is to be expected that personalizing the incentives and the way the reward for participation are presented to the individual would increase the effect of the incentives on their motivation. Modeling the changing needs of communities and adapting the incentive mechanisms accordingly can help attract the kind of contributions when they are most needed. Therefore, user modeling inside a community or a SN can be seen as an area that provides valuable insights and techniques in the design of adaptive incentive mechanisms [11]. Such mechanisms are able to motivate a user to participate or perform an action inside the network. Therefore, the purpose of incentive mechanisms is to change the state of the user (goals, motivations, etc.) to adapt the individual user to the benefit of the overall system or community (which is the opposite of the purpose of user adaptive environments, where the objective is to adapt the system to the needs of the individual user).

An incentive mechanism to switch from the user level to the community level can be viewed as an adaptation mechanism toward the behavior of a community of users. This incentive adaptation mechanism monitors the actions of the community represented in a community model, or in a collection of individual user models, and makes adaptations to the interface, information layout, functionality of the community, etc. to respond to the changes in the user model according to some predefined goals (e.g., maximizing the participation and content sharing).

Therefore, it can be noticed that it is necessary the integration of suitable computational techniques that let us face with all these aspects of modeling and adaptation. The next part introduces the main concepts related to such computational intelligence versant of user-centric IoT systems.

3.3 Computational Intelligence Perspective

From an operational perspective, the amount of data in the world is growing very fast. Computers are present in almost all life facets and low price of storage units, making very easy and cheap to store information that even were previously discarded. According to some estimates [12], the quantity of data stored in databases worldwide doubles every 20 months, approximately. As the world grows in complexity, the amount of data generated can be overwhelming, and Computational Intelligence (CI) could be a useful tool to discover patterns that underlie such data. Data, smartly analyzed, are a valuable resource, being able to lead to more and better understanding of the environment, or in the business world.

Machine learning can be defined as the ability of a computer to modify or adapt their actions in order to improve its performance in the future [12]. The "Knowledge Discovery in Databases" process has the overall goal of extracting information from a data set and transforming it into an understandable structure for further use, through statistical, machine learning, database management methods, etc.

The theoretical steps of such transformation are: (i) Selection; (ii) Preprocessing; (iii) Transformation; (iv) Data Mining; and (v) Interpretation/Evaluation. Pattern recognition and knowledge extraction from high-spatiotemporal multidimensional data involves two processes very suited in Human Machine Interface (HMI) applications.

Since IoT and Social Networks areas produce large amount of data, in several papers adaptive mining frameworks have already been proposed [13], which are used jointly with tools adapting to changes in data in various contexts such as pattern discovery from sensor data, analysis of data collected in a central data warehouse, and pattern mining from sensor data.

Taking into account all mentioned until now, i.e., both the challenges that IoT systems still present and the different perspectives that motive the novel user-centric approach of this paradigm, in the next section we present a proposal of architecture that covers fundamental aspects cited in Section 2 following a user perspective like this described in Section 3.

3.4 Architectural Perspective

Smart objects are viewed as one of the major Internet growth areas of the future. Each smart object has a set of Application Program Interfaces (APIs), by which the users or external software agents can access to its data or service via the network. To handle API calls, a smart object is usually equipped with at least a processor, a storage and a network interface. Thus, a network of objects is smart enough to define a programming paradigm for an easy management of data and services.

There are two different approaches: a distributed one and a service centralized architecture:

- Distributed architecture. Following this approach, a smart environment, where many smart objects are deployed, can be seen as a community of agents endowed with own goals and behavioral rules. This reflects into a Self-adaptive software community to realize the self-management of the federation of services provided by a network of smart objects. For instance, in [14] a management infrastructure is described. This provides capabilities to adapt services according to usage context.
- Centralized architecture. In this type of architecture, a gateway collects the data sensed by sensors and is in charge of dispatching the commands to the actuators. Such commands are centrally elaborated in a middleware. The communication among the devices is performed with an underlying network protocol which can be in general based on web services.

4 A User-Centric Architecture for Smart Environments

The IoT architecture presented in this work is depicted in Fig. 3. The design and implementation of it was developed at University of Murcia, specifically, in the Department of Information and Communications Engineering. This architecture promotes a high-level interoperability at the communication, information, and service layers, and it is generic enough to be applicable in different smart environments such as intelligent transport systems, security, health assistance, or smart buildings, among others.

Among smart environments mentioned, buildings affect the quality of life and work of citizens.They should prevent users from having performed routine and tedious tasks to achieve comfort, security, and effective energy management. Sensors and actuators distributed in buildings can make user's life more comfortable; for example: i) rooms heating can be adapted to user preferences and to the weather; ii) room lighting can change according to the time of the day; iii) domestic incidents can be avoided with appropriate monitoring and alarm systems; and iv) energy can be saved by automatically switching off electrical equipment when not needed, or regulating their operating power according to user needs.

On the other hand, to date, real-time information about the energy consumed in a building has been largely invisible to millions of users, who had to settle with

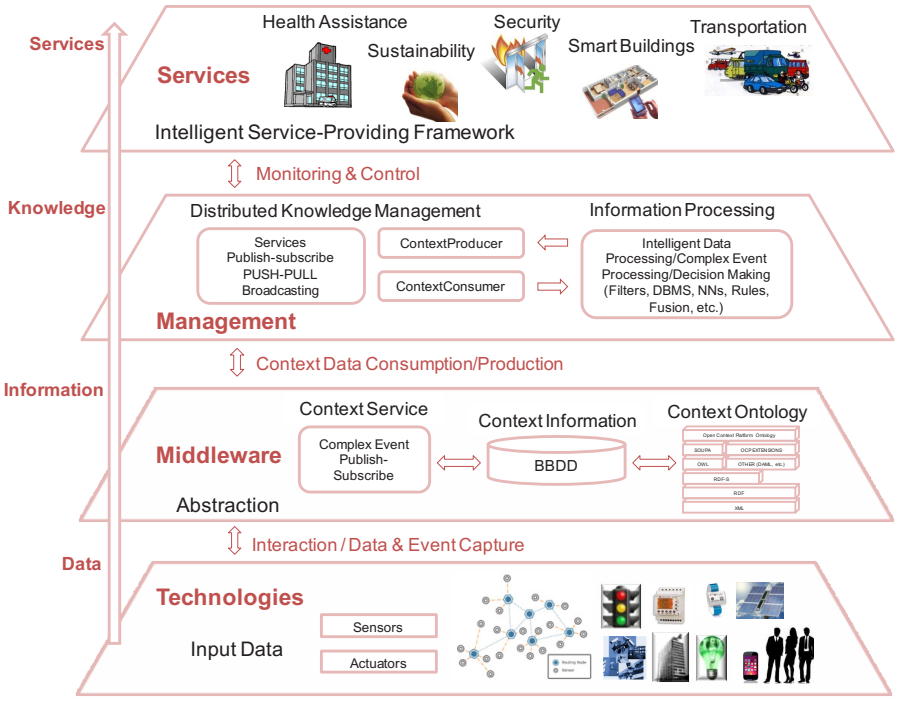


Fig. 3. IoT Architecture for smart systems

traditional electricity bills. Now there is a huge opportunity to improve the offer of cost-effective, user-friendly, healthy, and safe products for smart buildings, which provide users with increased awareness (mainly concerning the energy they consume), and permit them to be an input of the underlying processes of the system.

Bearing these aspects in mind, the rest of the paper focuses on the human-centric perspective of emergent IoT systems in the context of smart buildings, where users are both the final deciders of actions and system co-designers, since their feedback conditions the behavior of the software. Next more details about the architecture showed in Fig.3 are given, at the same time we describe it implementation over a real platform: City Explorer.

4.1 Technologies Layer

Looking at the lower part of Fig. 3, input data are acquired from a plethora of sensor and network technologies such as Web, local, and remote databases, wireless sensor networks or user tracking, all of them forming an IoT framework. Sensors and actuators can be self-configured and controlled remotely through the Internet, enabling a variety of monitoring and control applications.

For our proposed architecture, an automation system used to gather information from sensors and actuators has been developed as a particular study case framed in smart buildings. It follows an IoT approach and widely covers the functionalities found in the technologies layer. The base system is known as *City Explorer* and its main components were presented in detail in [19]. City Explorer is the basis for an automation system able to monitor environmental parameters, gather data for tracking occupants, detect anomalies (such as fire and flooding in buildings), and take actions to deal with key efficiency requirements, such as saving power or water consumption.

The main components of this platform are the network of Home Automation Modules (HAM) and the SCADA (supervisory control and data acquisition), as can be seen in Fig. 4. Each HAM module comprises an embedded system connected to all the appliances, sensors, and actuators of a specific area of the building. Thus, these devices centralize the intelligence of each space, controlling the configuration of the installed devices there. Additionally, the SCADA offers management and monitoring facilities through a connection with HAMs. In this way, all the environmental and location data measured by the deployed sensors are first available in the HAMs, and then reported to the SCADA, which maintains a global view of the whole building infrastructure.

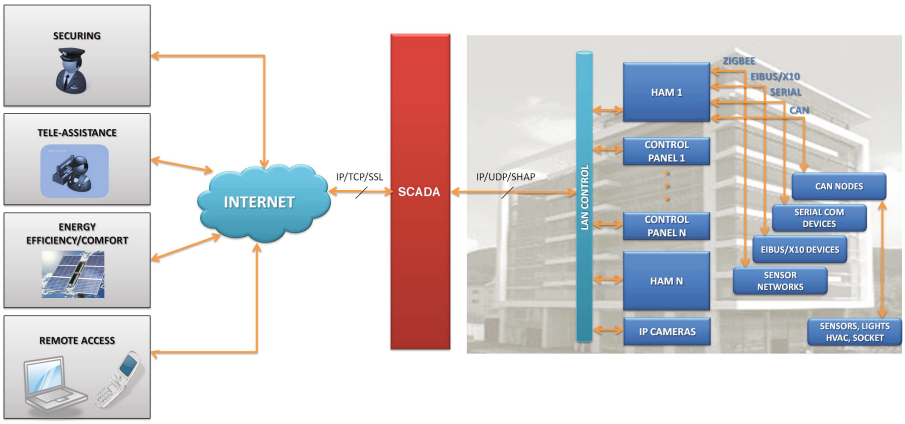


Fig. 4. Overall building automation system provided by City Explorer

The HAMs of City Explorer support several communication controllers in order to connect with many devices. In addition, by complementing the direct digital and analog I/O through common wiring, a CAN (Controller Area Network) bus can be used to extend the operation range or provide a more distributed wiring solution. X-10 connections over the power line are also available for low-cost domotic installations, whereas the EIB controller offers a powerful solution for connecting with more complex appliances. Serial-485 devices can be connected and the powerful Modbus protocol can be used. Finally, ZigBee (or

6LowPAN) and Bluetooth can be used to avoid wiring in already built buildings for instance, and connect new devices through a wireless sensor network.

In addition, a LAN installation is used in buildings to connect all IP-based elements with the HAMs, whereas a changeable communication technology can be used to connect the in-building network with Internet. Optical fiber, common ADSL, ISDN, or cable-modem connections could be enough to offer remote monitoring/management and a basic security system.

Given the heterogeneity of data sources and the necessity of seamless integration of devices and networks, a middleware mediator is proposed to deal with this issue. Therefore, the transformation of the collected data from the different data sources into a common language representation is performed in the middleware layer.

4.2 Middleware Layer

This layer is responsible for the management of the information flows provided by different sources. The different information sources could be: sensors, data bases, web pages, etc., whose data and behavior can be controlled. These data sources can be enquired through several coordination mechanisms, for instance through publisher/subscriber methods.

For our IoT-based architecture, we propose to use the OCP platform (Open Context Platform), developed by the University of Murcia and further described in [20]. OCP is a middleware to develop context-aware applications based on the paradigm of producer-consumer. Hence, in our case of study in the smart buildings field, the producer is the City Explorer platform which is in charge of collecting information from the sensors deployed in the building and from the automated devices, and adds information to the OCP. Meanwhile, one or more consumers interested in some specific context parameters are notified about the changes performed in this information. The context information is collected in an ontology defined according to the model that represents the knowledge of the application domain (an example could be the goal of achieving energy efficient buildings), while a service to manage this information using OCP is used by consumers and producers of the context.

4.3 Management Layer

This layer is responsible for processing the information extracted from the middleware and for making decisions according to the final application context. Then, a set of information processing techniques are applied to fuse, extract, contextualize, and represent information for the transformation of massive data into useful knowledge that is also distributed.

In this layer two phases can be distinguished. The first one acts as context consumer of the middleware, where intelligent data processing techniques are implemented over the data provided by the middleware layer. On the other hand, the second phase acts as context producer, where complex event and decision making processes are applied to support the service layer with useful knowledge.

During this stage new context information can be generated, which is provided to the middleware for its registration in the ontology containing the data context (acting then as context producer). Therefore, different algorithms must be applied for the intelligent processing of data, events and decisions, depending on the final desired operation of the system (i.e., the addressed services).

Considering the field of the smart buildings, and as Fig. 4 shows, in this layer it should be implemented the data processing techniques for covering services such as security, tele-assistance, energy efficiency, comfort, and remote control, among others. In this context, intelligent decisions are made through behavior-based techniques to determine appropriate control actions, such as appliance and lights control, power energy management, air conditioning adjustment, etc.

4.4 Services Layer

Finally, the specific features for service provisioning, which are abstracted from the final service implementations, can be found in the upper layer in Fig. 3. This way, our approach is to offer a framework with transparent access to the underlying functionalities to facilitate the development of different types of final applications.

As we showed in Fig. 4, a schema of our automation platform offering some ubiquitous services in the smart buildings field is represented. It is divided into the indoor part of the platform and all the connections with external elements for remote access, technical teleassistance, security, and energy efficiency/comfort provision.

Additionally, in order to provide a local human-machine interface (HMI), which can be considered trustworthy by users and lets them interact easily with the system, several control panels have been distributed through the building to manage automated spaces. This comprises an embedded solution with an HMI adapted to the controlled devices and able to provide any monitored data in a transparent way. Bearing users in mind, the HMI of the control panels of City Explorer manages to reduce the risk of injury, fatigue, error, and discomfort, as well as improves productivity and the quality of the user interactions.

Taking into account the common services provided in the smart buildings context, in the next section we present two examples of real use cases where our proposal of smart system (City Explorer) is deployed and working to provide the main common building services, such as remote access, technical tele-assistance, security, energy efficiency, and comfort provision in two different contexts: in a technology transfer center and in the main campus buildings of the University of Murcia.

5 Real Deployments at University of Murcia

Buildings represent one of the fields where energy sustainability must be satisfied to ensure energy sustainability of modern cities, mainly due to the increasing amount of time that people spend indoors. For instance - and as a reference

- in developed countries, the electric consumption of buildings covers between 20% and 40% of the total energy consumption. At the same time, it is common that in smart buildings, quality of life of occupants is ensured through three basic factors: thermal comfort, indoor air quality, and visual comfort [22]. Therefore, a proper system able to cope with a optimum tradeoff between energy consumption and user comfort is needed. City Explorer is able to cover this need, as it is showed in the two study cases presented in the rest of this section.

5.1 Study Case 1: Technology Transfer Center

Scenario. A reference building where our smart system is already deployed and working is the Technology Transfer Center of the University of Murcia[5], where City Explorer is installed. Fig. 5 depicts one of the floors of this reference building, where a set of laboratories is present on the lower part of the map. This screenshot has been obtained from our SCADA-web, which also offers the possibility of consulting any monitored data from the heterogeneous sensor network deployed in the building.

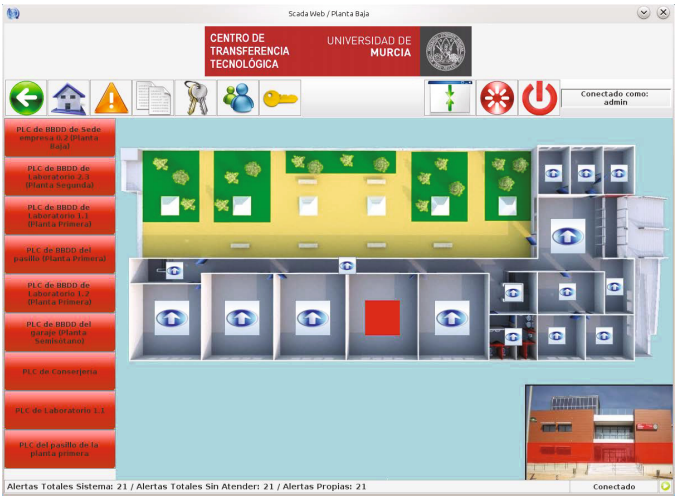


Fig. 5. SCADA-web view of the ground floor of the reference smart building

Every room of the building is automated through an HAM unit. Therefore, we hereby consider a management granularity at device level in every automated area of our reference building, i.e., in rooms, corridors, and shared areas like entrance, stairs, etc.

[5] Technology Transfer Center of the University of Murcia: `www.um.es/otri/?opc=cttfuentealamo`

Energy Efficiency and Comfort. In this building context, our proposal of intelligent management system has the main capability of adapting the behavior of automated devices deployed in the building in order to meet energy consumption restrictions while maintaining comfort conditions at the occupants' desired levels. More specifically, the goals of our intelligent management system here are the follow:

- High comfort level: learn the comfort zone from users' preferences, guarantee a high-comfort level (thermal, air quality, and illumination) and a good dynamic performance.
- Energy savings: combine the comfort conditions control with an energy saving strategy.
- Air quality control: provide CO_2-based demand-controlled ventilation systems.

For Satisfying the above requirements demands control implies controlling the following actuators:

- Shading systems to control incoming solar radiation and natural light as well as to reduce glare.
- Windows opening for natural ventilation or mechanical ventilation systems to regulate natural airflow and indoor air changes, thus affecting thermal comfort and indoor air quality.
- Electric lighting systems.
- Heating/cooling (HVAC) systems.

As a starting point, we focus on the management of lights and HVAC appliances, since they represent the highest energy consumption impact at building level. It is important to consider that HVAC implies 76% of energy consumption in buildings in European Countries [23]. From this use of case, we intend to propose an "energy consumption profile" of the buildings which can be used as basis for self-configured IoT system.

User-Centric Approach. User interactions have a direct effect on the system behavior, because the occupants can take the control of the comfort appliances at any time. Thus, the combined control of the system requires optimal performance of every management subsystem (lighting, HVAC, etc.), under the assumption that each one operates normally in order to avoid conflicts with users' preferences.

As regards the monitoring and control capabilities at room level, data involved in energy and comfort services provided in each room comprise the input data of the intelligent system integrated in the HAM installed in the target scenario. Additionally, separate automation functions for managing lighting and HVAC appliances distributed in each room are also provided by the HAM unit. Therefore, at room level, it is possible to minimize energy consumption according to the actions suggested by the management system allocated there, which

also takes into account user interactions with the system using the control panel associated to such room or using the SCADA-web access.

Looking at Fig. 5, we have taken the second laboratory starting from the left as the reference testbed for carrying out the experiments. In this test laboratory we have allocated different room spaces where sensors are distributed. Fig. 6 provides an overview of such deployments as well as the contexts of an office, a dining room, a living room, a corridor, and a bedroom.

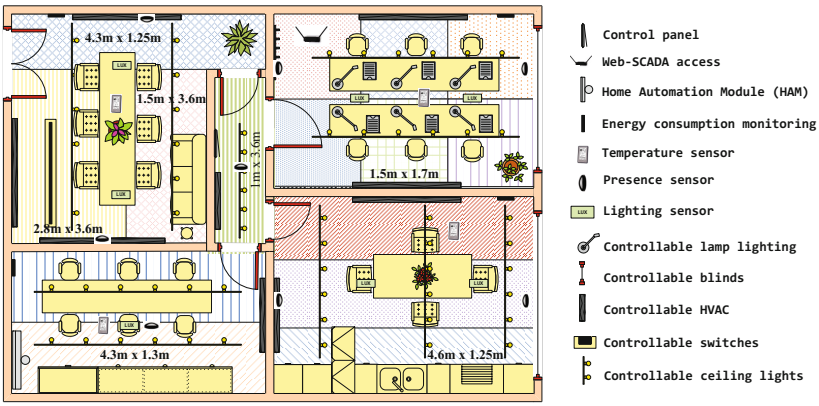


Fig. 6. Different contexts in a test lab of our reference smart building

We provide our intelligent system with deep user-centric principles, applying a cyclic learning/adjustment process that satisfies the user requirements of data gathering, deployment, and behavior assessment. Therefore, our system is gradually provided with innovative persuasive strategies and improvements based on the feedback received from users, who are active actors in the operation of the system rather than passive receivers. In this sense, and as starting point of our building management system, maximum, and minimum comfort parameters are established as control points for ensuring minimal comfort conditions of occupants while energy efficiency aspects are considered. For this purpose, we take into account the comfort models proposed in [25], which predict the comfort response of building occupants considering features such as location type, user activity and date.

The CEN standard EN 15251 [21] specifies the design criteria (thermal and visual comfort, and indoor air quality) to be used as input for calculating the energy performance of buildings, as well as the long-term evaluation of the indoor comfort. Our intelligent building system takes into account this standard of quality measurements, the user comfort requirements, and the generated energy in the building (i.e., by using the solar panels deployed in the roof's building) in order to define the best environmental comfort while energy efficiency is maintained. Therefore, the developed mechanism to control energy efficiency and

comfort conditions in our test building is fully integrated into the management layer of the IoT architecture presented in the previous section.

The parameters that have been identified to affect comfort and energy performance are: indoor temperature and humidity, natural lighting, user activity level, and power consumed by electrical devices. Environmental parameters (temperature, humidity, and natural lighting) present a direct influence on energy and comfort conditions but, in addition to them, thermal conditions also depend on the user activity level and the number of users in the same space. Depending on the indoor space (such as a corridor or a dining room), the comfort conditions are different and, therefore, the energy needed too. Moreover, the heat dissipated by electrical devices also affects thermal conditions.

One of the most relevant inputs to our energy efficiency management is the human activity level, which is provided by our indoor localization mechanism integrated in City Explorer and detailed in [24]. This is based on an RFID/IR data fusion mechanism able to provide information about the occupants' identity, indoor locations, and activity level. Hence, location data allow the system to control the HVAC and lighting equipment accordingly.

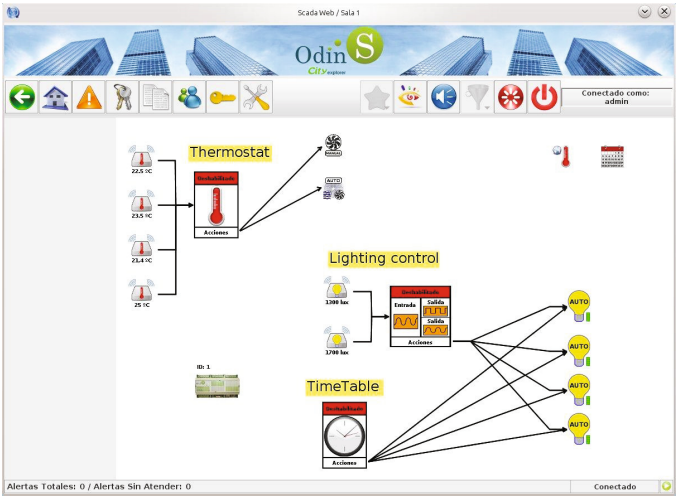


Fig. 7. Example of rules defined through the City Explorer's editor

After the identification and localization of occupants inside the building, different comfort profiles for each user are generated with default settings according to their preferences. In this way, considering accurate user positioning information (including user identification) as well as user comfort preferences for the management process of the appliances involved, energy wastage derived from overestimated or inappropriate settings is avoided. Nevertheless, occupants are free to change the default values for their own preferences when they do not feel comfortable. For this, users can communicate their preferences to the system

through the control panel of the HAM associated to their location, or through the SCADA-web access of City Explorer. Then, our management system is able to update the corresponding user profiles as long as these values are within the comfort intervals defined according to the minimum levels of comfort in the building context. On the other hand, when occupants are distributed in such way that the same appliance is providing comfort service to more than one occupant, our intelligent system is able to provide them with comfort conditions that satisfy the greatest number of them (always considering the minimum levels of comfort).

Regarding user interaction with the system to communicate their comfort preferences and energy control strategies, besides City Explorer lets users explore monitored data by navigating through the different automated areas or rooms of the building, its intuitive graphic editor also allows users to easily design any monitoring/control task and/or actions over the actuators (appliances) deployed in the building. The setting of the overall system can also be carried out by users using City Explorer, and without any need to program by code any controller. In this way, it is possible to setup the whole system by simply adding maps and pictures over which users can place the different elements of the system (sensors, HAM units, etc.), and design monitoring and control actions through arrows in a similar way to that in which a flowchart is built. Therefore, our system gives users integral control of any aspect involved in the management of the building. An example of the graphic editor of City Explorer where some rules have been defined by users is shown in Fig.7.

The system can detect inappropriate settings indicated by users according to both their comfort requirements and associated energy consumption parameters. Therefore, with the aim of offering users information about any unsuitable design or setting of the system, as well as to help them to easily understand the link between their everyday actions and environmental impact, City Explorer is able to notify them about such matters (i.e., acting as a learning tool). On the other hand, when the system detects disconnections and/or failures in the system during operation, it sends alerts by email/messages to notify users to check these issues. All these features, included in our management system, let contribute to user behavior changes and increase their awareness over time, and detect unnecessary stand-by consumption of the controllable subsystems of the building.

Evaluation. Despite the relatively short time of evaluation (two months), an early analysis shows that the system has already had a positive impact on user behavior, which can be translated into energy saving terms. Fig. 8 shows the energy savings achieved during the second month of operation of our energy management system in contrast to the first experimental month. It can be seen how we have experienced saves of up to 12% of the energy involved, with medium saves of 9%. Furthermore, the results reflect how the savings are more evident with time, specifically from the 17th day of the system operation. This is due to our system is able to learn and adjust itself to any feedback indicated by users regarding their comfort profile, and to recognize patterns of user behavior.

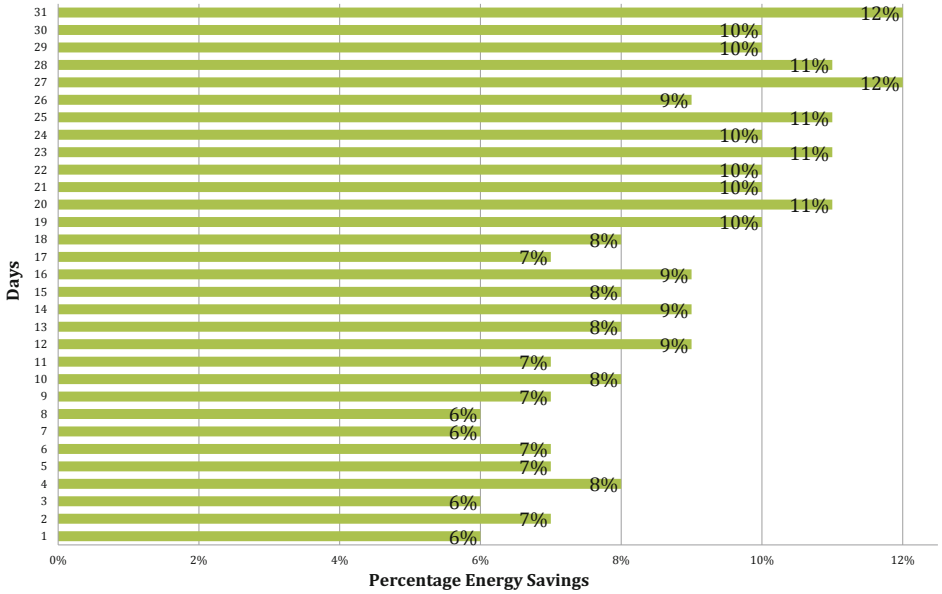


Fig. 8. Percentage of energy savings considering a user-centric approach for the comfort appliances management in smart buildings

5.2 Study Case 2: OpenData Project in the University Campus

The OpenData Project looks for the implantation of our holistic smart system in some campus buildings of the University of Murcia, intended for the control and management of different infrastructures with high energy consumption impact as well as for offering a set of indoor pervasive services. Currently, the project is in its early stages of implantation, but it is expected that it is completed by mid 2014. Below there is a description of the target services that our system will provide in these buildings.

– **Technical Tele-Assistance**
One of the main functionalities of the SCADA of City Explorer, which is only available for authorized personnel, is the technical tele-assistance. This allows technicians to remotely diagnose the operation of the various devices and subsystems instaled in buildings. Apart from a common monitoring access, where the technician can check the operation parameters of devices, it is possible to receive alarms that indicate anomalies in the system. Unattended ones are stacked and notified until the devices in question are checked.

When accessing the status of subsystems or individual devices, the tele-assistance support of City Explorer is able to provide the manufacturer-recommended operation parameters and compare them with the current ones. This makes easier the work of technicians, who can avoid traveling for in-situ assistance when the problem can be solved remotely. This way the maintenance costs during the system exploitation are reduced.

– **Securing the Building**
Due to the relevance of safety services in current building automation systems, the City Explorer architecture includes an integrated security system. Local sensors connected to the HAM, such as presence, noise, and door opening detectors, are used as inputs for the security system. As it can be seen in Fig. 4, a security control entity is in charge of receiving security events from the building. City Explorer interfaces with the software installed at the security company using standardized alarm messages. More details about the security provisioning in buildings can be found in [19].

Sometimes, automated buildings are included in an administrative domain (e.g., housing developments) and monitoring/securing tasks can be applied by local staff. The SCADA system can be used in these cases to receive security events from buildings and provide notifications to local security staff, apart from the security notifications sent to external security companies if desired.

– **Energy Efficiency & Comfort Provision**
As for the first case of study presented previously, in this an essential capability of City Explorer. This way the system gathers information from all devices about the energy production and consumption to propose new settings for HVAC and lighting appliances. This process is carried out by the energy efficiency/comfort module depicted in Fig. 4, with the aim of minimizing the energy consumption while maintaining the desirable comfort conditions.

The energy efficiency module monitors the building spaces, as it has been explained above, is in charge of proposing the needed actuation commands over key devices with the aim of saving energy (adapts lighting and HVAC systems, switch on/off appliances, etc).

6 Conclusion

The proliferation of ICT solutions, in general, and IoT approaches, in particular, represents new opportunities for the development of intelligent services to achieve more efficient and sustainable environments. In this sense, persuasive energy monitoring technologies have the potential to encourage sustainable energy lifestyles within buildings, as the proposal described in this chapter has demonstrated. Nevertheless, to effect positive ecological behavior changes, user-driven approaches are needed, whereby design requirements are accompanied by an analysis of intended user behavior and motivations. Nevertheless, large data samples are needed to understand user preferences and habits for the case of indoor services.

In this way, building management systems able to satisfy energy efficiency requirements, but user comfort conditions are also considered necessary. However, to date, studies have tended to bring users into the loop after the design is completed, rather than including them in the system design process.

In this work, after providing the main challenges that still have to be faced to achieve smart and sustainable environments from the IoT perspective, and

make a description of different views motivating the user-centric services in such paradigm, we propose a smart environments management architecture which is powered by IoT capabilities and novel context-and location-aware capabilities. Then, we show a real instantiation of this approach in the City Explorer platform, which involves a completely modular and flexible solution composed of a building infrastructure and a set of optional remote software elements providing added-value services for the indoor domain. It deals with the issues of data collection, intelligent processing and actuation, to modify the operation of relevant indoor appliances in terms of energy consumption (such as lighting and HVACs). An essential part of our intelligent management system is the user involvement, through their active interactions with the platform (using a proper HMI) and their passive participation, thanks to the data collected about identity, location, and activity. This way, customized and contextual services can be provided.

Nowadays, several buildings are being controlled at the University of Murcia using the same City Explorer/SCADA platform, as it has been showed in the two use cases described, and some private companies are considering its installation in the short term. Furthermore, a remarkable current line of work is focused on the expansion of the proposed smart system to networks of buildings and other smart city infrastructures as street lighting, where saving energy is also a key aspect to consider.

Acknowledgments. This work has been sponsored by European Commission through the FP7-SMARTIE-609062 and the FP7-SOCIOTAL-609112 EU Projects, as well as the Spanish Seneca Foundation by means of the Excellence Researching Group Program (04552/GERM/06) and the FPI program (grant 15493/FPI/10).

References

1. Atzori, L., Iera, A., Morabito, G.: The internet of things: A survey. Computer Networks 54(15), 2787–2805 (2010)
2. Ganti, R.K., Fan, Y., Hui, L.: Mobile crowdsensing: Current state and future challenges. IEEE Communications Magazine 49(11), 32–39 (2011)
3. SENSEI EU PROJECT, `http://www.sensei-project.eu`
4. Bélissent, J.: Getting clever about smart cities: new opportunities require new business models (2010)
5. Ducatel, K., et al.: Scenarios for ambient intelligence 2010, ISTAG report, European Commission. Institute for Prospective Technological Studies, Seville, `ftp://ftp.cordis.lu/pub/ist/docs/istagscenarios2010.pdf` (November 2001)
6. Newell, A.: Unified theories of cognition, vol. 187. Harvard University Press (1994)
7. Wasserman, S.: Social network analysis: Methods and applications, vol. 8. Cambridge University Press (1994)
8. ISTAG. Report on revising europe ict strategy. Technical report, European Commission (2009)
9. Spiliotopoulos, T., Oakley, I.: Applications of Social Network Analysis for User Modeling

10. Shi, Y., Larson, M., Hanjalic, A.: Towards understanding the challenges facing effective trust-aware recommendation. Recommender Systems and the Social Web, 40 (2010)
11. Vassileva, J.: Motivating participation in social computing applications: a user modeling perspective. User Modeling and User-Adapted Interaction 22(1-2), 177–201 (2012)
12. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann (2005)
13. Bin, S., Yuan, L., Xiaoyi, W.: Research on data mining models for the internet of things. In: 2010 International Conference on Image Analysis and Signal Processing (IASP). IEEE (2010)
14. Reilly, D., Taleb-Bendiab, A.: An jini-based infrastructure for networked appliance management and adaptation. In: Proceedings of the 2002 IEEE 5th International Workshop on Networked Appliances, Liverpool. IEEE (2002)
15. Sarikaya, B., Ohba, Y., Moskowitz, R., Cao, Z., Cragie, R.: Security Bootstrapping Solution for Resource-Constrained Devices. IETF Internet-Draft (2012)
16. Tschofenig, H., Gilger, J.: A Minimal (Datagram) Transport Layer Security Implementation. IETF Internet-Draft (2012)
17. Kivinen, T.: Minimal IKEv2, IETF Internet-Draft (2012)
18. Moskowitz, R.: HIP Diet EXchange (DEX), IETF Internet-Draft (2012)
19. Zamora-Izquierdo, M.A., Santa, J., Gomez-Skarmeta, A.F.: An Integral and Networked Home Automation Solution for Indoor Ambient Intelligence. IEEE Pervasive Computing 9, 66–77 (2010)
20. Nieto, I., Botía, J.A., Gómez-Skarmeta, A.F.: Information and hybrid architecture model of the OCP contextual information management system. Journal of Universal Computer Science 12(3), 357–366 (2006)
21. Centre Europeen de Normalisation: Indoor Environmental Input Parameters for Design and Assesment of Energy Performance of Buildings - Addressing Indoor Air Quality, Thermal Environment, Lighting and Acoustics. EN 15251 (2006)
22. Handbook, A. S. H. R. A. E. Fundamentals. American Society of Heating, Refrigerating and Air Conditioning Engineers. Atlanta (2001)
23. Perez-Lombard, L., Ortiz, J., Pout, C.: A review on buildings energy consumption information. Energy and Buildings 40(3), 394–398 (2008)
24. Moreno-Cano, M.V., Zamora-Izquierdo, M.A., Santa, J., Skarmeta, A.F.: An Indoor Localization System Based on Artificial Neural Networks and Particle Filters Applied to Intelligent Buildings. Neurocomputing 122, 116–125 (2013)
25. Berglund, L.: Mathematical models for predicting the thermal comfort response of building occupants. ASHRAE Transactions 84(1), 1848–1858 (1978)

Automatic Configuration of Mobile Applications Using Context-Aware Cloud-Based Services

Tor-Morten Grønli[1], Gheorghita Ghinea[2], Muhammad Younas[3], and Jarle Hansen[4]

[1] Norwegian School of IT, 0185 Oslo, Norway
tmg@nith.no
[2] Brunel University, London, UK
george.ghinea@brunel.ac.uk
[3] Oxford Brooks University, Oxford, UK
m.younas@brookes.ac.uk
[4] Systek AS, Oslo, Norway
jarle@jarlehansen.net

Abstract. The area of information technology continues to experience considerable progress and innovation in recent years. Computers have evolved from large and very expensive devices, to mainstream products we take for granted in our everyday lives. Increasingly, cloud-based services have come to the fore. Additionally, many people own multiple computing devices, from normal desktop computers to small mobile devices. We find mobile computing devices of particularly interest and this will be the focus of our study. In this chapter, we investigate a context-aware and cloud-based adaptation of mobile devices and user's experience. Our research displays new and novel contribution to the area of context-awareness in the cloud setup. We propose and demonstrated principles in implemented applications, whereby context-aware information is harvested from several dimensions to build a rich foundation for context-aware computation. Furthermore, we have exploited and combined this with the area of cloud computing technology to create a new user experience and a new way to invoke control over user's mobile phone. Through a developed application suite with the following evaluation, we have shown the feasibility of such an approach. Moreover, we believe our research, incorporating remote, and automatically configuration of Android phone advances the research area of context-aware information.

1 Introduction

We believe that mobile devices, especially in the last few years, have evolved considerably. Not only have they increased performance, they but also provide more features than before, such as the new and improved touch screens. However, in addition to the opportunities with mobile devices, they also present new challenges that are not present in standard desktop computing. Energy consumption, varying network coverage, and the relatively small screen sizes are examples of this. Moreover, in 2011 for

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_15

the first time smartphones exceeded PCs in terms of devices sold[1], which further highlights the importance of mobile devices and in this case smartphones specifically.

Innovations in hardware capabilities open up new opportunities and challenges when developing systems that run on or integrate with mobile devices. When combined with the wireless capabilities of high-speed Internet through EDGE, 3G, and 4G as well as Bluetooth and WLAN, many new research possibilities appear. Furthermore, with the updated network infrastructure and more affordable payment options from the ISPs (Internet Service Providers), the *always-connected* devices are becoming mainstream.

While smartphones are becoming increasingly powerful, the software they run has also gone through some major evolutionary steps. Particularly the Android and iOS marketplaces have been a very important factor in the platforms' success. In 2011 both Android Market[2] and the App Store[3] reached over 500,000 available applications. The maturity of the platforms and the popularity of apps are giving businesses a new channel to promote products, offer new features, and generally expand their methods of reaching out to potential customers.

Moreover, the ability to purchase and download native applications directly to the smartphones has proven to be a popular service for both consumers and developers. Developers are able to publish their applications quickly and users can navigate through a library consisting of many thousands of applications, providing everything from games and educational software to enterprise solutions. Additionally, the rating systems also provide end users with the ability to directly give feedback on the quality and price of the offered application. These features have certainly made people use their phone for more tasks than before. With the increase in usage and capabilities of the smartphones, new and interesting research opportunities have emerged. These include context-aware solutions and applications that have been around for some time now and have, successfully enriched mobile applications. In this work, we seek to build on these achievements and utilize context as a source of information for user information and interface tailoring [14]. The issue with much of the earlier approaches is that they have only looked at either one source of context-aware information or treated the context separately in the case of multiple sources. We propose a different approach, which combines context-aware information from several dimensions in order to build a rich foundation to base our algorithms on (what algorithms...this is not clear). We exploit cloud computing technology to create a new user experience and a new way to invoke control over user's mobile phone. Our solution is a remote configuration of an Android phone, which uses the context-aware information foundation to constantly adapt to the environment and change in accordance with the user's implicit requirements. Cloud-based service providers and developers are increasingly looking toward the mobile domain, having their expectations focused on

[1] http://www.smartplanet.com/blog/business-brains/milestone-more-smartphones-than-pcs-sold-in-2011/21828

[2] http://www.research2guidance.com/android-market-reaches-half-a-million-successful-submissions/

[3] http://www.apple.com/iphone/built-in-apps/app-store.html

the access and consumption of services from mobile devices [13]. Hence, integrating an application, running on a mobile device, with cloud computing services is becoming an increasingly important factor. Potentially, by utilizing such connectivity to offload computation to the cloud, we could greatly amplify mobile application performance at minimal cost [3]. Our work focuses on *data access transparency* (where clients transparently will push/pull data to/from the cloud), and *adaptive behavior of cloud applications*. We adapted the behavior of the Google App Engine server application based on context information sent from the users' devices thus integrating context and cloud on a secure mobile platform [1].

The main contribution of our work in this chapter describes a cross-source integration of cloud-based, context-aware information. This solution incorporates remote, web-based configuration of smartphones and advances the research area of context-aware information and web applications. By expanding and innovating our existing work we propose a new, novel solution to multidimensional harvesting of contextual information and allow for automatic web application execution and tailoring.

2 Concepts and Background

Mobile devices are integrated with an increasing number of tasks in our day-to-day activities. Not only is society very reliant on the devices themselves, but also the infrastructure. One relevant example in this context is an application developed by British Airways for handling airplane tickets. This application replaces, for many destinations, the check-in process and paper boarding passes completely[4]. A screenshot of the application is presented in Figure 1.

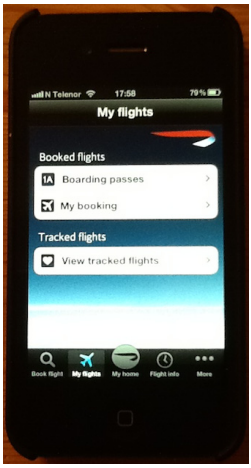


Fig. 1. British Airways iPhone app

[4] http://www.britishairways.com/travel/iphone-app/public/en_gb

Because mobile devices are usually carried around everywhere and have the capability to communicate with external resources, they provide an ideal match for these kinds of applications. It replaces other items and tasks, like boarding passes and check-in procedures, with a simple and well-integrated application that is more practical. Moreover, one does not need to print out a boarding pass or stand in queue at the check-in counter.

These features of the mobile devices are a result of many different components cooperating. Most applications communicate with various resources; these can be local to the phone, like sensors, or backend services that provide the wanted information. There are several research areas that have made significant contributions toward these technological advances we are able to use today. These areas are presented in the next section of this chapter. We will concentrate on four concepts that are particularly important when it comes to mobile devices and the integration of network communication, namely 1) *Distributed Computing,* 2) *Mobile Computing,* 3) *Pervasive Computing,* and 4) *Internet of Things.*

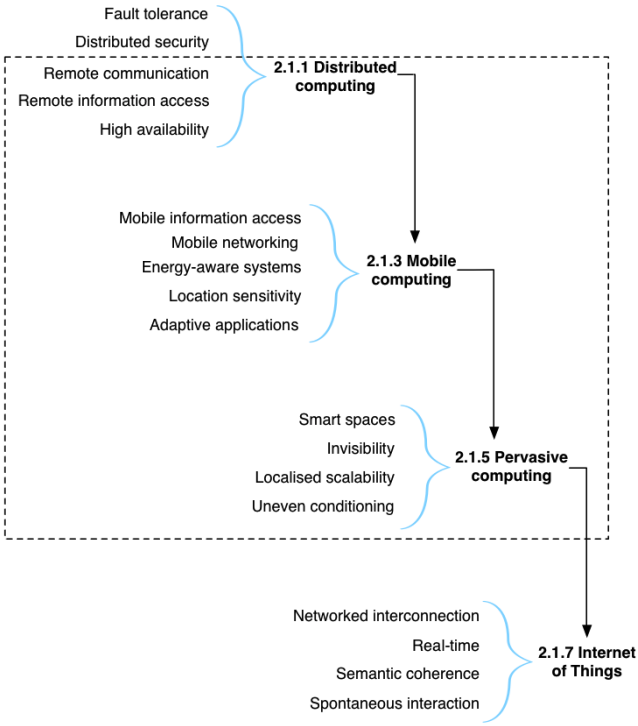


Fig. 2. Distributed Computing, Mobile Computing, Pervasive Computing, and Internet of Things overview. Adapted from Satyanarayanan [17] and Zhang et al.

As presented in Figure 2, these majors steps in mobility and computing are related both in regards to technology and research challenges. When moving toward the right of the figure, there is a tendency to either add new problems or make existing ones

more challenging [17]. On the figure, we have marked (with a stapled box) the most important issues that are investigated in this chapter.

Although new problems appear with the paradigms on the right, it is important to build on previous research findings. Distributed computing has a considerable knowledge base that helps both mobile and pervasive computing moving forward. Internet of Things is also building on knowledge learned from the previous paradigms. We will not go into detail on the Internet of Things, as this is outside the scope of our work, but we include a general description to complete the overall picture of the four research areas.

2.1 Distributed Computing

Up to the late 1970s / early 1980s computing systems were usually centralized mainframe installations and access to these was done through a text terminal [19]. When the Personal Computer (PC) started to enter the mainstream market it provided local workstations. However, these devices usually provided limited communication options. Since the 1990s, when the Internet became a commercial success, the computers have been connected in a large network. With the communication options and availability of the Internet, it was possible to share resources, like servers and printers, and enable a high degree of information sharing between users. With the Internet becoming popular, these local computers started to connect to external resources, and the initial distributed systems appeared.

A distributed system is (Tanenbaum & Van Steen [21]):

"… a collection of independent computers that appears to its users as a single coherent system".

This definition consists of two parts, which are the hardware and software. The hardware must cooperate to complete specific tasks, while the software should try to unify these hardware resources into one coherent system. Distributed computing thus takes advantage of networked resources, trying to share information, or even have different resources join forces to be able to achieve complex tasks that might take too long for just one standalone computer. Compared to implementing a single-machine system, distributed systems have their own challenges and issues.

One important concept within distributed computing is the remote network communication between devices. Different components of the system need to communicate, this is at the core of the distributed computing area. There are several attempts at making the development and overall design of such systems easier and more reliable from RPC to peer-to-peer communication and remote method invocation.

2.2 Mobile Computing

Mobile computing started with the appearance of the laptop computers and WLANs (Wireless LANs) in the early 1990s [17]. It emerged from the integration of cellular technology with the Web [20]. Mobile computing, as the name suggests, is the process of computation on a mobile device.

Network communication is a very important part of mobile computing. Usually, communication between the device and external resources is involved, where a device can request a specific set of services from back-end servers or other devices. One of the main goals of mobile computing is to offer mobility and computing power. This can be offered by providing decentralized and distributed resources on diversified mobile devices, systems, and networks that are interconnected via mobile communication standards and protocols [22]. This aspect is very important and we will address this in more detail in several of our experiments.

Creating a distributed system for mobile devices is usually even more challenging than the standard client-server architecture, where the client is situated on a desktop computer. It is difficult to create systems where there is no fixed infrastructure available and devices can enter or leave the system at any time. Also, devices can be disconnected due to issues such as an empty battery or poor network coverage. Satyanarayanan [23] identified four constraints that characterise mobile computing:

1) Mobile elements are resource-poor relative to static elements. Desktop computers and servers will usually have more hardware resources than laptop computers, smartphones, and especially lower-end mobile phones. With less hardware resources, certain considerations have to be taken when developing the systems, both on the server and client side. There are constraints specific to the applications that are created to run on mobile devices, which would not be a consideration for normal desktop clients. Energy-consumption and varying network coverage are two examples.

2) Mobility is inherently hazardous. Security in mobile computing is considerably more difficult than in environments that have a fixed infrastructure. In mobile computing the devices are *on the move* and used in many different locations and settings. This increases the possibility of theft or to simply misplace the devices. Another challenge is privacy. Smartphones often record various data about the user and the device, for instance the location. It is crucial that this kind of sensitive information is not accessible by anyone other than the authorized users.

3) Mobile connectivity is highly variable in performance and reliability. Moving from an environment with good network coverage and bandwidth to environments that offer only low speed connections or no network connection at all is an important factor the systems must handle. There are also differences in the hardware capabilities of the devices. Some devices might only offer slow wireless connections, like EDGE, while others have the capability to connect to faster networks, with, for example, 4G. If the system does not recognize and adapt to these differences, it can impact the user experience. For example, if a system sends high-quality video to a device with a very limited wireless connection, the result is long loading times and a poor user experience.

4) Mobile elements rely on a finite energy source. Battery technology has improved over time, but it still remains one of the main challenges for mobile computing. Although hardware components are created to be vey energy efficient, the smartphones are often equipped with large touchscreens. These large screens use a considerable amount of power. The concern for power consumption must span many levels of hardware and software to be fully effective. Applications should thus be able to adapt the content and settings on the device according to the battery level.

One common example of a feature that is implemented to minimize the battery usage is the light sensor, which registers the amount of light in the room and adjusts the screen brightness accordingly.

Based on these characteristics of mobile computing, Satyanarayanan [17] identified five main research challenges. These are *Mobile Information Access, Mobile Networking, Energy-Aware Systems, Location Sensitivity,* and *Adaptive Applications*, all of which form the bulk of research interest in the area today.

2.3 Pervasive Computing

The initial move towards pervasive computing started in the 1970s, when the PC brought computers closer to people [20]. However, it was not until the early 1990s the idea of truly ubiquitous/pervasive computing started to take shape. In 1991, Mark Weiser wrote about what he envisioned the *computer for the twenty-first century* would be like. The work was based on research done at Xerox, where staff had working prototypes of what they called *ubiquitous computing.* The terms *ubiquitous* and *pervasive* computing are used interchangeably throughout this chapter. The idea was that computers blend into the environment, to the point where people would no longer notice their presence. Pervasive computing thus consists of a new class of devices that make information access and processing available for everyone from everywhere at any time.

Mark Weiser [24] stated that: "Prototype tabs, pads and boards are just the beginning of ubiquitous computing. The real power of the concept comes not from any one of these devices – it emerges from the interaction of all of them." When looking at the current state of mobile computing it is quite impressive how this vision is getting increasingly closer to reality. Mobile phones, tablets, and e-readers are now very much a part of our everyday lives. Pervasive computing provides a rich diversity of applications and can connect to worldwide networks, thereby, providing access to a wealth of information and services. Pervasive computing builds on mobile computing, but adds characteristics such as transparency, application-aware adaptation, and an environment sensing ability [25].

In many situations, mobile devices require the ability to communicate with each other. This leads to a close connection of both distributed and mobile computing with pervasive computing. Pervasive systems require support for *interoperability*, *scalability*, *smartness*, and *invisibility* to ensure that users have seamless access to computing [20]. All of these features and advantages create new challenges that did not exist before [25]. In some cases, research done in distributed and mobile computing can be applied directly to the pervasive computing area. For others, the demands of pervasive computing are sufficiently different that new solutions have to be sought [17].

One important challenge in this research area is privacy. Privacy in pervasive computing environments can be a very difficult problem to solve [26]. According to Satyanarayanan [17], privacy in pervasive systems is greatly complicated when moving from distributed and mobile computing. Tracking location and other possibly sensitive data resources can cause considerable challenges. Not only can this cause a problem technically, but also from a user acceptance perspective.

Another challenging aspect is in the hardware domain. Satyanarayanan [17] and West [26] mention the issue that mobile devices are becoming increasingly smaller and placing severe restrictions on battery capacity. One example of research done in this field is a system developed by Parkkila and Porras [28]. They use a method called *cyber foraging*, where the idea is to use nearby computers in the same local network to handle the heavy tasks.

3 Internet of Things and Cloud

Internet of Things (IoT) is a networked interconnection of everyday objects and is rapidly gaining popularity, thanks in part to the increased adoption of smartphones and sensing devices [29]. Zhang et al. [18] categorises the Internet of Things with four main challenges:

1. **Networked interconnection** - all physical objects must be mapped into the Internet.
2. **Real-time** - requires real-time searching techniques.
3. **Semantic coherence** - recognize objects accurately and support a light-weight representation to accommodate semantics across smart spaces.
4. **Spontaneous interaction** - handle each interaction in an efficient manner such that the entire system is scalable and real-time.

It is evident that Internet of Things builds on pervasive computing, where words like *smart spaces, scalability,* and *interconnection* are all described in the previous sections. In other words, IoT further enhances pervasive computing through the communication among physical objects (or things) which have computing capabilities as well as physical attributes [16]. However, like the other research directions, Internet of Things adds its own challenges and opportunities.

One of these issues is scalability. As reported by BBC[5], the number of devices connected to the Internet is expected to reach 15 billion by the year 2015. This increase in network connected devices causes problems for the infrastructure, with issues like IPv4 addresses quickly running out. Even though the new IPv6, that was approved as early as 1998, is currently being pushed out to companies, it is a slow process. IPv6 is a network layer protocol, which was designed to increase the address space for nodes within the Internet [30].

Another major issue is the resource scarcity of things in IoT, as unlike classical/standard computing systems, things (e.g.,) may not have high processing and computing capacities. However, there is a growing trend of exploiting cloud computing capabilities in order to cater for resource scarcity in IoT. The idea of cloud is built around an economy of scale and the provision of more resources, better scalability and flexibility of service provision [1] . Cloud computing tends toward computing as a service [9] that deals with the utilization of reusable fine-grained components across a vendor's network, and cloud-based services are usually billed using a pay-per-use model [15]. Large IT companies like Microsoft, Google, and IBM, all have initiatives relating to cloud computing which have spawned a number of emerging research

[5] http://www.bbc.co.uk/news/technology-13613536

themes, among which we mention: *cloud system design* (Mell and Grance, 2011), *benchmarking of the cloud* [10], and *provider response time comparisons*. Mei et al. [11] have pointed out four main research areas in cloud computing that they find particularly interesting is the *Pluggable computing entities, data access transparency, adaptive behavior of cloud applications and automatic discovery of application quality.*

The Internet of Things is an up and coming area, which will undoubtedly attract more research interest in the near future. The popularity is increasing, as shown by Google trend graph searches.

4 Design and Implementation

This research looks into the utilization of web resources for tailoring of the user experience. Earlier approaches have typically looked at one source of context-aware information or, in case of more than one source, the information is utilized separately. Based on earlier work of ours [7],[8], we propose a different approach, where we combine context-aware information from several dimensions to build a rich foundation to base our algorithms on. This will allow for cloud computing to be used to create new user experiences and new ways to invoke control over a smartphone. With this a basis, it is possible to have an always adapting user interface.

As a part of our solution we chose the cloud computing platform in order to have a feature rich, scalable, and service-oriented server framework. Traditional REST framework services were considered, but found to be insufficient in terms of scalability and extensibility, i.e., to add and remove context-aware sources in an ad hoc manner. The cloud-based approach also has the advantage of being run as a platform as a service instance in the separate hosting instance of Google App Engine.

For our user experiment we implemented an application suite, a fully functional demonstration of the system. One of the main technical goals of our system is to make the interaction between the cloud and the mobile device as seamless as possible for the user.

The system was designed with three major components: an Android client, a cloud server application, and the remote Google services. Figure 1 gives an overview of the implementation of the system. The blue (or shaded) boxes in the diagram represent the parts of the system we created). The white boxes, like Google calendar and contacts, are external systems the system communicates with. The server application was deployed remotely in the cloud on the Google App Engine, while data was also stored remotely in Google cloud services.

After the Android client was instaled on the mobile device, the device will register itself to the Google. The users would start by logging in to the webpage. This webpage is part of the server application hosted on the Google App Engine. The login process uses the Google username/password. By leveraging the possibilities with Open Authorization (OAuth) the system provides the user with facility of sharing their private calendar appointments and contacts stored in their Google cloud account without having to locally store their credentials. OAuth allowed us to use tokens as means of authentication and enabled the system to act as a third-party granted access by the user.

After a successful authentication the user is presented with a webpage showing all configuration options. Because the configuration for each user is stored in the cloud, the system avoided tying it directly to a mobile device. One of the major benefits of

this feature is that the user did not need to manually update each device; users have a "master configuration" stored externally that can be directly pushed to their phone or tablet. It is also easier to add more advanced configuration options when the user can take advantage of the bigger screen, mouse, and keyboard on a desktop/laptop PC for entering configuration values than those found on mobile devices. On the webpage, by selecting the applications user wants to store on the mobile device and pressing the "save configuration"-button, a push message is sent to the client application.

4.1 Cloud to Device Messaging

The system exploits the push feature of Android 2.2 in order to send messages from cloud to devices, i.e., the C2DM (Cloud to Device Messaging). The C2DM feature requires the Android clients to query a registration server to get an ID that represents the device. This ID is then sent to our server application and stored in the Google App Engine data store. When a message needs to be sent, the "save configuration"-button is pushed. We composed the message according to the C2DM format and sent it with the registration ID as the recipient. These messages are then received by the Google C2DM servers and finally transferred to the correct mobile device.

The C2DM process is visualized in Figure 2. This technology has a few very appealing benefits: messages can be received by the device even if the application is not running; saves battery life by avoiding a custom polling mechanism; and takes advantage of the Google authentication process to provide security.

Our experience with C2DM was mixed. It is a great feature when you get it to work, but the API is not very developer friendly. This will most likely change in the future since the product is currently in an experimental state, but it requires the developer to work with details like device registration and registration ID synchronization. Although C2DM does not provide any guarantees when it comes to the delivery or order of messages, we found the performance to be quite good in most of the cases. It is worth mentioning that we did see some very high spikes in response time for a few requests, but in the majority of cases the clients received the responses within about half a second. Performance measurements (we recorded while doing the user experiments) reported an average response value of 663 milliseconds. It is also important to note that issues like network latency will affect the performance results.

The calendar and contacts integration was also an important part of the Android application. We decided to allow the Android client to directly send requests to the Google APIs instead of going the route through the server. The main reason for this is that we did not think the additional cost of the extra network call was justified in this case. The interaction is so simple and there is very little business logic involved in this part so we gave the clients the responsibility for handling it directly. The implementation worked by simply querying the calendar and contact API and then using XML parsers to extract the content.

4.2 Meta-Tagging

To make it possible for users to tag their appointments and contacts with context information we added special meta-tags. By adding a type tag, for example, *$[type = work]* or *$[type= leisure]*, we were able to know if the user had a business meeting or

a leisure activity. We then filtered the contacts based on this information. If the tag *$[type=work]* was added, this lets the application know that the user is in a work setting and it will automatically adapt the contacts based on this input. In a work context only work-related contacts would be shown. To add and edit these tags we used the web-interface of Google contacts and calendar.

5 Prototype and Evaluation

The developed prototype was evaluated in two phases. In the first, a pilot test was performed with a total of 12 users. These users were of mixed age, gender, and computer expertise. The results from this phase were fed back into the development loop, as well as helped remove some unclear questions in the questionnaire. In the second phase, the main evaluation, another 40 people participated. Out of the 40 participants in the main evaluation, two did not complete the questionnaire afterwards and were, therefore, removed making the total number of participants 38 in the main evaluation. All 50 participants were aged between 20 and 55 years old, had previous knowledge of mobile phones and mobile communication, but had not previously used the type of application employed in our experiment. None of the pilot test users participated in the main evaluation.

The questionnaire that was employed in the second phase had three different parts, dealing with the *web application*, *context-awareness*, and *cloud computing,* respectively, in which participants indicated their opinions on a 4-point Likert scale anchored with *strongly disagree(SD)/ disagree (D)/ agree(A)/ strongly agree(SA)*. Evaluation results are summarized in Table 1.

Table 1. User evaluation questionnaire and results

Statement	Domain	Mean	Std. Dev.
	Web application		
S1	I was able to register my device application configuration in the web application	3.61	0.59
S2	I was not able to store and push my configuration to my mobile device from the web page	1.47	0.80
S3	We would like to configure my phone from a cloud service on a daily basis, (webpage user config and Google services like mail/calendar/contacts)	3.18	0.69

Table 1. (*continued*)

Context-awareness			
S4	The close integration with Google services is an inconvenience. We are not able to use the system without changing my existing or creating a new e-mail account at Google	1.76	0.88
S5	Calendar appointments displayed matched my current user context	3.58	0.55
S6	The contacts displayed did not match my current user context	1.29	0.52
S7	I would like to see integration with other online services such as online editing tools (for example Google Docs) and user messaging applications (like Twitter and Google Plus)	3.29	0.73
Cloud computing			
S8	I do not mind Cloud server downtime	2.08	0.78
S9	I do not like sharing my personal information (like my name and e-mail address) to a service that stores the information in the cloud	2.16	0.79
S10	Storing data in the Google Cloud and combining this with personal information on the device is a useful feature.	3.26	0.60
S11	I find the cloud-to device application useful	3.53	0.51

5.1 Web Application

The statements dealing with the web application at Google App Engine (Figure 3) show that the web application performed as expected, by letting participants register their devices as well as pushing performed configurations to the devices. Also answers from S3 are quite interesting, highlighting a positive attitude toward cloud-based services.

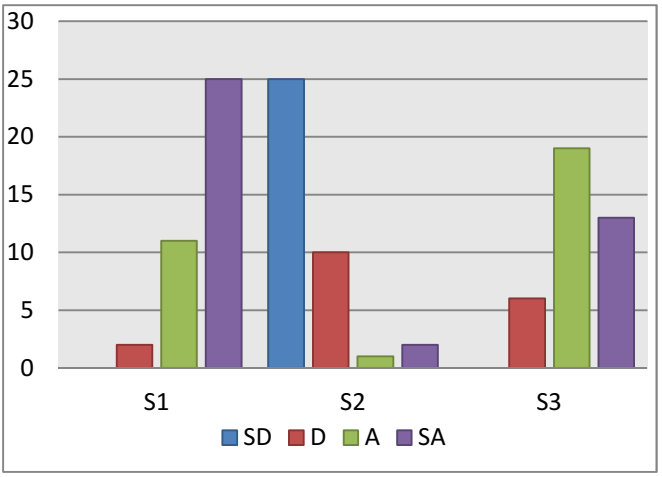


Fig. 3. Web application statements

5.2 Context-Awareness

In terms of context-aware information, participants were asked to take a stand in respect of four statements, with results shown below (Figure 4). For the first statement (S4), although a clear majority supported this assertion, opinions were somewhat spread and this answer was not statistically significant. For the next two statements a very positive bias was registered, indicating correctly computed context-awareness and correct presentation to the users. Again for S7, users are eager to see more cloud-based services and integration.

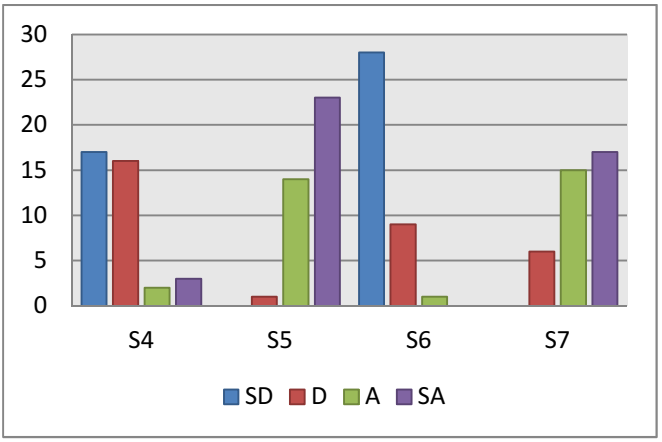


Fig. 4. Context-awareness statements

5.3 Cloud Computing

When inspecting results from the cloud-computing section (Figure 5), results are mixed and differences in opinions do occur. For S8 and S9 the results are not statistically significant, but they indicate a mixed attitude toward cloud vulnerability and cloud data storage. The two statements with statistically significant results, S10 and S11, participants find storage of data in the cloud and using this as part of the data foundation for the application a useful feature and are positive toward it. Their answers also suggest a fondness for push-based application configuration.

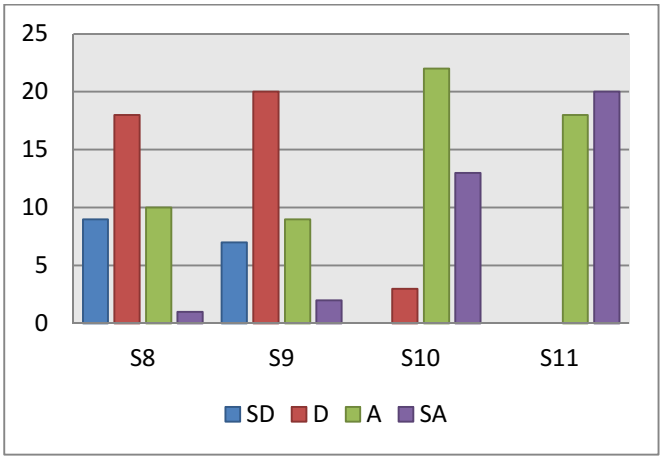


Fig. 5. Cloud computing statements

6 Analysis and Discussion

From the literature we point at the ability for modern applications to adapt to their environment as a central feature [4]. Edwards [5] argued that such tailoring of data and sharing of contextual information would improve user interaction and eliminate manual tasks. Results from the user evaluation support this. The users find it both attractive as well as have positive attitudes toward automation of tasks such as push updates of information by tailoring the interface. This work has further elaborated on context-aware integration and has shown how it is possible to arrange interplay between device context-aware information, such as sensors, and cloud-based context-aware information, such as calendar data, contacts, and applications, building upon suggestions for further research on adaptive cloud behavior as identified by Christensen [2] and Mei et al. [10][11].

To register the tags the standard Google Calendar and Contacts web interface were used. Such a tight integration with the Google services and exposure of private information was not regarded as a negative issue. As shown in the results, most of the users surveyed disagreed that this was an inconvenience. This perception makes room for further integration with Google services in future research, where, among them,

the Google+ platform will be particularly interesting as this may bring opportunities for integrating the social aspect and possibly merge context-awareness with social networks.

Sensors are an important source of information input in any real-world context and several previous research contributions have looked into this topic. The work presented in this chapter follows in the footsteps of research such as that of Parviainen et al. [31], and extends sensor integration to a new level. By taking advantage of the rich hardware available on modern smartphones, the developed application is able to have tighter and more comprehensively integrated sensors in the solution. Although sensor integration as a source for context-awareness is well received, it still needs to be father enhanced. In particular it would be useful to find out appropriate extent and thresholds that should be used for sensor activation and deactivation. We have shown that it is feasible to implement sensors and extend their context-aware influence by having them cooperate with cloud-based services in a cross-source web application scenario. Further research includes investigating sensor thresholds and the management of different sources by different people in a web scenario.

In this chapter we investigated into context-aware and cloud-based adaptation of mobile devices and user's experience. Our research has added a new and novel contribution to the area of context-awareness in the cloud setup. We have proposed and demonstrated principles in implemented applications, whereby context-aware information is harvested from several dimensions to build a rich foundation on which to base our algorithms for context-aware computation. Furthermore, we have exploited and combined this with the area of cloud computing technology to create a new user experience and a new way to invoke control over user's mobile phone. Through a developed application suite, we have shown the feasibility of such an approach, reinforced by a generally positive user evaluation. Moreover, we believe our solution, incorporating remote and automatically configuration of Android phone advances the research area of context-aware information.

7 Future Research

It can be very expensive to write separate applications in the native language of the platforms. Therefore, when developing multi-platform systems there is also the possibility to use HTML 5 technology to create a shared application. While we acknowledge this fact, we also see that in today's environment there are scenarios where HTML 5 does not provide the wanted performance requirements or lacks specific features. In the topic of heterogeneity, we see the potential for future research providing more detail on this idea of a common platform, such as a closer investigation of HTML 5 compared to native applications. Looking at large cloud-computing providers, they have access to an enormous amount of data, and the lack of transparent knowledge on how this information is used has provoked concerns. This would certainly be an interesting topic for future research on the topic of cloud computing acceptance in regard to personal information sharing.

From several scenarios described in this chapter, we see that future research should continue to innovate and expand the notion of context-awareness enabling further automatic adaptation and behavior altering in accordance with implicit user's needs.

References

[1] Binnig, C., Kossmann, D., Kraska, T., Loesing, S.: How is the weather tomorrow?: towards a benchmark for the cloud. In: Proceedings of the Second International Workshop on Testing Database Systems. ACM, Providence (2009)

[2] Christensen, J.H.: Using RESTful web-services and cloud computing to create next generation mobile applications. In: Proceedings of the 24th ACM SIGPLAN Conference Companion on Object Oriented Programming Systems Languages and Applications. ACM, Orlando (2009)

[3] Cidon, A., et al.: MARS: adaptive remote execution for multi-threaded mobile devices. In: Proceedings of the 3rd ACM SOSP Workshop on Networking, Systems, and Applications on Mobile Handhelds, MobiHeld 2011, pp. 1:1–1:6. ACM, New York (2011)

[4] Abowd, G.D., Dey, A.K.: Towards a better understanding of context and context-awareness. In: Gellersen, H.-W. (ed.) HUC 1999. LNCS, vol. 1707, pp. 304–307. Springer, Heidelberg (1999)

[5] Edwards, W.K.: Putting computing in context: An infrastructure to support extensible context-enhanced collaborative applications. ACM Transactions on Computer-Human Interaction (TOCHI) 12, 446–474 (2005)

[6] Elsenpeter, R.C., Velte, T., Velte, A.: Cloud Computing, A Practical Approach, 1st edn. McGraw-Hill Osborne Media (2009)

[7] Grønli, T.-M., Hansen, J., Ghinea, G., Younas, M.: Context-Aware and Cloud Based Adaptation of the User Experience. In: Proceedings of the 2013 Advances in Networking and Applications (AINA), pp. 885–891. IEEE Computer Society (2013)

[8] Grønli, T.-M., Ghinea, G., Younas, M.: Context-aware and Automatic Configuration of Mobile Devices in Cloud-enabled Ubiquitous Computing. Journal of Personal and Ubiquitous Computing (2013)

[9] Khajeh-Hosseini, A., et al.: The Cloud Adoption Toolkit: supporting cloud adoption decisions in the enterprise. Software: Practice and Experience, Software: Practice and Experience 42(4, 4), 447–465 (2012)

[10] Mei, L., Chan, W.K., Tse, T.H.: A Tale of Clouds: Paradigm Comparisons and Some Thoughts on Research Issues. In: Proceedings of the 2008 IEEE Asia-Pacific Services Computing Conference, pp. 464–469. IEEE Computer Society (2008)

[11] Mei, L., Zhang, Z., Chan, W.K.: More Tales of Clouds: Software Engineering Research Issues from the Cloud Application Perspective. In: Proceedings of the 2009 33rd Annual IEEE International Computer Software and Applications Conference (2009)

[12] Mell, P., Grance, T.: The NIST Definition of Cloud Computing (2011)

[13] Paniagua, C., Srirama, S.N., Flores, H.: Bakabs: managing load of cloud-based web applications from mobiles. In: Proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services, iiWAS 2011, pp. 485–490. ACM, New York (2011)

[14] Strobbe, M., Van Laere, O., Ongenae, F., Dauwe, S., Dhoedt, B., De Turck, F., Demeester, P., Luyten, K.: Integrating Location and Context Information for Novel Personalised Applications. IEEE Pervasive Computing, 1 (2011)

[15] Vaquero, L.M., et al.: A break in the clouds: towards a cloud definition. SIGCOMM Comput. Commun. Rev. 39(1), 50–55 (2008)
[16] Vermesan, O., et al.: Internet of Things Strategic Research Roadmap. European Research Cluster on the Internet of Things, Cluster Strategic Research Agenda (2009)
[17] Satyanarayanan, M.: Pervasive computing: vision and challenges. IEEE Personal Communications 8(4), 10–17 (2001)
[18] Zhang, D., Yang, L.T., Huang, H.: Searching in Internet of Things: Vision and Challenges. In: 2011 IEEE 9th International Symposium on Parallel and Distributed Processing with Applications (ISPA), pp. 201–206 (2011)
[19] Boger, M.: Java in Distributed Systems: Concurrency, Distribution and Persistence, 1st edn. Wiley (2001)
[20] Saha, D., Mukherjee, A.: Pervasive Computing: A Paradigm for the 21st Century. Computer 36(3), 25–31 (2003)
[21] Tanenbaum, M., Van Steen, A.: Distributed Systems: Principles and Paradigms. Prentice Hall (2002)
[22] Kamal, R.: Mobile Computing. Oxford University Press, USA (2008)
[23] Satyanarayanan, M.: Fundamental challenges in mobile computing. In: Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing, PODC 1996, pp. 1–7. ACM, New York (1996)
[24] Weiser, M.: The computer for the 21st century. Scientific American 3(3), 3–11 (1991)
[25] Hansmann, U., et al.: Pervasive Computing: The Mobile World, 2nd edn. Springer (2000)
[26] West, M.T.: Ubiquitous computing. In: Proceedings of the 39th ACM Annual Conference on SIGUCCS, SIGUCCS 2011, pp. 175–182. ACM, New York (2011)
[27] West, M.T.: Ubiquitous computing. In: Proceedings of the 39th ACM Annual Conference on User Services Conference, SIGUCCS 2011, pp. 175–182. ACM, New York (2011)
[28] Parkkila, J., Porras, J.: Improving battery life and performance of mobile devices with cyber foraging. In: 2011 IEEE 22nd International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), pp. 91–95 (2011)
[29] Patel, P., et al.: Towards application development for the internet of things. In: Proceedings of the 8th Middleware Doctoral Symposium, MDS 2011, pp. 5:1–5:6. ACM, New York (2011)
[30] Perkins, C.E.: Mobile networking in the Internet. Mob. Netw. Appl. 3(4), 319–334 (1998)
[31] Parviainen, M., Pirinen, T., Pertilä, P.: A speaker localization system for lecture room environment. In: Renals, S., Bengio, S., Fiscus, J.G. (eds.) MLMI 2006. LNCS, vol. 4299, pp. 225–235. Springer, Heidelberg (2006)

A Socialized System for Enabling the Extraction of Potential Values from Natural and Social Sensing

Ryoichi Shinkuma[1], Yasuharu Sawada[1], Yusuke Omori[1], Kazuhiro Yamaguchi[2], Hiroyuki Kasai[3], and Tatsuro Takahashi[1]

[1] Graduate School of Informatics, Kyoto University, Japan
shinkuma@i.kyoto-u.ac.jp
[2] Kobe Digital Labo, Inc, Japan
k-yamaguchi@kdl.co.jp
[3] The University of Electro-communications, Japan
kasai@is.uec.ac.jp

Abstract. This chapter tackles two problems we face when extracting values from sensing data: 1) it is hard for humans to understand raw/unprocessed sensing data and 2) it is inefficient in terms of management costs to keep all sensing data 'usable'. This chapter also discusses a solution, i.e., the socialized system, which encodes the characteristics of sensing data in relational graphs so as to extract values that originally contained the sensing data from the relational graphs. The system model, the encoding/decoding logic, and the real-dataset examples are presented. We also propose a content distribution paradigm built on the socialized system that is called SocialCast. SocialCast can achieve load balancing, low-retrieval latency, and privacy while distributing content using relational metrics produced from the relational graph of the socialized system. We did a simulation and present the results to demonstrate the effectiveness of this approach.

1 Background

The recent development of sensing devices has enabled us to collect many kinds of sensing data from the natural and social environments around us. Sensing data here include historical data that people generate in society when they visit somewhere, purchase something, communicate with others, as well as typical sensing data about temperature, and humidity [1][2]. We believe such sensing data contain 'value', which could stimulate our potential abilities and raise our intelligence levels. For instance, such sensing data are expected to let us recognize potential needs of people or society [3]. Although it is still an open issue as to how 'value' is defined and measured, one possible way could be that we define how big a value is compared to how large its expected economic impact will be, i.e., how much money the value is expected to produce. However, we face two fundamental problems to be solved to leverage sensing data. First, it is

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*, Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_16

hard for humans to understand raw/unprocessed sensing data [4] and second, it is inefficient in terms of management costs to keep all sensing data 'usable' because the size of sensing data is huge and they even include even sensing data that are very unlikely to be referred to [5]. Note that 'usable' here means that sensing data are not just archived but are stored in a database system that accepts queries to discover required data from outside and responds quickly to them. An option for us is not to keep sensing data usable but to keep their characteristics usable so that, at least, we can extract values contained in them. A typical example of this is that it is inefficient we keep the historical GPS data of all mobile users usable, while it might be sufficient to retain the characteristics of the data. One of the typical statistical-characteristics could be the places and times many people stayed at specific locations, which could provide useful information to the marketing departments of retailers. However, this is just one of the hypotheses we humans could expect in the values extracted from sensing data. If we only select a part of the data from the whole sensing data according to hypotheses predefined by humans and archive the rest of the data, we might lose numerous opportunities to obtain values that could be extracted from them. Therefore, our objective here should be to model characteristics of sensing data that could produce values without any hypotheses predetermined by humans and keep them usable.

We propose a new-generation system to solve these that models the characteristics of sensing data and extracts values from these characteristics, which is called a socialized system[6][7][8][10]. The socialized system produces network graphs from various kinds of sensing data. Suppose that, when you send and receive e-mails to and from your colleagues, you are connected with each of them via a link, which is a network-graph representation of your relationships extracted from the historical data of e-mail communications[11]. To apply this to other people and integrate your network graph and other obtained network graphs into a network graph, the structure of the integrated network graph represents your characteristics, i.e., who has a close relationship with you, how many people have close relationships with you, and how 'central' you are in a community. The socialized system develops the above concept so that it can be applied to any general sensing data. For example, if we produced a network graph from historical data about locations people visited and what they purchased, the network graph would not only represent people but also locations and products as nodes and the relationship between any pair of them as a link. Then, we could expect that the structure of the produced network graph would model the characteristics of people, locations, and products and it would retrieve the values extracted from the historical data of people's movements and purchases.

The following sections are organized as follows: Section 2 and 3 will discuss the system model and algorithms of the socialized system. Then, Section 4 will provide network-graph examples of the socialized system. We will also describe SocialCast, which is a new content delivery paradigm built on the socialized system in Section 5. The last section concludes this chapter.

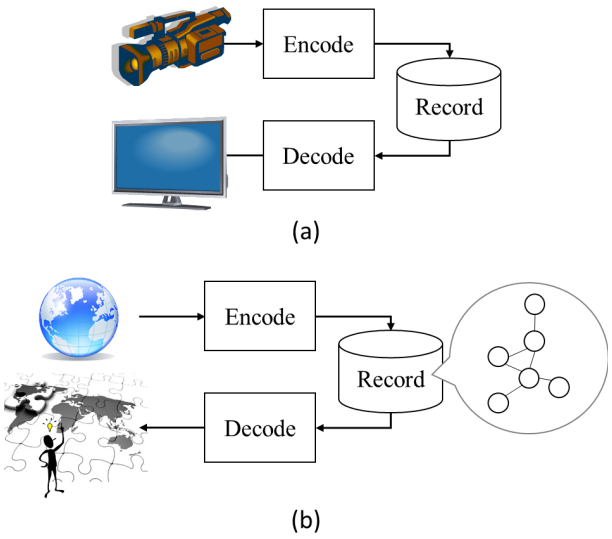


Fig. 1. (a) Image system (b) Socialized system

2 System Model

Before discussing the system model of the socialized system let us first consider, a simple system model that enables us to record data and then to reproduce their characteristics. As seen in Fig. 1 (a), first, a real space is captured by a high-resolution camera device. The input image of the real space is encoded so that it can be recorded as digital data. Then, the recoded data are decoded so that the output image can be displayed on a high-resolution display device. A question raised here is what is required for the encoder. The answer should be not to lose the characteristics of the input image expected to be reproduced when it is decoded and displayed. A simple example is that, if color is expected to be reproduced on the display device, the color characteristics have to be encoded without loss. As seen in Fig 1 (b), the socialized system can be illustrated in the same way as in Fig. (a). The encoder in the socialized system produces network graphs from the input sensing data, which are called relational graphs and whose structure models the characteristics of the original sensing data. The decoder reproduces the characteristics of the original sensing data.

3 Encoding and Decoding Logic

This section discusses the basic logic to design the encoder and the decoder in Fig 1(b).

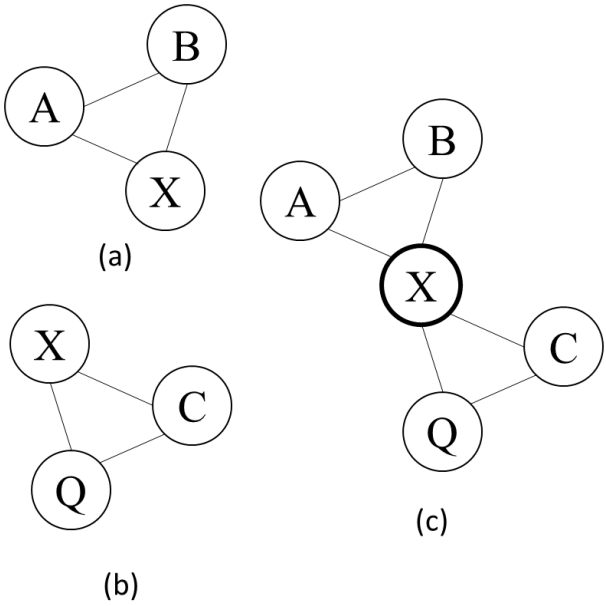


Fig. 2. Production of relational graph

3.1 Encoding Logic

Suppose that two people A and B visited location X. Once this historical event is input to the encoder, the encoder produces three nodes that correspond to A, B, and X. Then, by placing a link between each pair, the small relational graph illustrated in Fig. 2(a) is produced. Furthermore, if person C visited location X and purchased product Q at location X, the encoder can produce another small relational graph (2(b)). Let us consider integrating the first and second relational graphs into a bigger relational graph. As seen in Fig. 2(c), nodes A and B and nodes C and Q are indirectly connected via node X. Thus, as a larger number of partial relational graphs are integrated via common nodes, the integrated relational graph grows. In addition, if the socialized system is capable of recording the link length of each link as well as the existence of a link between two nodes, it is reasonable that we can increase the strength of the link between two nodes that input into the system more frequently. Note that the encoder of the socialized system uniformly deals with people, locations, and products and it does not define categories or change how we deal with data according to categories.

3.2 Decoding Logic

Relational graphs enable us to decode various kinds of characteristics that other network graphs generally have. The simplest one is if a node is directly connected

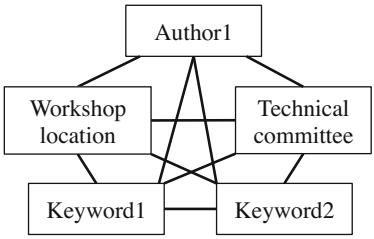


Fig. 3. Example of partial graph obtained from technical paper

with another node via a direct link. Seven examples of decodable characteristics are listed below:

1. Connectivity between a pair of nodes: no connection, direct connection via a link, or indirect connection via other nodes
2. Link strength between two directly connected nodes
3. Number of common nodes between two directly connected nodes
4. Shortest route between two indirectly connected nodes
5. Number of links each node has (degree)
6. Centrality of each node
7. Clustering characteristics of the graph

The characteristics listed above have been conventionally discussed about human networkgraphs [9], nodes represent people. Therefore, discussing the centrality of a node means discussing the centrality of a person who corresponds to that node. However, relational graphs represent many other kinds of objects than people, in which, for example, discussing the centrality of a node might mean discussing the centrality of a product, a location, or something else. If some product or some location has a high centrality, it means the product and the location might have a significant impact on the market. Note that the centrality of the location and the product explains how significant it is for other related objects, i.e., other locations, other products, and other people in the relational graph, which is different from just statistical number of visits of the location or of sales of the product.

4 Examples Using Real Datasets

We used three real datasets and applied encoding and decoding to the socialized system.

4.1 Encoding Process

The Enron email dataset contains all kind of emails that are personal and official [11]. Dr. William Cohen from Carnegie Mellon University put up the dataset on the Web for researchers. This dataset contains around 517,431 emails from 151

Table 1. Characteristics of relational graphs we produced

Dataset	No. of nodes	Avg. hop distance	Clustering coefficient
Enron	4908	4.10	0.309
UCSD	803	2.10	0.372
IEICE	21699	3.23	0.855

users distributed in 3500 folders. Each message presented in the folders contains the senders' and the receivers' email addresses, dates and times, subjects, body text, and some other technical details specific to emails. We produced a relational graph from the dataset as Shetty and Abid [11] did even though they called it as social network. A link between two people is only established if they exchange e-mails at least five times. We only considered bidirectional links, which means that we only considered a contact had occurred if both sent emails to each other. What we additionally did and Shetty and Abidi did not do [11] was to remove unsent or duplicated e-mails. Furthermore, we assumed the number of e-mails sent/received between two people would equal the link strength between them.

McNett and Voelker [12] collected movement trace data from approximately 275 mobile users who were students at the University of California in San Diego for an 11-week period in 2002. Each mobile device they used was equipped with a Symbol Wireless Networker 802.11b Compact Flash card. They identified users according to their registered wireless card MAC address, and assumed that there was a fixed one-to-one mapping between users and wireless cards. However, we produced a relational graph in which people, locations, and days (weekday/weekend) were mixed as nodes and assumed that the strength of the link between two objects was equal to how long they stayed with each other.

We also used a dataset of technical reports published by the Institute of Electronics, Information, and Communication Engineers (IEICE) in Japan [13], where each report included four kinds of objects on authors, technical committees, workshop locations, and keywords. We obtained a full-mesh partial graph from each technical report in which objects were represented as nodes and connected with once another like they are in Fig. 3. Then, all the partial graphs generated from technical reports could be integrated into one big relational-graph because partial graphs could be connected via the common nodes between them. The link strength between two nodes was determined by how many times the link appeared when partial graphs were integrated. We used the datasets from the technical reports from 2008 to 2010 of the Communications Society of IEICE.

4.2 Decoding Process

The original purpose of decoding in the socialized system should be to decode characteristics that obtain values; in other words, its expected functionality is to evaluate how valuable each node in the relational graph is. However, we tried to

decode characteristics in basic observation that any network graphs generated in nature or society should have from the three relational graphs we produced.

We will first discuss, whether the relational graphs we produced had the small-world characteristics [14]. If a network-graph has small-world characteristics, the average hop distance of the shortest route between two nodes is generally very close relative to the total number of nodes in the network. We here simply consider the number of hop counts between two nodes as the distance between them; we did not consider the link strength or the link length. As summarized in Table 1, the average hop distance of every relational graph we produced is quite short relative to the number of nodes.

We will then discuss whether the relational graphs we produced had clustering characteristics [15]. If nodes in a network graph form clusters, the clustering coefficient of the network should generally be around from 0.1 to 0.7. As listed in Table 1, it is obvious that the clustering coefficients of the relational graphs we produced are in the range from 0.1 to 0.7.

Finally, we will discuss the scale-free characteristics of the produced relational graphs we produced. If they have the scale-free characteristics, the distribution of the number of links (degree) for each node follows a power-law distribution [16]. We have plotted the degree distributions of the relational graphs we produced in Figs. 4, 5, and 6, where the approximated lines on a double-log scale have also been plotted.

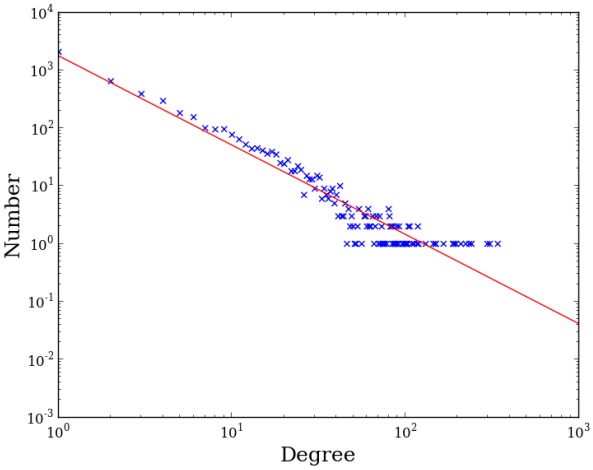


Fig. 4. Degree distribution of relational graph produced from Enron dataset, which is approximated by $y = 10^{3.25}x^{-1.54}$, where the correlation factor is -0.944

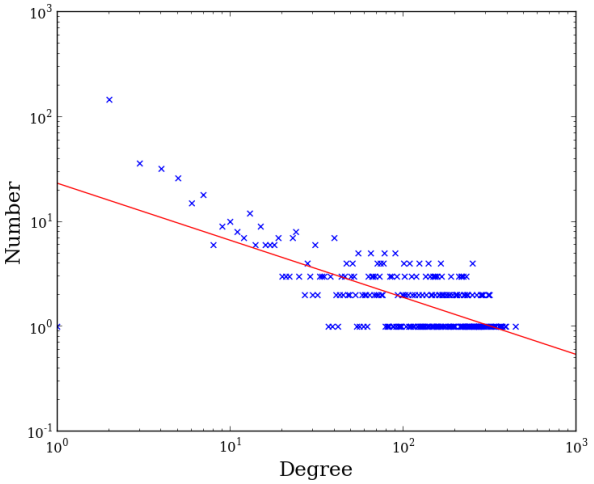


Fig. 5. Degree distribution of relational graph produced from UCSD dataset, which is approximated by $y = 10^{1.37}x^{-0.54}$ after top 20% of nodes with largest degree are removed, where the correlation factor is -0.679

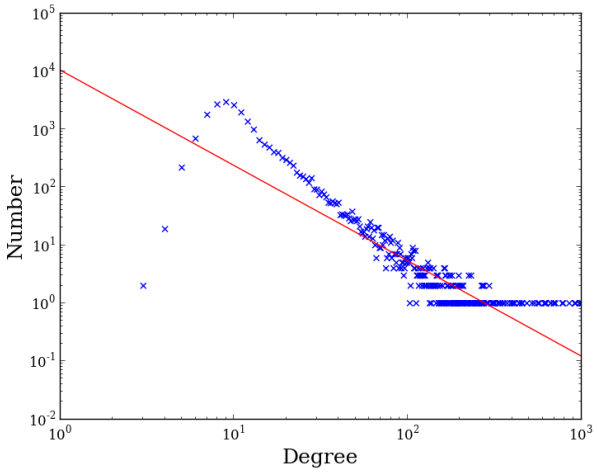


Fig. 6. Degree distribution of relational graph produced from IEICE dataset, which is approximated by $y = 10^{4.02}x^{-1.64}$ after top 10% of nodes with largest/smallest degree are removed, where the correlation factor is -0.894

5 Example of Application: Socialcast

The socialized system we discussed in the previous sections can be commercialized as follows: on one hand, the most typical applications of the socialized system could be a product recommendation system for online shoppers and a navigation system for car drivers. Why the socialized system can be applied to these services is because it enables us to decode what product or location the shopper or the driver has a close relationship with from the relational graph. Especially, even if a person and a product/location were not directly connected but were indirectly close each other via other nodes, we could expect the person to purchase the product or visit the location very likely in the near future. The centrality of the product or the location would also be useful standards to evaluate how important they were from a more general viewpoint. As readers may know, one of the most well-known algorithms used for product recommendation is cooperative filtering [17], which still relies on statistic characteristics of historical data and is an essentially different concept from the socialized system.

On the other hand, the socialized system would also work effectively in delivering digital content in computer networks. This is built on the hypothesis that digital content is requested and exchanged based on relationships between people, locations, and products, e.g., if you have a close friend that you would like to share your digital content with like your photos and videos and if you visit theme parks every month and are interested in sharing digital content on special events there. The relational graph provided by the socialized system should successfully model such relationships. We named the new delivery scheme built on the socialized system SocialCast, which will be described in detail in the next sections.

5.1 Objective

Much research has been focused on how to achieve load balancing to deal with the problem of increasing traffic on networks. Open Shortest Path First (OSPF)[21] is a common routing protocol and is used in intra-domain Internet situations. The network operator in this protocol assigns a metric to each physical link. In this section, physical routers and physical links mean physical routers in a computer network and links between the physical routers; we will use a link or a node only when it means a link or a node in the relational graph. As recommended by Cisco[22], it is often assigned the inversely proportional value of the bandwidth of each physical link. In such cases, the metric of physical links that have a high bandwidth decreases and is frequently chosen as the shortest path, which is then likely to be overloaded. Fortz and Thorup [25] suggested a way of optimizing the metrics used in OSPF to prevent physical links from such overloading. SocialCast uses relational metrics as a metric for physical links. A relational metric is produced from a relational graph and represents the degree to which two or more objects relate. For example, if user A frequently downloads a certain type of content, e.g., content X, the relational metric between user A and content X should be large. As another example, if user A and user B share private

content Y, the relational metric between user A, user B, and content Y should be large. Since relational metrics differ in terms of content to be distributed, and the physically shortest path is not always chosen as a distribution path. Load can be balanced without special techniques by using relational metrics like that in demand projection [25]

Reducing retrieval latency is also an important issue. Caching can be one approach for achieving this. Popularity is frequently used as a metric to determine which content is to be cached; the more popular content is, the more frequently it will be cached. Naturally, popular content has a high likelihood of being requested by many users, but that does not mean that retrieval latency for all kinds of content will be reduced. If popular content is preferentially cached, less popular content is less frequently cached. As the number of requests for content on the Web is known to follow a Zipf-like or power-law distribution [26], there is a considerable amount of unpopular content, and popularity-based caching cannot handle this. Coordinated caching has been proposed [27] to use distributed caches more efficiently; neighboring routers should not cache the same content and even low-popularity content should be cached. However, the main drawback of coordinated caching is its complexity. Relational metrics represent the relationships between content and users, and therefore, the metric used for cache management will differ between distributed content and users who cache content. The caches of the content can thus be effectively distributed without any complex mechanisms.

SocialCast also effectively ensures privacy and security. Considering relational metrics and calculating a path that has a large metric value means that distributed content will only be shared with those who have some relationship with the content. Calculating a distribution path on the lower layer also has an advantage because the range of disclosures will be physically limited.

5.2 System Model

System Overview. Fig. 7 has a conceptional model of our system using a layered structure. This model has three layers: a layer of the real world, a layer of publisher/subscriber (pub/sub) space, and a layer of the physical network. Each layer has a main controller: a relational metric-based controller for the real world, a pub/sub-based controller for the pub/sub space, and a network controller for the physical network. These three main controllers control each layer and cooperate with one another.

The following subsections introduce the main components of each layer and how the required functions mentioned above are satisfied.

Relational Metric-Based Controller. The relational metric-based controller is positioned as a controller for the real world. The main function of this controller is to produce relational metrics from sensing data based on the relational graph provided by the socialized system and pass them on to the lower layer in a form that can be used for content distribution. As previously stated, a relational metric represents the degree to which two or more objects relate.

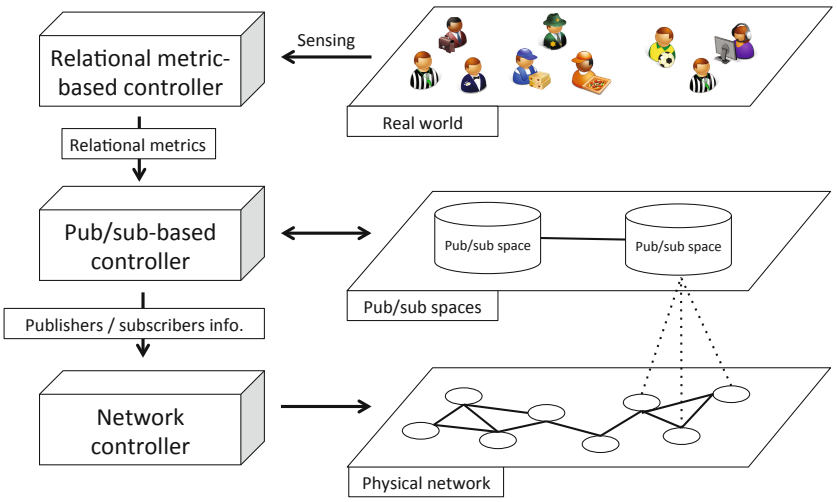


Fig. 7. Layered model for proposed system

Pub/Sub-Based Controller. The pub/sub-based controller of our system manages the pub/sub space and enables efficient distribution and matching of content routers. The pub/sub space stores logical information such as the existence and availability of the content and the publishers and the (potential) requesters of the content.

The relational metrics obtained from the relational metric-based controller are used when matching publishers and subscribers. The pub/sub-based controller considers the stored information and relational metrics, and binds the published content and routers that will receive content when the content is published. This means that content X published by user A is only sent to users who have large relational metrics in common with content X and user A; i.e., content X can be shared with others who have some relation with content X and user A.

After matching, the controller passes information on publishers/subscribers, content, and the obtained relational metrics to the lower layer and directs the physical distribution of content.

Network Controller. The network controller physically manages routers and contents on the network. This controller has three major functions described below.

The first function is to be aware of the physical states of routers and determine a physical distribution path by taking into consideration the relational metrics of each router in the network associated with the delivered content and the publisher. Relational metrics obtained from the upper layer will be used when determining this path. When the network controller is triggered for content distribution, the controller looks for a distribution path to all users bound with

the content and determines the distribution path using the relational metrics provided by the upper layer.

The second function is to manage content, which includes content discovery and cache replacement. The network controller stores the physical location of all content in the network, i.e., what content exists in each router on its own cache. When published content is delivered, the caches of the routers on the distribution route can be replaced. The updated cache status should be known by the network controller for the next discovery of content.

The third function is to direct a specific router to redistribute the content of its cache. In essence, the router acts like a cache server on content delivery networks (CDNs) [24]. After the distribution path is established, the controller checks if any router on the path has the content in its cache. If any router has the content to distribute on its cache, the content will be distributed from the router that has the cache of the content instead of the original source of the content.

5.3 Proposed Method

Mathematical form of Relational Metrics. Assume $G = (V, E)$ is a bidirected graph [28] and all edges are assigned positive values, which is the generalization of undirected graphs. Relational metrics of edges will be calculated by the function

$$m(e_{ij}) = \begin{cases} f(i,j,c) & (if \;\; e_{ij} \in E(G)) \\ 0 & (otherwise), \end{cases} \tag{1}$$

where $m(e_{ij})$ is the relational metric value assigned to directed edge e_{ij}, and i represents the tail and j represents the head of the edge in edge e_{ij}. Here, $E(G)$ is the set of all edges in graph G, c is the content that will be distributed, and $f()$ represents the function to generate relational metric value. This equation means $m(e_{ij})$ is used as the metric to forward content c from router i to the next router, j,

Algorithm for Building Distribution Path. When building a distribution path, SocialCast chooses the path that has the minimum sum of the reciprocal values of the relational metric to the destination router.

The algorithm for building the distribution path follows the steps below. The basic idea is the same as building the shortest path with Dijkstra's algorithm [29]. Our proposed method uses the reciprocal value of the relational metric (calculated as described in Subsection 5.3) as the distance of each physical link. The building distribution path follows the two steps below.

First, we assign the physical link distance on each physical link. The distance of each physical link is calculated on the basis of the content to be distributed, c. All edges in the graph are assigned the reciprocal value of the relational metric

of the physical link as the distance of the physical link if the physical link exists in the graph, on $+\infty$ otherwise. This means the distance of edge e_{ij} is calculated by

$$d(e_{ij}) = \begin{cases} \frac{1}{f(i,j,c)} & (if \;\; e_{ij} \in E(G)) \\ +\infty & (otherwise), \end{cases} \tag{2}$$

where $d(e_{ij})$ represents the distance of a directed edge from tail router i to head router j.

Second, calculate the shortest path to the destination router u. Shortest path p_u is built using the algorithm described in problem 2 of Dijkstra [29].

Caching Algorithm. Assume C_i represents the capacity of the cache of router i and $\mathbb{C}_i^t$ represents the set of cached content of router i at time t.

We use enroute caching and content passing router i can be cached [30]. When caching the content, the relational metric value, $f(i, j, c)$, among router i, and distributed content, c, are also stored as $v(c)$. If content c passed router i at time t, the cache of router i at time $t + 1$ would be

$$\mathbb{C}_i^{t+1} = \mathbb{C}_i^t \cup \{c\} \tag{3}$$

If the cache of a router is about to exceed its capacity when caching content, cache replacement will occur. This replacement is done by solving the optimization problem written as

$$\max_{c \in \mathbb{C}_i^{t+1}} \sum_{j \in \mathbb{C}_i^{t+1} \setminus \{c\}} v(j) \tag{4}$$

5.4 Simulation Model

We first set up a simulation model to evaluate the effectiveness of our system, and investigated the performance of our proposed method where the overheads caused by fetching physical links and relational metric information were ignored to simplify the model. The following sections describe the model and the conventional method used in this simulation in detail.

System Description. We set up a scenario in the simulation to determine whether or not our proposed method, which uses the relational metric-based distribution path and cache mechanism described in Subsection 5.3, was actually more effective than the conventional approach.

Routers are randomly distributed over the simulation field in the scenario. The pub/sub space, which was the only distributor of content in the simulation, was at the center of the simulation field, and all routers including pub/sub space had physical connectivities with routers within the range of radius r. Routers were redistributed randomly in the simulation until all routers had at least one path to reach the pub/sub space. This physical network was meant to represent an abstract model of a local area network with wired/wireless physical links.

Table 2. Parameters used in simulation

Size of simulation field	200 m × 200 m
No. of kinds of content	5209
No. of routers (n)	3896
Physical communication range (r)	30 m
Bandwidth of each physical link (B)	100 Mbps
Size of content (C_s)	10 MB
Size of cache capacity of each user (C_i)	1, 3
Threshold (t_{stop})	3000

Simulation Scenario. The simulation scenario unfolds as follows.

1. The relational graph was created. Each link between objects was calculated using the method described as in Subsection 4.1.
2. The physical network was created. Each router had connectivity with routers within the range of radius r. All users were randomly assigned to each router in the physical network.
3. After the physical network was created, the distribution was processed until the caches of all routers were filled. The method of cache replacement method followed Eqs. (3) and (5).
4. One of the users is randomly selected in time slot t as a requester of content. The content to be distributed was chosen from content that the requester was directly interested in. The chosen content was regarded as content that the requester asked for in the simulation. The possibility of which content was chosen based on the relational metric between each content and the requester.
5. The distribution path was constructed based on the proposed method described in Susection 5.3 or the conventional method, which will be described in Subsection 5.4. The content was actually distributed along the path to the routers that destination users were assigned to. During the distribution, all routers placed and replaced their own caches according to Eqs. (3) and (5). We check which physical link was used in each distribution, the averaged latency until targeted users retrieved content, and the relational metric of the physical links on the distribution path.
6. We incremented time slot t, and repeated steps 3 and 4 until it reached threshold t_{stop}.

Parameters. We used 2460 technical reports to produce relational metrics, which were published by the Communications Society of IEICE in 2010. We assumed 'keyword' and 'author' in the relational graphs in our simulation corresponded to content and a router; the distances between the keyword that corresponds to the content and the authors who correspond to routers i and j in the relational graph are measured to forward content from router i to the next router, j, and the average of them is used as the relational metric for the

physical link. The physical link distance was normalized in a range from 0.0 to 1.0. Other simulation parameters were listed in Table 2. The bandwidth of all physical links of the physical network and the size of all content to be distributed in the simulation corresponded to constant B and C_s. Also, all routers had the same constant cache capacity, C_i in the simulation. Threshold t_{stop} was the threshold for the time slot to stop simulation.

Conventional Method. We used a conventional minimum physical distance path distribution with a popularity-based cache mechanism for comparison.

Minimum physical distance path distribution. Distance $d(e_{ij})$ of all edges $e_{ij} \in E(G)$ was set to 1.0. That means the length of the distribution path was regarded as hop counts on the path. Apart from this, the steps to construct the distribution path were the same as those described in Subsection 5.3.

Popularity-based cache mechanism. If the cache did not exceed capacity, the distributed content was cached according to Eq. (3). If the cache did exceed capacity, the cache was replaced by solving:

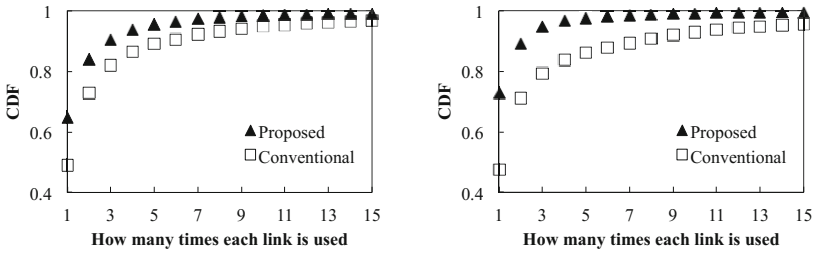
$$\max_{c \in \mathbb{C}_i^{t+1}} \sum_{j \in \mathbb{C}_i^{t+1} \setminus \{c\}} popularity(j), \tag{5}$$

where $\mathbb{C}_i^{t+1}$ represents the cache set on router i at time slot $t+1$, c, and j are content, and $popularity(j)$ represents the popularity of content j. The popularity of content j was assigned in the simulation according to the number of users who directly had an interest in content j. Since the distribution of the number of users assigned to each content followed a power law, the popularity of each content also follows the power law. Therefore, we assumed the model for users and content and the model for assigning popularity to all content were appropriate for considering the situation with the distribution of the contents that have power-law popularity [31].

Simulation Results. This section compares the results obtained from the conventional and proposed methods for each item when cache size C_i for all routers in the physical network was equal to one and three. We will then discuss the possible reasons for these.

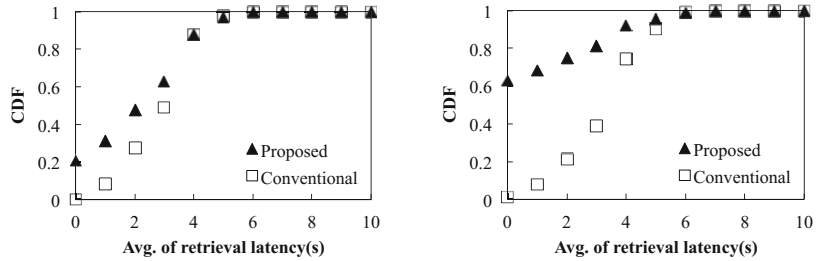
Load balance. We counted the times of each physical link was used for content distribution and calculated the ratio of the times each physical link was used to the number of times all physical links were used to evaluate load balance. We also compared the cumulative distribution function to the number of times each physical link was used in all used physical links with the conventional and proposed methods.

Figs. 8(a) and 8(b) plot the results for load balancing for the conventional and proposed methods when the cache sizes of the routers were $C_i = 1$ and $C_i = 3$. The proportion of physical links only used once in the distribution



(a) $C_i = 1$ for all router i. (b) $C_i = 3$ for all router i.

Fig. 8. Ratio of times each physical link used to sum of times of all physical links used during simulation trial



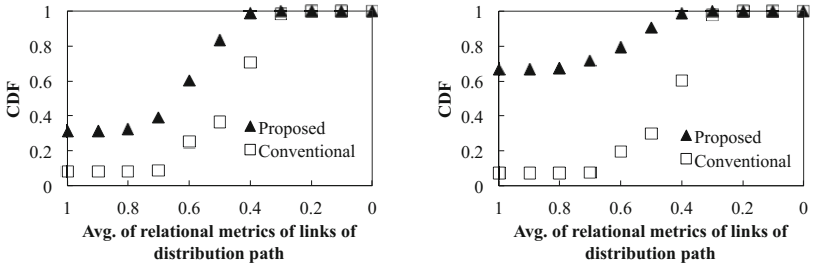
(a) $C_i = 1$ for all router i. (b) $C_i = 3$ for all router i.

Fig. 9. Cumulative distribution function of content retrieval latency for users

using the conventional method is lower than that using the proposed method. That means many physical links are used multiple times in the distribution with the conventional method. This is because the conventional method always uses the physically shortest path, which does not change from content to content. However, since the proposed method takes into consideration the relationship between intermediate routers and the content to be distributed, it works so that the distribution path will differ.

Average of retrieval latency for users. We measured the retrieval latency for each user to evaluate the averaged retrieval latency. Here, retrieval latency means the time between content being published and a user obtaining it. We then calculated the averaged time and compared the cumulative distribution function to the averaged retrieval latency.

Figs. 9(a) and 9(b) plot the results for averaged retrieval latencies with the conventional and proposed methods when the cache sizes of the routers were $C_i = 1$ and $C_i = 3$. Here, the proposed method out-performed the conventional method: 50% of all content distributed reached targeted users within about 3.0 s



(a) $C_i = 1$ for all router i. (b) $C_i = 3$ for all router i.

Fig. 10. Cumulative distribution function of averaged relational metric value of each physical link in distribution path

with the conventional method, while it only took about 2.0 s with the proposed method when the cache size is equal to one. A possible reason for this is that the cache of content is well distributed with the proposed method. As described in Subection 5.4, the distribution path will differ depending on content c, so different content is cached on different routers. If a cache is well distributed, the probability of the existence of a router that has a cache of the requested content near the routers increases, and the latency then becomes smaller because the content can be distributed from the cache. The results for when the cache size is equal to three are even better. This is because, if routers have a bigger cache, the probability of the existence of a router that has a cache of the requested content near the routers increases even more.

Privacy/security. We measured the relational metric value of each physical link in the distribution path to evaluate privacy/security and compared the cumulative distribution function to the averaged relational metric value of each physical link in the distribution path. Since the relational metric represents the relationships between objects, it automatically includes some sense of privacy. If this is the case, the averaged relational metric value of physical links in the distribution path in each distribution can be regarded as how related routers in the distribution path are.

Figs. 10(a) and 10(b) plot the results for the averaged relational metric value of each physical link with the conventional and proposed methods when the cache size is $C_i = 1$ and $C_i = 3$ for all routers i. Figs. 10(a) and 10(b) reveal that the proposed method includes more related routers in the distribution path than the conventional method does: 80% of all distribution paths are higher than 0.4 for the averaged relational metric value with the conventional method when $C_i = 3$, while 80% of all distribution paths are less than about 0.6 with the proposed method when $C_i = 3$.

6 Conclusion

This chapter has tackled two problems we face when extracting values from sensing data: 1) it is hard for humans to understand raw/unprocessed sensing data and 2) it is inefficient in terms of management costs to keep all sensing data 'usable'. This chapter also discussed a solution, i.e., the socialized system, which encodes the characteristics of sensing data in relational graphs so as to extract values that originally contained the sensing data from the relational graphs. The system model, the encoding/decoding logic, and the real-dataset examples were presented.

We also proposed a content distribution paradigm built on the socialized system that was called SocialCast. SocialCast can achieve load balancing, low-retrieval latency, and privacy-conscious delivery by distributing content using relational metrics produced from the relational graph of the socialized system. We did a simulation and presented the results to demonstrate the effectiveness of this approach.

We here list the four remaining issues that need to be addressed to commercialize the socialized system:

Definition of nodes. A person can be naturally dealt with a node in a relational graph because she/he cannot be 'divided' into a smaller unit. However, since a location can be variously defined like coordinates, addresses, buildings, or blocks, defining a location as a node in a relational graph is an issue.

Definition of temporal sensing range. Even when we observe a person at a location, we might not need to input a log to the socialized system and establish a link between them at the relational graph if the person is actually just passing through the location only for a second. This suggests we should define a temporal sensing range for the socialized system.

Definition of spatial sensing range. Even when we simultaneously observe two people in a geographical area, we might need to input a log to the socialized system and establish a link between them in the relational graph. However, this depends on the geographical closeness between them. Therefore, we should define an appropriate spatial sensing range for the socialized system.

Growth and aging of relations. When we generally consider relationships between things that could be between people, locations, products, or different kinds of things, it is inevitable to consider growth and aging. The socialized system should enable the relational graphs to grow and age as relationships between things in society change.

Acknowledgments. This work is supported in part by the National Institute of Information and Communications Technology (NICT), Japan. The authors would like to thank Dr. Oscar Mayora, Dr. Francesco De Pellegrini, and Dr. Elio Salvadori, CREATE-NET, Italy for their contributions from academic

viewpoints. The authors also would like to thank the industrial forum for Mobile Socialized System (MSSF), Japan for their contributions from industrial viewpoints.

References

1. Aizawa, K., Tancharoen, D., Kawasaki, S., Yamasaki, T.: Efficient retrieval of life log based on context and content. In: Proceedings of the the 1st ACM Workshop on Continuous Archival and Retrieval of Personal Experiences (CARPE 2004), pp. 22–31 (2004)
2. Laurila, J., Gatica-Perez, D., Aad, I., Blom, J., Bornet, O., Dousse, D.O., Eberle, J., Miettinen, M.: The mobile data challenge: Big data for mobile computing research. In: Proceedings of Mobile Data Challenge by Nokia Workshop (2012)
3. LaValle, S., Lesser, E., Shockley, R., Hopkins, M.S., Kruschwitz, N.: Big Data, Analytics and the Path From Insights to Value. MIT Sloan, Management Review 52(2) (2011)
4. Lymberopoulos, D., Bamis, A., Savvides, A.: Extracting spatiotemporal human activity patterns in assisted living using a home sensor network. In: Proceedings of the 1st International Conference on PErvasive Technologies Related to Assistive Environments (PETRA 2008), Article No. 29 (2008)
5. Lynch, C.: Big data: How do your data grow? Nature 455, 28–29 (2008)
6. Shinkuma, R., Kasai, H., Yamaguchi, K., Mayora, O.: Relational Metric: A New Metric for Network Service and In-network Resource Control. In: Proceedings of IEEE Consumer Communications and Networking Conference (CCNC 2012), Work-In-Progress session (2012)
7. Kida, A., Shinkuma, R., Takahashi, T., Yamaguchi, K., Kasai, H., Mayora, O.: System Design for Estimating Social Relationships from Sensing Data. In: Proceedings of IEEE International Conference on Advanced Information Networking and Applications (AINA 2013), Workshop on Data Management for Wireless and Pervasive Communications (2013)
8. Yogo, K., Kida, A., Shinkuma, R., Kasai, H., Yamaguchi, K., Takahashi, T.: Extraction of Hidden Common Interests between People Using New Social-graph Representation. In: Proceedings of International Conference on Computer Communications and Networks (ICCCN 2011), Workshop on Social Interactive Media Networking and Applications (August 2011)
9. Borgatti, S.P.: Centrality and network flow. Social Networks 27(1), 55–71 (2005)
10. Nishio, T., Shinkuma, R., Pellegrini, F.D., Kasai, H., Yamaguchi, K., Takahashi, T.: Trigger Detection Using Geographical Relation Graph for Social Context Awareness. Mobile Networks and Applications 17(6), 831–840 (2012)
11. Shetty, J., Adibi, J.: The Enron email dataset database schema and brief statistical report. database schema and brief statistical report. Information Sciences Institute, vol. 4 (2004)
12. McNett, M., Voelker, G.M.: Access and mobility of wireless pda users. Technical report, Computer Science and Engineering, UC San Diego (2004)
13. The Institute of Electronics, Information and Communication Engineers (IEICE), Japan, `http://www.ieice.org/jpn/`
14. Newman, M.E.: Models of the Small World. Journal of Statistical Physics 101(3-4), 819–841 (2000)

15. Fronczak, A., Hołyst, J.A., Jedynak, M., Sienkiewicz, J.: Higher order clustering coefficients in Barabási-Albert networks. Physica A: Statistical Mechanics and its Applications 316 (1), 688–694 (2002)
16. Barabási, A.-L., Albert, R., Jeong, H.: Mean-field theory for scale-free random networks. Physica A: Statistical Mechanics and its Applications 272(1), 173–187 (1999)
17. Linden, G., Smith, B., York, J.: Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing 7(1), 76–80 (2003)
18. Pan, J., Paul, S., Jain, R.: A survey of the research on future internet architectures. IEEE Communications Magazine 49(7), 26–36 (2011)
19. Ahlgren, B., Dannewitz, C., Imbrenda, C., Kutscher, D., Ohlman, B.: A survey of information-centric networking. IEEE Communications Magazine 50(7), 26–36 (2012)
20. Carzaniga, A., Papalini, M., Wolf, A.L.: Content-based publish/subscribe networking and information-centric networking. In: Proceedings of the ACM SIGCOMM Workshop on Information-centric Networking (ICN 2011), pp. 56–61 (2011)
21. Moy, J.: OSPF Version 2. RFC 1247 (Draft Standard). Obsoleted by RFC 1583, updated by RFC 1349 (July 1991)
22. Cisco, Configuring OSPF, `http://www.cisco.com/en/US/docs/ios/120/np1/configuration/guide/1cospf.html`
23. Borst, S., Gupta, V., Walid, A.: "Distributed Caching Algorithms for Content Distribution Networks. In: Proceedings of IEEE International Conference on Computer Communications (INFOCOM 2010), pp. 1–9 (2010)
24. Vakali, A., Pallis, G.: Content delivery networks: Status and trends. IEEE Internet Computing 7(6), 68–74 (2003)
25. Fortz, B., Thorup, M.: Optimizing OSPF/IS-IS weights in a changing world. IEEE Journal on Selected Areas in Communications 20(4), 756–767 (2002)
26. Breslau, L., Phillips, G., Shenker, S.: Web caching and Zipf-like distributions: evidence and implications. In: Proceedings of 18th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 1999), vol. 1, pp. 126–134 (1999)
27. Korupolu, M., Dahlin, M.: Coordinated placement and replacement for large-scale distributed caches. IEEE Transactions on Knowledge and Data Engineering 14(6), 1317–1329 (2002)
28. Appa, G., Kotnyek, B.: A bidirected generalization of network matrices. Networks 47(4), 185–198 (2006)
29. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik 1(1), 269–271 (1959)
30. Shinkuma, R., Jain, S., Yates, R.: In-network caching mechanisms for intermittently connected mobile users. In: Proceedings of 34th IEEE Sarnoff Symposium, pp. 1–6 (2011)
31. Adamic, L.A., Huberman, B.A.: Zipf's law and the Internet. Glottometrics 3, 143–150 (2002)

Providing Crowd-Sourced and Real-Time Media Services through an NDN-Based Platform

G. Piro[1], V. Ciancaglini[2], R. Loti[2], L.A. Grieco[1], and L. Liquori[2]

[1] DEI - Politecnico di Bari, Italy
{g.piro,a.grieco}@poliba.it
[2] INRIA - Sophia Antipolis, France
{vincenzo.ciancaglini,riccardo.loti,luigi.liquori}@inria.fr

Abstract. The diffusion of social networks and broadband technologies is letting emerge large online communities of people who stay always in touch with each other and exchange messages, thoughts, photos, videos, files, and any other type of contents. At the same time, due to the introduction of *crowd-sourcing strategies*, according to which services and contents can be obtained by soliciting contributions from a group of users, the amount of data generated and exchanged within a social community may experience a radical increment never seen before. In this context, it becomes essential to guarantee resource scalability and load balancing to support real-time media delivery. To this end, the present book chapter aims at investigating the design of a network architecture, based on the emerging Named Data Networking (NDN) paradigm, providing crowd-sourced real-time media contents. Such an architecture is composed by four different entities: a very large group of heterogeneous devices that produce media contents to be shared, an equally large group of users interested in them, a distributed Event Management System that creates events and handles the social community, and an NDN communication infrastructure able to efficiently manage users requests and distribute multimedia contents. To demonstrate the effectiveness of the proposed approach, we evaluate its performance through a simulation campaign based on real-world topologies.

1 Introduction

Thanks to online social networks (like Facebook, Google+, LinkedIn, Myspace, Twitter, and Spotify), which are experiencing an explosive growth since the past few years, people can always stay in touch with each other and exchange messages, thoughts, photos, videos, files, and any other type of contents. This growth is sustained by the widespread adoption of new generation devices (such as notebooks, smart phones, and tablets) [1], together with emerging broadband wired and wireless technologies and, without any doubt, it will become more and more evident in the coming years. As a final result, people continuously generate and request a massive amount of data, provided by other users worldwide.

Horizons of online social networks can be further extended, thus enhancing users' interaction and data discovery, by introducing *crowd-sourcing approaches*,

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_17

which refer to the practice of obtaining services and contents by soliciting contributions from a group of people who, in most cases, form an online community [2].

When used together, online social networks, mobile devices, and crowd-sourcing platforms have all the potential to create connected communities scattered all over the world, thus paving the way to several novel applications. Their joint exploitation, for example, can be used to capture media contents, i.e., audio and video, from a very large number of users during an event (think for example of a football match, a concert, a public event, a dangerous situation, and so on) and delivering them in real-time to any other user. This way, anyone can see (and listen) what others see (and listen) with their eyes (and ears) in a specific place, while being physically far. To the best of our knowledge, network architectures enabling such kind of *social audio-video real-time services* have not yet been fully standardized. Hence, novel ideas and technologies have to be promoted by the research community in order to make this a very attractive application achievable as soon as possible.

In parallel, the so-called Information Centric Network (ICN) approach is emerging to foster a transition from a host-centric to a content-centric Future Internet [3]. The ICN approach is currently investigated and developed in several projects, such as Data-Oriented Network Architecture (DONA) [4], Publish Subscribe Internet Technology (PURSUIT) [5], Scalable and Adaptive Internet Solutions (SAIL) [6], CONVERGENCE [7], and Named Data Networking (NDN) [8], COntent Mediator architecture for content-aware nETworks (COMET), and MobilityFirst [9].

Despite some distinctive differences (e.g., content naming scheme, security-related aspects, routing strategies, cache management), they share a common receiver-driven data exchange model based on content names [10,3].

In the authors' opinion, the ICN represents a powerful technological basis to distribute *crowd-sourced* contents. Unfortunately, despite the number of works available in the literature that investigate and propose innovative techniques to deliver contents in ICN, solutions properly designed for the considered scenario have not been proposed yet.

To bridge this gap, we conceive herein a network platform based on the NDN rationale, which is able to efficiently discover and distribute *crowd-sourced* multimedia contents within a distributed and online social network. The devised platform is composed of (i) a community of users that are in the same place to take part in an event record and broadcast data and multimedia streams from their multiple points of view, (ii) a number of remote users interested in such information, (iii) a distributed *Event Management System*, which creates events and handles the social community, and (iv) an NDN communication infrastructure able to efficiently manage users requests and distribute contents. Moreover, it addresses four different tasks: *event announcement*, *event discovering*, *media discovering*, and *media delivering*. In addition, to offer an optimal management of multiple and heterogeneous events, we also designe a hierarchical *name-space* and a *sliding-window* scheme to efficiently download real-time media contents through NDN primitives. We evaluate the performances of the our proposal through the *ccnSim* simulator [11]. In particular, focusing the attention

on a complex network architecture, which is composed of 68 routers connected among them according to the Deutsche Telekom topology, we evaluate the performance of multimedia services by varying the number of users that participate at a given event and generate different media contents, the amount of network bandwidth dedicated to this service, and other parameters characterizing the *sliding-window* mechanism. Simulation results will clearly demonstrate that the conceived service architecture always guarantees the highest Quality of Experience (QoE) to end users with respect to a baseline scenario where NDN features are not implemented.

The rest of the chapter is organized as follows: a background on social networks, audio-video distribution in Peer-to-Peer (P2P) systems, *crowd-sourcing* paradigm, as well as on the emerging NDN network architecture, is provided in Sec. 2. An accurate study of the state of the art related to the management of streaming services in social, *crowd-sourced*, and NDN-based platforms will be presented in Sec. 3, whereas the analysis of issues and challenges arising from the considered scenario will be discussed in Sec. 4. The conceived NDN-based solution will be deeply described in Sec. 5. Moreover, a performance evaluation of the proposed approach will be proposed in Sec. 6. Finally, Sec. 7 will conclude this chapter and draw future research.

2 Preliminary Concepts, Background, and Definitions

2.1 Online Social Networks and the *Crowd-sourcing* Paradigm

The past 10 years have witnessed the rise of *Online Social Networks* as the predominant form of communication over the Internet. They are a software platform, in the form of a mobile application or an internet website, that enable social relations among people, and the exchange of information between them.

Users of an online social network usually maintain a list of first-degree contacts (i.e., friends and families, or direct collegues), with whom executing direct interactions (i.e., the exchange of contents). The type of contents being published depends on the scope of the Social Network. Accordingly, we can have:

- general purpose networks, such as Myspace [12] or Facebook [13];
- geographically centered social networks, such as Sina Weibo [14] for the Chinese market, Orkut [15] and Hi5 [16] in South America, NK [17] in Poland, VKontakte [18] in Russia;
- micro-blogging platforms, supporting only content in the form of text posts of limited length, such as Twitter [19], Thumblr [20], Identi.ca [21];
- media-driven online social networks, i.e., networks dedicated to sharing a particular kind of media only. This is the case of Vine [22], Flickr [23], or Spotify [24];
- interest-driven Online Social Networks, where the social graph between a user and its contacts is created according to specific criteria (common profession, similar interests). This is the case of LinkedIn [25].

The emerging *crowd-sourcing* paradigm refers to the practice of obtaining services and contents by soliciting contributions from a group of people and/or devices worldwide [2]. Online social networks may natively enable the distribution of *crowdsourced media content* since they are composed by users that are both creators and fruitors of data at the same time.

When used to distribute crowdsourced media contents, online social networks could produce a variety of media contents much higher than a normal media server. At the same time, each user could potentially manage the download and the upload of a significant amount of data, thus becoming the bottleneck of the service. As a consequence, it is necessary to adopt specific network architectures able to efficiently support crowd-sourced and real-time media services, while guaranteeing an acceptable quality of service to end users.

2.2 Collaboration in Real-Time Video Distribution: P2P-TV

In a gossip-based Peer-to-Peer (P2P) system, nodes interested in the same content, namely peers, buildup a high-level overlay network [26]. In that architecture, each node establishes links with a limited set of peers called neighbors, exchanges pieces of data with them, and receives contents from multiple sources. In general, every peer offers its own upload bandwidth for content distribution, thus eliminating the need for high-capacity servers. This advantage has been fruitfully exploited in the past to design powerful file transfer applications [27].

Recently, the attention of the scientific community and industry has turned toward P2P-TV, which represents a P2P architecture, generally built as an overlay network, mainly devoted to the distribution of media contents [28]. In particular, a multimedia data source generates a series of *chunks*, which store a portion of the audio/video bitstream, and makes them available to all peers connected to the overlay [29]. Accordingly, P2P-TV could represent a valid architecture enabling streaming services in online social networks.

With respect to common file sharing applications, P2P-TV systems need for strict delay requirements: a chunk received after that its *playout delay* (i.e., the deadline) has elapsed, will be considered lost [30]. In this context, the higher is the playout delay, the higher is the amount of chunks received by the user. However, this advantage comes at the expense of the QoE of the users, which is very sensitive to the timeliness of TV services [31]. On the other hand, performances of P2P-TV systems are also influenced by both bandwidth and content bottlenecks [32]. It is possible, in fact, that a chunk whose deadline is going to expire cannot be downloaded by a given node because none of its neighbors have that chunk (content bottleneck) or the upload bandwidth of the neighborhood is insufficient to transmit the data in time (bandwidth bottleneck). In these cases, the chunk would be lost, and, consequently, the QoE would be impaired. Unfortunately, such issues cannot be resolved only by increasing the degree of cooperation of the overlay network (i.e., by allowing each peer to establish direct links with many other peers). In fact, this does not represent a viable solution because of the overhead required to handle the high number of connections in each neighborhood.

2.3 Background on NDN

The ICN architecture conceived within the NDN project [8], known with the name Content-Centric Networking (CCN), is based on the "data-centric" approach: all contents are identified by a unique name, allowing users to retrieve information without having any awareness about the physical location of servers (e.g., IP address) [33,34]. In NDN, each content is uniquely identified by the *Content Name*, which is composed by multiple human-readable components (e.g., strings), optionally encrypted, arranged in a hierarchy (i.e., hierarchical naming scheme). This means that a *name tree* is introduced to identify all the available contents.

The communication in the NDN network architecture is handled through the exchange of only two kinds of messages: *Interest* and *Data*. The structure of both *Interest* and *Data* packets has been reported in Tabs. 1 and 2, respectively[1].

Table 1. Main fields of the *Interest* packet

Field	Description
Content Name	Specifies the requested item. It is formed by several components that identify a subtree in the name space.
Min Suffix Components	Enables the access to a specific collection of elements according to the prefix stored into the *Content Name* field.
Max Suffix Components	Enables the access to a specific collection of elements according to the prefix stored into the *Content Name* field.
Publisher Public Key Digest	Imposes that only a specific user can answer to the considered *Interest*.
Exclude	Defines a set of components forming the content name that should not appear in the response to the *Interest*.
Child Selector	In the presence of multiple answers, it expresses a preference for which of these should be returned.
Answer Origin Kind	Several bits that alter the usual response to *Interest* (i.e., answer can be "stale" or answer can be generated).
Scope	Limits where the *Interest* may be propagated.
Interest Life time	It indicates, approximatively, the time interval after which the *Interest* will be considered deprecated.
Nonce	A randomly genenerated byte string used for detecting duplicates.

An user may ask for a content by issuing an *Interest*, which is routed toward the nodes in posses of the required information (e.g., the permanent repository, namely *publisher*, or any other node that contains a valid copy in its cache), thus, triggering them to reply with *Data* packets. Routing operations are executed by

[1] It has been taken from the official software implementation of the NDN communication protocol suite, i.e., CCNx, available at `http://www.ccnx.org`.

Table 2. Main fields of the *Data* packet

Field	Description
Content Name	Identifies the requested item.
Signature	Guarantees the publisher authentication.
Publisher Public Key Digest	Identifies the users that generated data.
Time Stamp	Expresses the generation time of the content.
Type	Explicits what the *Data* packet contains (i.e., data, encrypted data, public key, link, and NACK with no content).
Freshness Seconds	Defines the life time of the carried payload. It is used to schedule caching operations (eventually prohibited).
Final Block ID	Indicates the identifier of the final block in a sequence of fragments.
Key Locator	Specifies where to find the key to verify this content.
Content	It is the content with an arbitrary length.

the strategy layer only for *Interest* packets. Whereas, *Data* messages, just follow the reverse path toward requesting user, allowing every intermediate node to cache the forwarded content.

To accomplish these activities, each NDN node exploits three main data structures: (i) the Content Store (CS), which is a cache memory, (ii) the Forwarding Information Base (FIB), containing list of *faces* through which forwarding *Interest* packets asking for specific contents, and the (iii) the Pending Interest Table (PIT), which is used to keep track of the *Interest* packets that have been forwarded upstream toward content sources, combining them with the respective arrival faces, thus, allowing the properly delivery of backward *Data* packets sent in response to *Interests*.

Basically, when an *Interest* packet arrives to an NDN node, the CS is in charge of discovering whether a data item is already available or not. If so, the node may generate an answer (i.e., a *Data* packet) and send it immediately back to the requesting user. Otherwise, the PIT is consulted to retrieve if others *Interest* packets, requiring the same content, have been already forwarded toward potential sources of the required data. In this case, the *Interest*'s arrival face is added to the PIT entry. Else, the FIB is examined to search a matching entry, indicating the list of faces through which forwarding the *Interest*. At the end, if there is not any FIB entry, the *Interest* is discarded. On the other hand, when a *Data* packet is received, the PIT table comes into play. It keeps track of all previously forwarded Interest packets and allows the establishment of a backward path to the node that requested the data.

3 State of the Art, Literature Review

3.1 The Diffusion of Real-Time Streaming Services in Social and Crowdsourcing Platforms

Some significant works focusing on streaming services in social networks and *crowd-sourced* platforms are described herein:

- **Incentive cooperation strategies for P2P live multimedia streaming social networks [35].** This paper focuses on live multimedia streaming services in P2P networks under limited bandwidth conditions. In particular, it proposes a framework based on game theory to model user behavior in a P2P network and designs incentive-based strategies to stimulate user cooperation in live streaming services.
- **Nettube: Exploring social networks for P2P short video sharing [36].** In this contribution, the adoption of clustering techniques is exploited to realize a P2P version of YouTube. According to the presented solution, users are grouped in clusters according to their interests. In each cluster, the designed solution tries to store locally most requested videos, thus accelerating their downloading time. On the other hand, those videos that are unavailable in the cluster can be still recovered from the remote site. The conducted analysis demonstrates that this approach outperforms other classical approaches, especially in the presence of short videos.
- **AMES-Cloud: A Framework for mobile social video streaming and sharing [37].** The authors realized a two part framework addressing the bandwidth limitations that can be found in mobile networks during the execution of video streaming services. The first part is a software agent that adapts, in real-time, encoding parameters based on currently available bandwidth. Its main goal is to dynamically avoid the loss of frames when the bandwidth is more limited and, at the same time, to properly increase the video bitrate when bandwidth availability goes up again due to better network conditions. The second part of the framework, instead, uses users' preferences matured in the social community to prefetch contents that have been recently viewed and rated.
- **Peer-Assisted social media streaming with social reciprocity [38].** This contribution tries to improve media services in P2P networks through the implementation of a credit system based on the user social relationships. To this end, it integrates game theory strategies and social group interactions.
- **Quantification of YouTube QoE via crowdsourcing [39].** This work focuses on the problem of evaluating QoE of YouTube videos by crowdsourcing it. In that procedure, votes of users are adopted to generate the rating.
- **Recursive fact-finding: truth estimation in crowdsourcing [40].** The authors propose a recursive algorithm for user-submitted-facts verification in crowdsourcing, in the absence of a reputation system, based on other observations previously made by the same user. The approach is especially valid in

real-case scenarios as it works also with continuous, and thus ever-changing, streams of new data, compared to other algorithms only working on statical scenarios. While not directly related to video streaming, the authors points out that a useful use case could be validating crowdsourced user rating and tagging of video, a scenario where the speed and dynamical properties of the proposed algorithm, as well as the results quality, are very important.

- **A measurement study of a large-scale P2P IPTV system [41].** The performance evaluation of a large scale deployment of P2P IPTV, i.e., the PPLive system that is deployed and used mainly in China and Asia, has been presented in this work. The behavior of this network architecture has been also compared to a typical TV set utilization patterns. The conducted analysis demonstrated that (i) users belonging to the P2P network have similar viewing behaviors as regular TV users, (ii) during a downloading session, a peer exchanges video data dynamically with a large number of peers, (iii) just a small set of peers act as video proxy and significantly contribute to the uploading process, and (iv) users of P2P IPTV system still suffer from long start-up delays and playback lags, ranging from several seconds to a couple of minutes.
- **PRIME: Peer-to-peer receiver-driven mesh-based streaming [42].** This contribution presents *PRIME*, i.e., a scalable mesh-based P2P streaming mechanism for live content. The discussed solution exploits a receiver-driven approach to fetch contents of a video stream and adopts an innovative mechanism to minimize the bandwidth bottleneck among participating peers.
- **Improving Large-Scale P2P Streaming Systems [43].** This PhD thesis identified issues arising from the provisioning of media streaming services in P2P architectures and proposed some modifications to the normal behavior of these systems, which are able to resolve these problems, especially in mobile environments. The proposed solutions have been evaluated through real experiments conducted on BitTorrent and Spotify platforms [24].

3.2 Real-Time Streaming Services in NDN Networks

Recently, some research activities are focusing the attention to the management of multimedia applications in NDN [44][45]. The most interesting proposals related to real-time applications are presented in the sequel:

- **Voice Over Content-Centric Networks (VoCCN) [46].** This service architecture supports a voice communication between two users. It supposes that each user is identified by an unique $userId$ and that it must announce its availability for the voice service by registering itself to a specific *namespace*. Moreover, a deterministic algorithm is used to generate *names* for identifying both users and contents during the execution of the service. At the beginning, the *service rendezvous* procedure is exploited to configured the VoIP session through the SIP protocol. The user that wish to initialize a call (i.e., the caller), maps a SIP INVITE message to an *Interest* packet that will be

forwarded to the remote users (i.e., the callee). The *Content Name* of the first *Interest* packet is built by appending to the aforementioned *namespace* the content, optionally encrypted, of the SIP INVITE message. The callee will answer to this request by generating a *Data* packet containing the SIP response. From this moment on, the exchange of media contents is done using the RTP protocol. To each media chunk is assigned an unique sequence number and, in order to fetch quickly voice packets, the caller can generate and send multiple *Interest* packets at the same time. Every time a new *Data* packet is received, a new *Interest* is released, thereby restoring the total number of pending *Interests*.

- **Audio Conference Tool (ACT) [47].** It is a more complex architecture enabling audio conference services, which exploits the named data approach to discover ongoing conferences, as well as speakers in each conference, and to fetch voice data from individual speakers. A specific *namespace* and an algorithm to create *names* have been tailored to support all of these tasks. Before joining the conference, an user issues specific requests to collect information about the list of ongoing conferences and to known the list of speakers in a conference (i.e., the group of participants that produce voice data from which fetching *Data* packets). Similarly to VoCCN [46], each *Data* packet is identified by an unique segment number and an user could request multiple *Data* packets at the same time. Hence, whenever a new *Data* packet comes back, the user will issue a new *Interest*.
- **The MERTS platform [48].** It has been designed to handle at the same time real-time and non real-time flows in a NDN architecture. To this end, a new field in the *Interest* message, e.g., the Type of Service (TOS), has been introduced to differentiate real-time and nonreal-time traffics, thus allowing each NDN node to classify the type of service to which each packet belongs to. In addition, a flexible transport mode selection scheme has been devised to adapt the behavior of the NDN node according to the TOS associated to a specific packet. In order to serve real-time applications, the *one-request-n-packets* strategy is proposed, according to which the user issues a *Special Interest* (SI) asking for n consecutive *Data* packets. Such a request can be satisfied by more nodes inside the network that may store in their repository/cache one, more, or all requested chunks. When all the n chunks will be received by the user, a new SI is generated. To ensure that the end-to-end route will not be deleted after the reception of only a subset of chunks requested with the SI, the normal functionality of the PIT table is modified by imposing that the SI can be erased only after the expiration of its *life time*. Finally, with the aim of optimizing the memory utilization of the cache and improving performances of nonreal-time services, the caching of real-time contents is completely disabled.
- **Adaptive retransmission scheme for real-video streaming [49].** This work proposes a novel and efficient retransmission scheme of *Interest* packets, which has been conceived to reduce video packet losses. In particular, the retransmission of requests not yet been satisfied after a given timeout is introduced to offer a minimum level of reliability in NDN. From one side, a

lifetime estimation algorithm captures the RTT variation caused by the in-network caching and dynamically evaluates the value of the retransmission timeout. From another hand, the Explicit Congestion Notification (ECN) field is added to the *Data* packet for signaling the ongoing network congestion to the end user. This information will be used by the retransmission control scheme to differentiates channel errors from network congestion episodes. Hence, based on the reason of packet losses, this algorithm adaptively adjusts the retransmission window size, i.e., the number of total *Interest* that can be retransmitted by the client.

- **Time-based Interest protocol [50].** This proposal tries to avoid the waste of the uplink bandwidth due to the generation of multiple and contemporary *Interest*. To reach this goal, it introduces a new *Interest* packet, which is sent by the user in order to ask for a group of contents that are generated by the publisher during a specific time interval. During such time interval, all chunks generated by the remote server or transferred from other nodes can be delivered to the user. This novel scheme requires a modification of the normal behavior of a NDN router: the *Interest* packet should not be deleted from the PIT table until that its *life time* will expire.
- **The NDNVideo architecture [51].** It has been designed and implemented on top of CCNx, in order to offer both real-time and nonreal-time video streaming services. A first important issue addressed by the NDNVideo project is the design of the *namespace* that enables the publisher to uniquely identifying every chunk of the multimedia content and allows the consumer to easily seek a specific place in the stream. In particular, the *Content Name* is built in order to provide information about the video content, the encoding algorithm, and the sequence number associated to a given chunk. Moreover, to facilitate the seeking procedure, the user can specify in its first request a timecode that will be used by the server to select the most suitable *Data* packet within the video stream. Then, after the reception of the first *Data* packet, the user will ask for video data using consecutive segment numbers. To supports real-time streaming services, the client may issues multiple *Interest* packets at the same time. However, to avoid that it will fetch data too quickly and request segments that does not yet exist, the client estimates the generation rate of *Interest* packets by knowing the time at which previous data packet ware generated by the publisher (this information is stored in a specific field of the *Data* packet). If data are not received fast enough to playback the video at the correct rate, the client may skip to the most recent segment, thus continuing to see the video from there instead of pausing the playback. Finally, a low-pass filter similar to the one defined in the TCP protocol is adopted to adjusts the retransmission timeout based on previous RTT values.

4 Problems, Issues, and Challenges Identified

The present contribution focuses the attention on the design of a network architecture able to distribute, in real-time, users generated contents (e.g., multimedia

flows) within a online social community. The scenario considered in our work is composed by a group of users that captures and transmits multimedia contents to a second group of consumers diffused worldwide. It covers a number of significant use-cases, such as:

- real-time broadcasting of social events: users participating to a given event capture media contents from different point of views and share such media streams with remote clients;
- virtual tourism: users visiting monuments (or other kind of tourist attraction) capture media contents and share them with other ones;
- real-time broadcasting of activities and environmental conditions during dangerous situations.

In such a scenario, multimedia contents are not provided by a media server, but they are shared by a (potentially large) number of users. This scheme fully reflects the *crowd-sourcing* paradigm, presented in the Sec. 2.

In our opinion, the design of an network architecture able to efficiently distribute *crowd-sourced* multimedia contents within a social community spread around the world is flanked by the born of a number of issues and challenges that need to be carefully investigated.

4.1 Issues Arising from the Considered Scenario

First of all, when thinking at crowd-sourced media streaming, the issue of scalability is the first one comes into play. In a canonical media portal, where there is only one content generator, the content provider needs to maintain an infrastructure capable of serving contents to a vast number of consumers. Generally, a Content Delivery Network (CDN) architecture can be exploited to maintain multiple and parallel multimedia sessions and to efficiently handle the distribution of media contents. In a CDN, in fact, contents are replicated among many servers, which are strategically located in different places, thus allowing users to seamlessly connect to the closest server [52][53]. However, in this network architecture, consumers do not necessarily generate a huge upload traffic, being mostly responsible of posting text comments, ratings, and other operations less bandwidth consuming. In the case of crowd-sourced media services, where a group of users is in charge of producing and sharing multimedia contents, high computational and bandwidth capabilities are no more required for a static infrastructure of servers (like in the previous case), but they need to be distributed among the users themselves. Unfortunately, when people participate to an event, it is impossible to guarantee these requirements. From one side, in fact, users generate contents through devices with scarce computational capabilities (i.e., smartphones and tablets). On the other hand, instead, they could be connected to the network by means of a link with limited capacity. Based on these considerations, it is necessary to conceive a service architecture that should be able to reduce, as more as possible, the traffic load managed by content producers, while ensuring a good level of the quality of service to other users of the social community that want to download media streams.

Another important issue refers to the availability of contents. In these years, in fact, we are assisting to a radically change of the way digital resources are used: users are interested to share contents rather than to interact with remote devices [33]. This means that they need to fetch contents in a fast, reliable, and effective way, without knowing their location a priori. In other words, it is not more necessary to identify a media stream through the IP addresses of the device that provides it but using an unique *name*. Accordingly, the service platform should enable the classification, the retrieving, and the delivering of media contents, just exploiting their names.

Furthermore, the seamless support of mobile users, which should not experience any service interruption when moving across different access networks, represents another relevant aspect to consider. Finally, other minor aspects that deserve our attention are: the possibility to trust contents independently from the location and the identity of who is providing them and the capability to guarantee the resilience of ICT services to system failures.

4.2 Towards an ICN-Based Solution and Related Challenges

Nowadays, it is widely recognized that the current Internet architecture is not able to efficiently face all of the issues illustrated in the previous Sections [33], [54], [55].

Fortunately, the emerging ICN paradigm seems to have all the potentials to offer novel network architectures for the Future Internet, which will be able to overcome these problems [3]. In fact, among other novel and interesting features, the ICN approach provides (i) the addressing of contents through names, (ii) a faster content distribution due to the adoption of in-network caching strategies, (iii) the delivering of user demands through *routing-by-name* approaches, (iv) a simplified management of mobility, and (v) a native support for both the security and multicast services [55].

In line with this premise, we decided to base our proposal on an emerging NDN network paradigm, which is a promising ICN architecture presented in literature.

Obviously, the design of a NDN-based platform enabling crowd-sourced and real-time media services is not a simple task to accomplish because of the presence of several challenges, which include:

- *Namespace definition.* NDN architecture defines the resources exchanged in the systems via a URI notation. When designing a NDN-based application, the first step consists in defining what kind of data will be exchanged and how that data will be represented as a URI. In this regard, the design of the *name space* is a crucial activity that will impact on the performance of the entire system. In order to simplify and ameliorate routing operations and packet processing, it is necessary to conceive a hierarchical *name tree*, which will be able to efficiently map all the data that can be exchanged among nodes in the network.
- *Identification of events and media contents.* In the considered scenario it is important to devise a way to track and identify the list of active events and

the set of available multimedia contents (i.e., those captured by users of the social community that are participating to a given event). According to the NDN paradigm, this challenge has to be addressed through a completely host-less protocol, which should allow any user to download a media stream for a given event by adopting NDN primitives.

– *Management of content requests.* According to the NDN paradigm, audio, and video contents should be divided into a list of consecutive chunks that should be requested by the client during the service execution. Unlike Video-On-Demand, the distribution of a real-time stream has to deal with a specific class of problems to ensure the timely delivery of an ordered stream of chunks. Video and audio chunks, in details, have to be received in playing order and within a given time interval (the *playout delay*), before they are actually played, thus "expiring". A chunk not delivered before its expiration will result in degradation of the rendered video, impacting the end user QoE. To solve these challenges, client nodes implement a receiving buffer queue where the chunks are stored in order, that is emptied while the video is being played. Therefore, any chunk not received before its playing instant becomes useless. To reduce the chance of chunk loss, an efficient mechanism that control the retransmission of user's requests (e.g., requests for chunks close to expiration and not yet received), should be conceived.

We highlight that these challenges have been only partially investigated in literature and, at the same time, solutions dedicated to the envisaged scenario have never been discussed.

5 The Proposed NDN-Based Architecture

The main goal of the service architecture described in this chapter is to discover, organize, store, process, and deliver media contents, which are captured according to the *crowd-sourcing* paradigm, within a social community spread around the world. Such an architecture has been conceived as an extension of our previous contributions discussed in [56] and [57].

As shown in Fig. 1, the service platform is composed by:

1. a community of users that, being into the same place to take part to an event (i.e., concert, football match, sightseeing, city accident, and so on) record and broadcast it from their multiple points of view;
2. a number of users interested to see what the aforementioned social community is capturing;
3. a distributed *Event Management System*, which creates events and handles the social community;
4. a NDN-based communication infrastructure able to efficiently manage user's requests and distribute multimedia contents.

The *Event Management System* covers a crucial role because it is used to (i) track active events, (ii) register users that would work as publishers in the

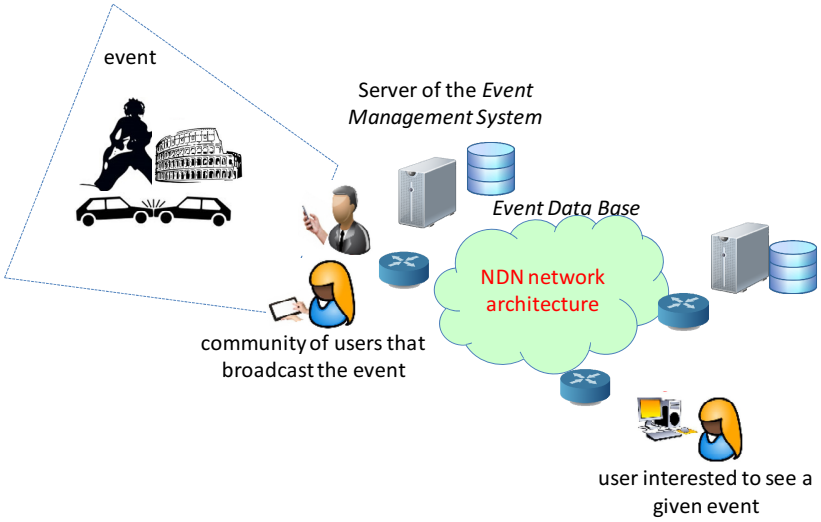


Fig. 1. Network architecture of the conceived platform

considered architecture, and (iii) provide information about available events to other users that request them. It may be composed by a number of dedicated servers, possibly spreaded over a cloud architecture, distributed within the NDN infrastructure. Each node of the *Event Management System* uses a *Event Data Base* to store details about active events, such as the name, the location, the duration. Moreover, the list of potential publishers is saved too. For each of them, the database will contain some parameters that are typically shared within a social community (like nickname, age, location, work, and so on), as well as the *media capabilities* specifying the type of device, the encoding algorithms, the resolution, and the encoding bit rate adopted by the user to generate video and voice data. Therefore, the user that would transmit its media stream, i.e., the *publisher*, has to register itself to the *Event Management System*. On the other hand, an user that wishes to see and listen these contents, i.e., the *client*, should retrieve all the information related to a specific event by interrogating the *Event Management System* through NDN primitives. To this end, the devised platform addresses four different tasks: *event announcement*, *event discovering*, *media discovering*, and *media delivering*. Moreover, to efficiently support these operations, a *name space* has been properly designed to classify and map all the services activities in an ordered and hierarchical manner, thus, guaranteeing enormous advantages for routing operations and packet processing.

5.1 Design of the *Name Space*

The definition of the *name tree* is a key aspect of a NDN network because the *Content Name* field strongly influences the way *Interest* and *Data* packets are treated.

In line with theoretical suggestions presented in [46], the *name space* we conceived herein is based on both *contractable names* and *on-demand publishing* concepts. The *contractable names* approach identifies the ability of an user to construct the name of a desired content through specific and well-known algorithms (e.g., it knows the structure of the name tree and the value that each field of the *Content Name* may assume). The *on-demand publishing* criteria, instead, defines the possibility to request a content that has not yet been published in the past, but it has to be created in answer to the received request (this may occur very often during a real-time communication). The *name tree* structure adopted in our platform is:

/domain/ndn_streaming/activity/details

Similarly to [58], starting from the root tree, which is identified with "/", we introduced the *domain* field in order to explicitly indicate the domain in which the service is offered. The second field adopts the keyword *ndn_streaming* to explicit the considered service (i.e., the streaming of a video content over the conceived NDN-based platform). The *activity* field specifies the task to which the *Interest* or *Data* packet belongs to. As anticipated before, it may assume four different values. i.e., *event_announcement*, *event_discovering*, *media_discovering*, and *media_delivering*. Finally, the latest field, i.e., *details*, is used for appending to the *Content Name* specific values that can be exchanged among nodes during the service execution.

5.2 Event Announcement

An user that wants to provide multimedia contents for a specific event should register itself and announce its *media capabilities* to the *Event Management System*. To this end, it firstly sends a *Interest* packet asking for the list of events active within a given location (i.e., the geographical area in which the user is located). The *Content Name* of this request is set to:

/domain/ndn_streaming/event_announcement/event-list/location

This message will reach the closest node (i.e., a server of the *Event Management System* or a NDN router storing this data within its cache) able to provide these information, which will answer with the corresponding *Data* packet.

Then, the user will generate a new *Interest* packet, whose *Content Name* will contain, in the *details* field, its position and its *media capabilities*, as reported in the sequel:

/domain/ndn_streaming/event_announcement/

registration/event/position/media_capabilities

This message will be processed by the *Event Management System* that will update the *Event Data Base* and will answer with a *Data* packet of confirmation.

5.3 Event and Media Discovering

In order to discover events in a given geographical area, an *Interest* packet is released with a *Content Name* set to:

/domain/ndn_streaming/event_discovering/location

This message will be routed toward the first node in posses of this information, which will respond with a *Data* packet containing the requested information.

Once the user has identified the event of its interest, he will retrieve the list of available multimedia contents, together with their characteristics, by sending an *Interest* packet with the *Content Name* equal to:

/domain/ndn_streaming/media_discovering/event

To ensure that this request will be handled by a node of the *Event Management System*, which is the only device storing updated information, the *PublisherPublicKeyDigest* and *AnswerOriginKind* fields of the *Interest* packet contain the hash function of the public key of the *Event Management System* and the numerical value 3, respectively. The corresponding *Data* packet, generated in answer to this request, will allow the user to select its preferred content among those available. From this moment on, the user can start fetching the multimedia content from a specific source.

With the aim of designing a more flexible platform as possible, we assume also that the user can change the source of the media stream during the time. To better support this feature, it is necessary to send periodically the aforementioned *Interest* packet, thus continuously updating information about available multimedia contents.

5.4 Media Delivering

The media delivering process consists of a channel bootstrap phase, a flow control strategy, and an efficient mechanism for retransmitting *Interest* packets. It is performed after that the user has selected the event of its interest, i.e., the *event_name*, and the media source, identified through its social nick name (i.e., *source_nick_name*), from which fetching media data. To enable these functionalities we need to extend the basic structure of the *Interest* packet by introducing an additional *Status* field marking if the *Interest* is related to the channel bootstrap phase or to a retransmission.

As detailed in [56] and [57], the bootstrap phase is in charge of retrieving the first valid chunkID of the video stream, which is the latest generated I-frame. For this reason, it sends an *Interest* packet in which a timecode (e.g., HH:MM:SS:FF) is appended to the *Content Name* (this approach has been already introduced in [51]):

/domain/ndn_streaming/media_delivering/

event_name/source_nick_name/HH:MM:SS:FF

The *Status* field set to *BOOTSTRAP* and the *Nonce* field set by the client. An *Interest* with *Status* = *BOOTSTRAP* would travel unblocked until it reaches the first good stream repository (i.e., a node who can provide a continuous real-time flow of chunks, not just cached ones). As soon the node receives the bootstrap data message, it can initiate the sliding window mechanism to request the subsequent chunks.

Each chunks of the video stream is identified with an unique $chunkID$ and can be retrieved by issuing an *Interest* packet having the *Content Name* set to:

/domain/ndn_streaming/media_delivering/event/source_nick_name/chunkID

Each node has a windows to store W pending chunks. We define *pending chunk* a chunk whose *Interest* has been sent by the node, and the window containing the pending chunks a *Pending Window*. Together with the chunkID, we store in the pending window other information, such as the timestamp of the first request and the timestamp of the last retransmission. Whenever a new data message is received, the algorithm described in Fig. 2 runs over the Pending Window, to execute the following operations:

1. purge the Pending Window from all the chunks who are expired, i.e., who have already been played, to free new space in the sliding window;
2. retransmit all chunks that have not been received for a given timeout (onward denoted as *windowTimeout*;
3. transmit, for each slot that got freed by the received or expired chunks, the *Interest* for a new one.

Furthermore, the same operations are performed if a node does not receive any data for at least *windowTimeout* seconds, in which case, all the *Interests* for not expired chunks in the Pending Window are retransmitted, together with new chunks if new slots have been freed due to expired chunks.

Fig. 2 details the implemented algorithm; for the purpose of brevity and readability, the variable names have been contracted: PW is the Pending Window, W is the aforementioned system parameter, indicating how many *Interests* should a node have ongoing, $WinT$ is the window timeout, after which *Interests* in the Pending Window are resent, Int is a new *Interest* message, CID is a $chunkID$ in the pending window, $lastTx$ is the transmission time of the most recent $Interest$ for a given $chunkID$, LC is the $chunkID$ of the most recent requested chunk and NNC is the number of new chunks to request, after the pending window has been purged. Moreover, to provide a further insight, we reported in Fig. 3 an example of the conceived sliding window algorithm, in which we have set the value of W to be equal to 3.

As described in Sec. 2, NDN nodes along the routing path of an *Interest* will stop its propagation if they have previously routed another *Interest* for the same resource, and the correspondent data has not been sent back yet (in that case, they will simply update their Pending Interest Table adding the face from where this newcomer *Interest* was originated, so to reroute the data back recursively along the path the *Interest* has gone through). However, to enable retransmission

```
1: procedure SendInterests(PW, W, WinT, Now, LC)
2:     # Remove all expired Interest
3:     for all CID in PW do
4:         if CID is expired then
5:             remove CID from PW
6:         end if
7:     end for
8:     # Resend stale Interests
9:     for all CID in PW do
10:        if lastTx(CID) < Now − WinT then
11:            resend(Int(CID))
12:            lastTx(CID) ← Now
13:        end if
14:    end for
15:    # Send Interests for new chunk
16:    NNC ← W − size(PW)
17:    for i ← 1, NNC do
18:        send(Int(LC))
19:        lastTx(LC) ← Now
20:        add LC to PW
21:        LC ← LC + 1
22:    end for
23: end procedure
```

Fig. 2. Sliding window algorithm

procedures, a retransmitted *Interest* must be propagated all the way up to the remote publisher, or to the first node with the desired chunk in cache. Therefore, retransmitted *Interests* carry the *Status* field set to *Retransmission* to mark if the *Interest* is a retransmission or not, and each node along the routing path propagates the *Interests* marked as retransmitted, thus skipping the usual NDN mechanism, unless the correspondent chunk is found in the cache.

6 Evaluation of the Proposed Approach

To evaluate the performances of the conceived solution under different network configurations, we used *ccnSim*, i.e., an open source and scalable chunk-level simulator of NDN [11] built on top of the Omnet++ framework [59].

6.1 Simulation Platform

By itself, *ccnSim* models a complete data distribution systems, with a high degree of fidelity concerning catalogs, requests and repositories distribution, and network topologies. Unfortunately, in its original version, *ccnSim* does not support real-time video transmissions. Hence, to evaluate the performances of the proposed platform, we extended the simulator in several aspects, by adding:

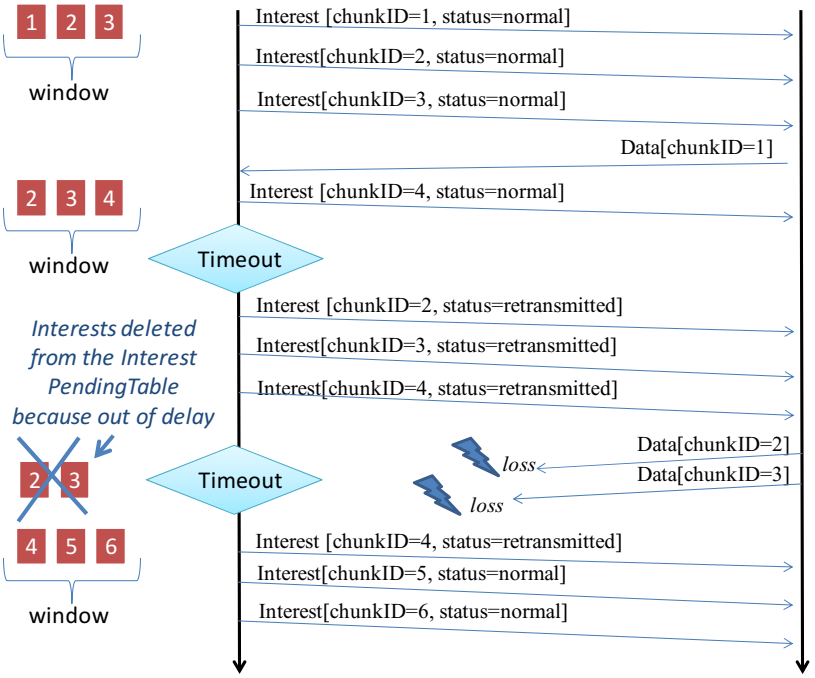


Fig. 3. Sliding window example

- a support for links with bounded capacity and packets with a well-defined size, which was missing in *ccnSim*, to be able and estimate the NDN behavior under some bandwidth constraints;
- a transmission queue for each face of each node, in order to properly manage the packet transmission under constraints due to the data-rate of channels,
- the support for synthetic video traces, so to be able to transmit and receive chunk of real videos, and consequently being able to reconstruct the received video and evaluate its Peak Signal-to-Noise Ratio (PSNR);
- a cleanup mechanism for each node's PIT, to avoid having long-term stale entries due to expired chunks;
- an improved logging system, so to be able to record each node's received chunks and reconstruct the received video;
- more controls server-side, to send a data only for those chunks who have already been generated;
- the sliding window mechanism described above, and all the related data structures;
- the *Interest* forceful propagation in case of retransmission.

6.2 Network Configuration and Parameters

We considered a complex network architecture composed by 68 routers connected among them according to the Deutsche Telekom topology. Furthermore,

it is assumed the presence of only one event where participate a number of users that produce, in real-time, video streams with different encoding characteristics. In particular, without loss of generality, in every simulation round, each video content, is mapped to a video stream compressed using H.264 [60] at a average coding rate randomly chosen in the range $250 \div 2000$ kbps. We suppose also that these users are connected to the network through the same access point, which is identified by one router of the aforementioned topology, randomly chosen every run among available ones. On the other hand, clients of the social community, that are interested in downloading video contents generated by the previous group of users, are connected to remaining nodes (1 client per node). Further, the selection of a media stream has been modeled considering that contents popularities follow a Zipf distribution, which is commonly adopted for user generate contents [61].

In our study, we adopted the optimal routing strategy, already available within the *ccnSim* framework. According to it, *Interest* packets are routed toward router to which are attached users that generate video data along the shortest path. On the other hand, three caching strategies have been considered in our study: *no-cache*, *LRU*, and *FIFO* [62]. When well known *LRU* or *FIFO* policies are adopted, we set the size of the cache to 10000 chunks. The *no-cache* policy is intended to evaluate the performance of the NDN without using any caching mechanism. Furthermore, a *baseline scenario*, in which the *no-cache* policy is enabled and the PIT table is totally disabled (this means that each user establishes with the service provider a unicast communication and the server should generate a dedicated Data packet for each generated Interest), has been considered as a reference configuration.

Once a client selects the video content of its interest, it performs the bootstrap process described in the previous section and then starts sending *Interest* packets following the designed sliding window mechanism. The window size W has been set to 10, ensuring that faces of the server are almost fully loaded in all considered scenarios. Also, the transmission queue length associated to each face, Q, has been set, in order to be larger than

$$Q = 2 \cdot L_c \cdot P_D,$$

where L_c and P_D represent link capacity and maximum propagation delay in the considered network topology.

We evaluated the designed NDN-based service architecture by varying the amount of the network bandwidth dedicated to real-time streaming services (set in the range $50 \div 100$ Mbps), the *playout delay* (chosen in the range $10 \div 20$ s), the *windowTimeout* (chosen in the range $1/10 \div 1/2$ of the *playout delay*), the number of users participating to the event (set in the range $10 \div 100$), and the popularity of available contents (in line with [62], α has been chosen in the range $1 \div 1.5$). In addition, each simulation lasts 600s and all results have been averaged over 15 simulations. To conclude, all simulation parameters have been summarized in Tab. 3.

Table 3. Summary of simulation parameters

Parameter	Value
Topology	Deutsche Telekom with 68 routers
Link capacity	50 Mbps and 100 Mbps
Number of active events	1
Number of user generate contents	10, 20, 50, 100
Number of clients	67
Chunk size	10Kbytes
Video average bit rate	250kbps, 600kbps, 1000kbps, and 2000kbps
W (window size)	10
Playout delay	10 s and 20 s
Window timeout	1/10, 1/5, and 1/2 of the playout delay
Protocol configuration	No cache, LRU, FIFO, *baseline scenario*
Cache size	10000 chunks
Simulation time	600 s
Number of seeds	15

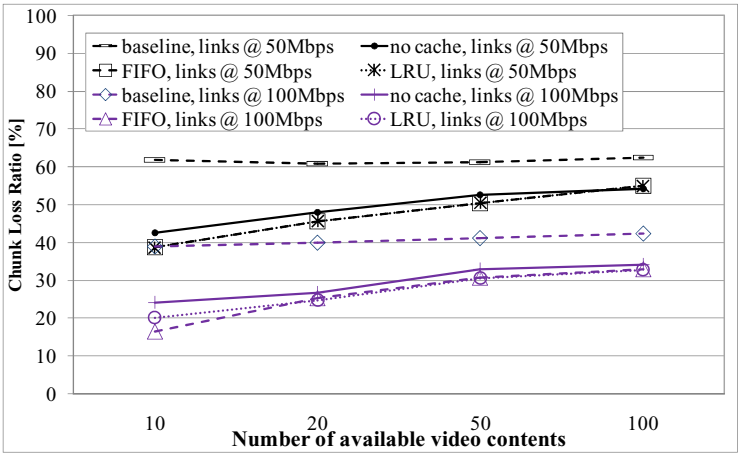
6.3 Performance Evaluation

The first important performance metric that describes how the parameter settings affect the Quality of Service (QoS) offered to end users is the chunk loss ratio, which represents the percentage of chunks that have not been received in time (i.e., before the expiration of the *playout delay*) by clients.

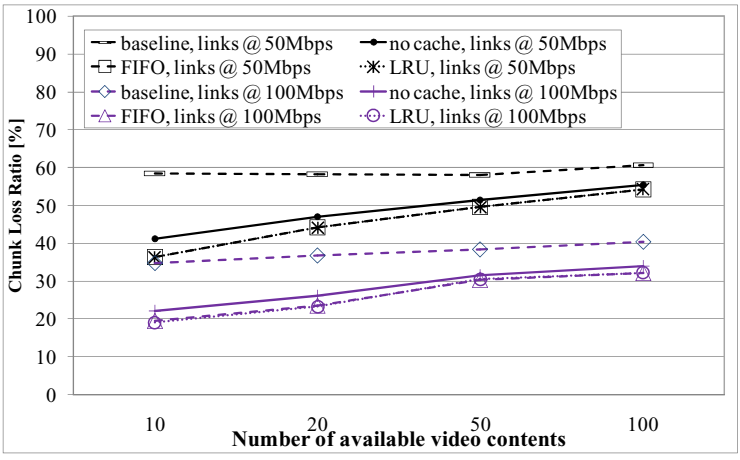
Figs. 4-7 show the chunk loss ratio measured in all considered network scenarios, when α, i.e., the parameter of the Zipf's law, has been set to 1 and 1.5, respectively. From reported results, several conclusions can be drawn.

Fist of all, it emerges that a reduction of the link capacities leads to a higher number of lost chunks, due to increased latencies introduced by network congestion. Moreover, we note that the *playout delay* plays a fundamental role. When the *playout delay* increases, in fact, the client could receive a *Data* packet within a longer time interval, thus reducing the amount of chunks discarded because out of delay. On the other hand, a slight increment of the chunk loss ratio can be registered by increasing the *windowTimeout*. If the client retransmits an *Interest* packet after long time, there is the risk that the *Data* packet will be reached by the destination after the expiration of the *playout delay*. This result is more evident when the *playout delay* is set to 10s, which defines more strict constraints on the packet's delivery.

It is very important to remark that the number of users that generate multimedia contents influences significantly the performance of the proposed architecture. In particular, we found that the higher is the amount of available video contents, the higher is the registered chunk loss ratio. The reason is that when the number of available video increases, users may request different contents, thus growing the traffic load generated in the network. Obviously, the content popularity plays a crucial role in this case: worst performances are observed in scenario when a group of contents have the highest popularity (i.e., $\alpha = 1$).



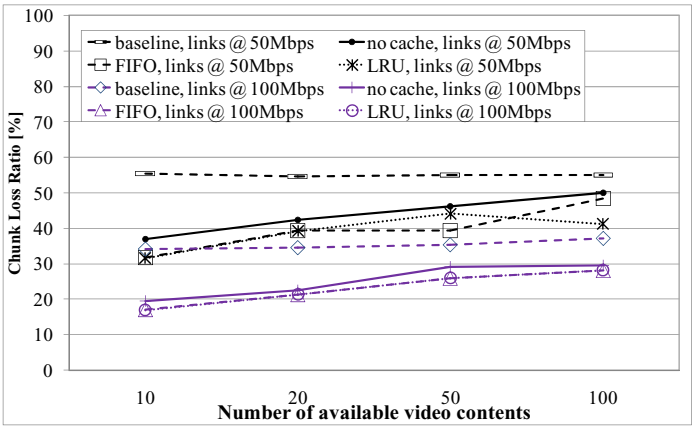
(a)



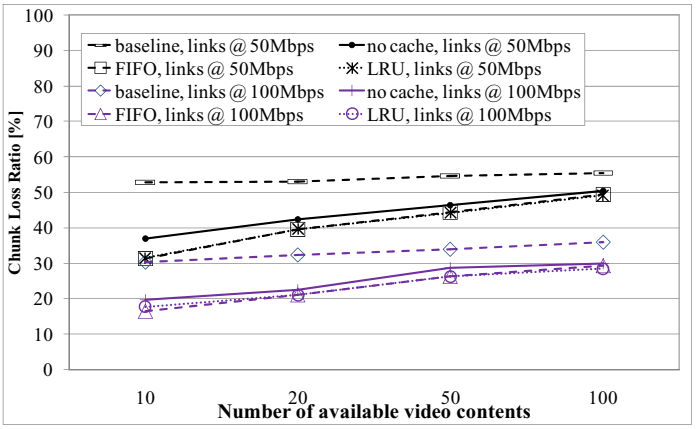
(b)

Fig. 4. Chunk loss ratio (scenario with $\alpha = 1$) when the PD is set to 10s and the $winT$ is equal to (a) 1/10 PD and (b) and 1/2 PD, respectively

By handling unicast communications, the *baseline scenario* generates the highest network congestion level, thus registering the worst performances. Thanks to the adoption of caching policies and the PIT table NDN provides a native support for multicast communication. In the considered scenario, where some users may request the same video content at the same time, this feature notably improves final performances. However, in our tests we realized that the presence of the cache can guarantee only a small reduction of the chunk loss ratio. In the presence of real-time flows, in fact, the cache does not represent an important NDN feature. On the other hand, we noticed that the PIT plays a more relevant



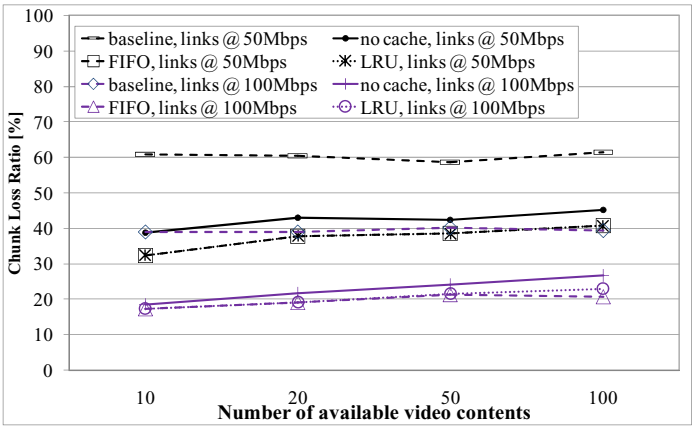
(a)



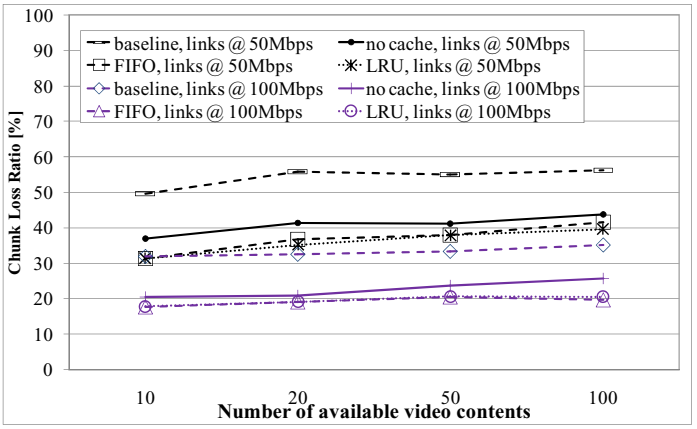
(b)

Fig. 5. Chunk loss ratio (scenario with $\alpha = 1$) when the PD is set to 20s and the $winT$ is equal to (a) 1/10 PD and (b) and 1/2 PD, respectively

role. In presence of live video streaming services, clients that are connected to a channel request the same chunks simultaneously. In this case, a NDN router has to handle multiple *Interest* messages that, even though sent by different users, are related to the same content. According to the NDN paradigm, such a node will store all of these requests into the PIT, waiting for the corresponding *Data* packet. As soon as the packet is received, the router will forward it to all users that have requested the chunk in the past. According to these considerations, the use of the cache will not produce a relevant gain of network performances. Indeed, the PIT helps reducing the burden at the publisher side by avoiding that many *Interest* packets for the same chunk are routed to the server.



(a)

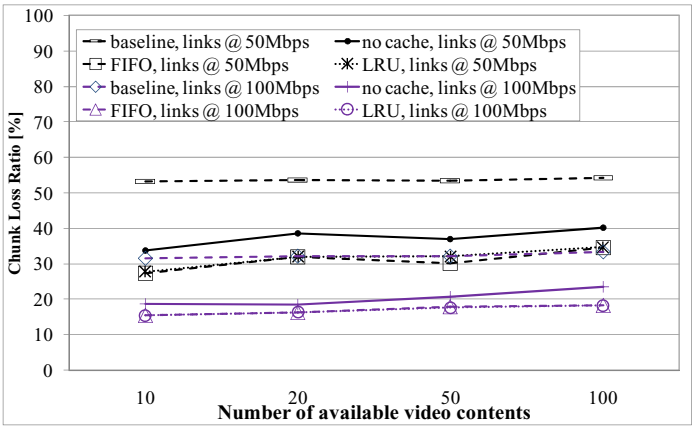


(b)

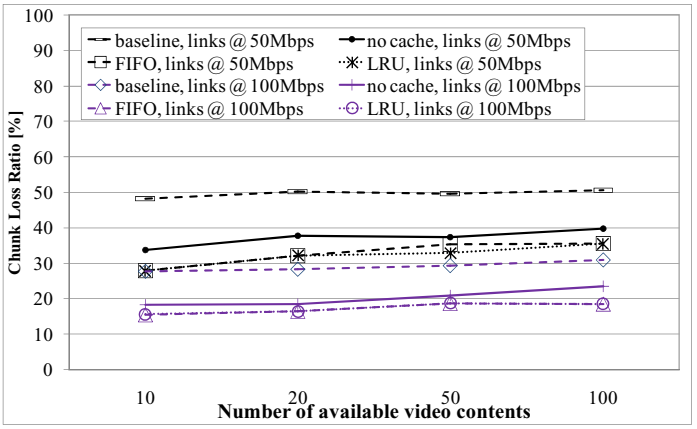
Fig. 6. Chunk loss ratio (scenario with $\alpha = 1.5$) when the PD is set to 10s and the $winT$ is equal to (a) 1/10 PD and (b) and 1/2 PD, respectively

To provide a further insight on this aspect, we also reported in Figs. 8-11 the percentage of *Interest* packets sent by users and directly received by the publishers, when α has been set to 1 and 1.5, respectively.

In the *baseline scenario*, the total amount of generated *Interests* reach the remote server, thus excessively overloading its faces. By enabling the PIT table, even without implementing any caching mechanism, the system is able to halve the traffic load at the server side, thus improving significantly network performances. With this very important finding, we demonstrate how the adoption of a NDN-based network infrastructure represents a winning solution to efficiently distribute *crowd-sourced* contents within a social network.



(a)

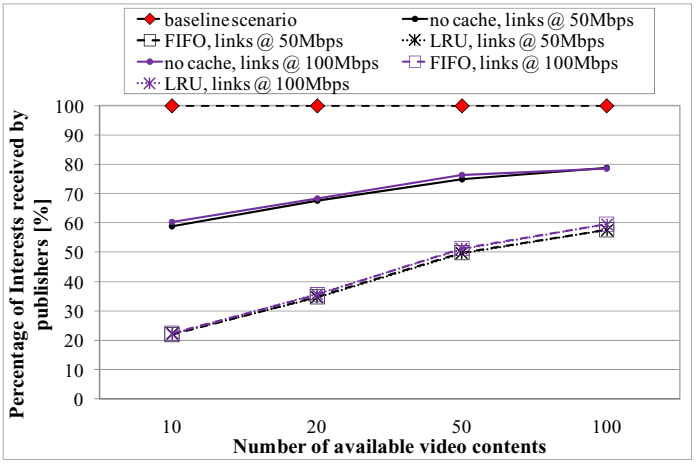


(b)

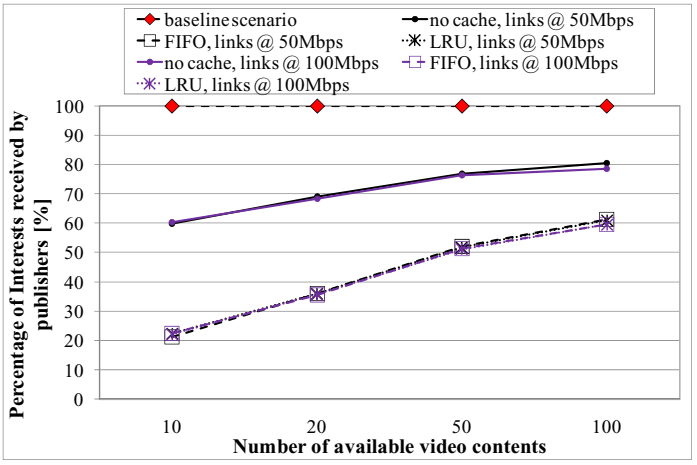
Fig. 7. Chunk loss ratio (scenario with $\alpha = 1.5$) when the PD is set to 20s and the $winT$ is equal to (a) 1/10 PD and (b) and 1/2 PD, respectively

Another important finding is related to the impact that the content's popularity has on the amount of requests reached by publishers. From Figs. 8-11, in fact, it is possible to observe that the higher is the α value, the lower is the percentage of requests that reach the publishers. The reason is that when α increases, the probability that an user of the social community is interested to one of the most popular contents increases as well, thus amplifying the capability of the PIT table of blocking the propagation of multiple requests asking for the same data packet.

To conclude our study, we have computed the PSNR, which is nowadays one of the most diffused metrics for evaluating user satisfaction, i.e., the QoE, together with interactivity level, in real-time video applications [63]. Results shown in



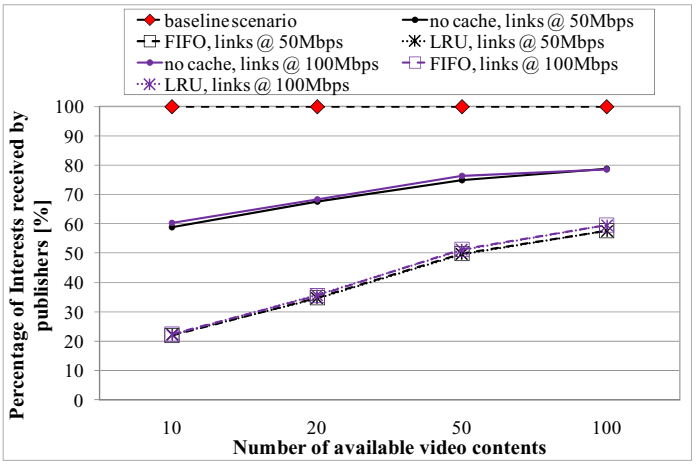
(a)



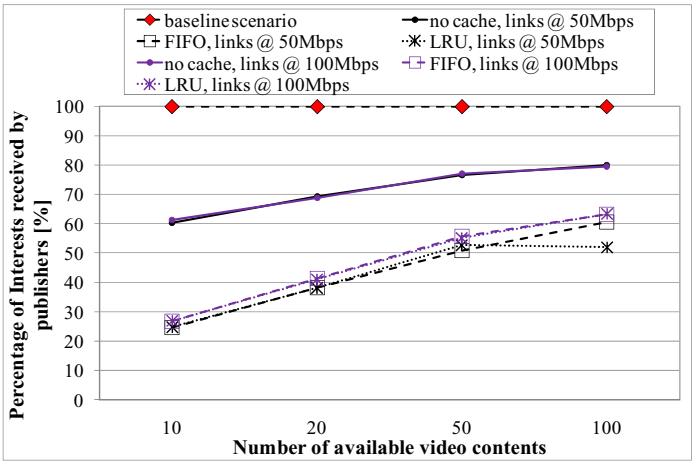
(b)

Fig. 8. Percentage of interests received by publishers (scenario with $\alpha = 1$) when the PD is set to 10s and the $winT$ is equal to (a) 1/10 PD and (b) 1/2 PD, respectively

Figs. 12-15, which reports the PSNR computed when α has been set to 1 and 1.5, respectively, are in line with those reported for chunk loss ratio: the PNSR is higher in the same case in which the chunk loss ratio is lower. Hence, also in this case we can verify that when both the link capacities and the playout delay increase, the total amount of chunks received by end users increases, thus obtaining an higher satisfaction level. In addition, while with the absence of the cache is registered a limited worsening of the PSNR, the *baseline scenario* reaches always the lowest performances.

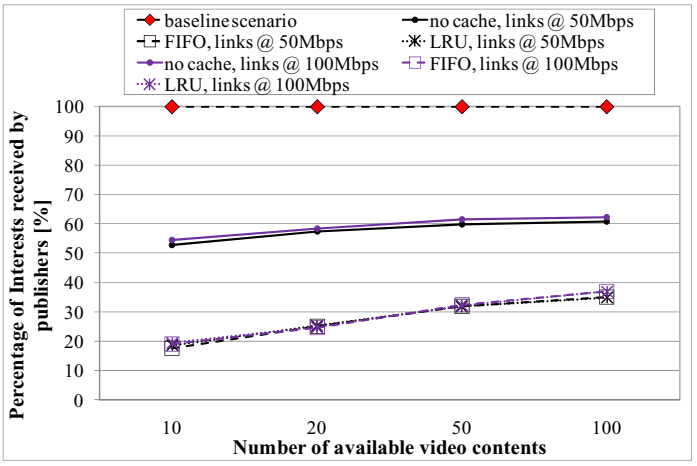


(a)

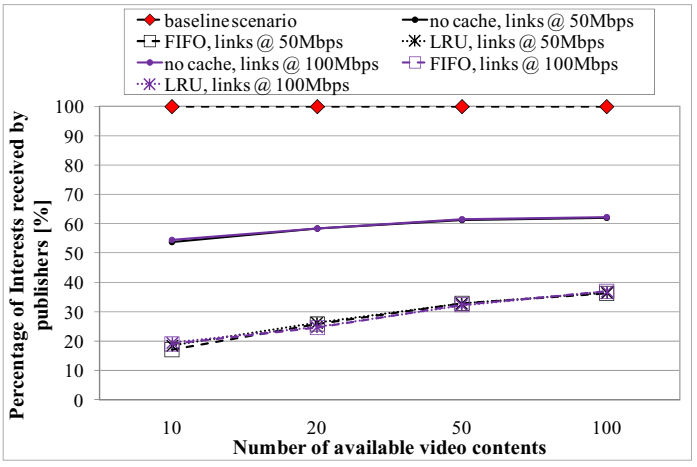


(b)

Fig. 9. Percentage of interests received by publishers (scenario with $\alpha = 1$) when the PD is set to 20s and the $winT$ is equal to (a) 1/10 PD and (b) 1/2 PD, respectively

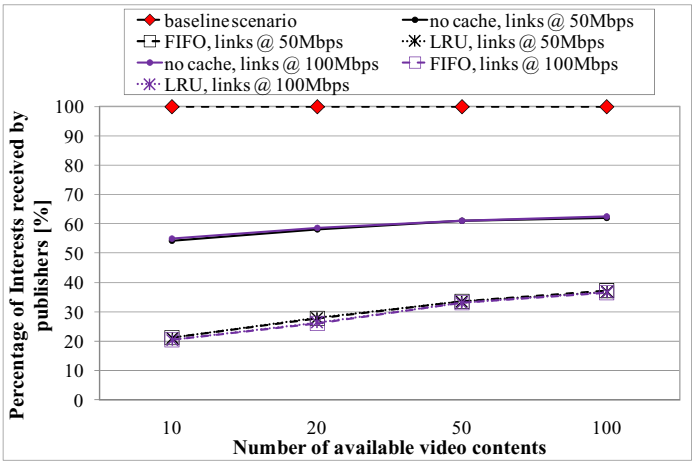


(a)

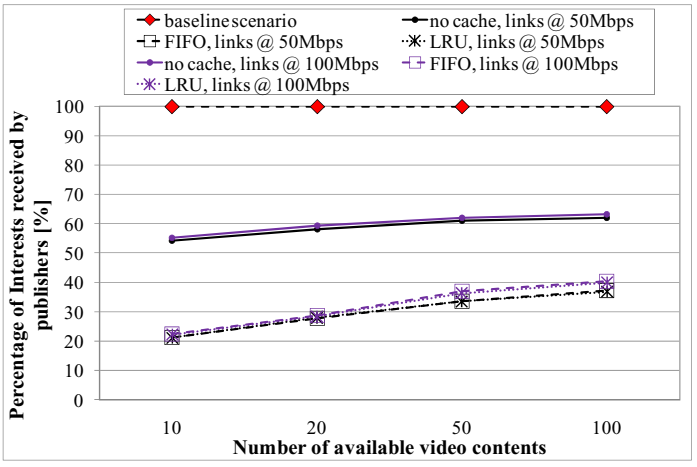


(b)

Fig. 10. Percentage of interests received by publishers (scenario with $\alpha = 1.5$) when the PD is set to 10s and the $winT$ is equal to (a) 1/10 PD and (b) 1/2 PD, respectively

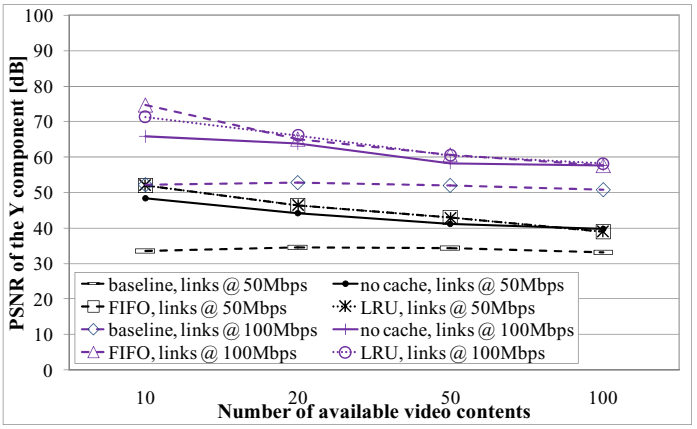


(a)

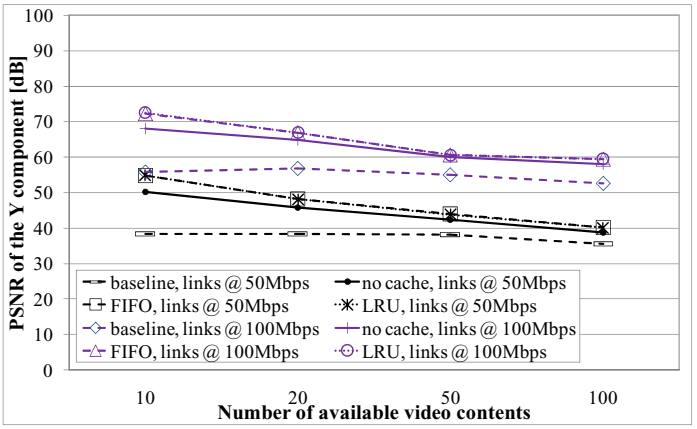


(b)

Fig. 11. Percentage of interests received by publishers (scenario with $\alpha = 1.5$) when the PD is set to 20s and the $winT$ is equal to (a) 1/10 PD and (b) 1/2 PD, respectively

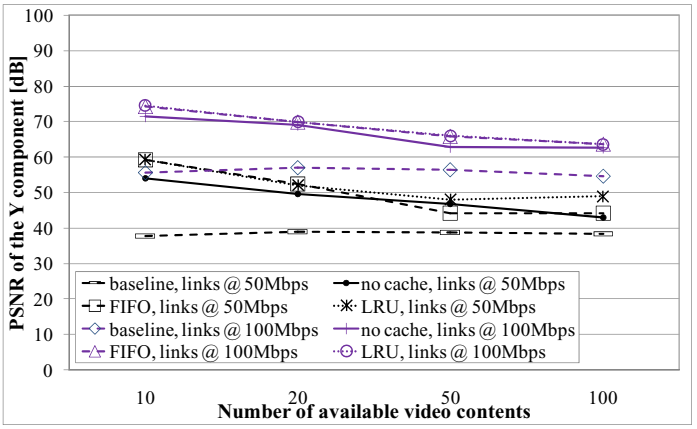


(a)

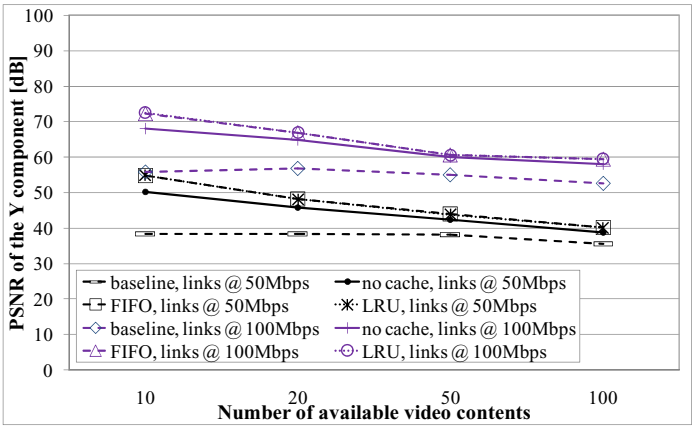


(b)

Fig. 12. PSNR of the Y component (scenario with $\alpha = 1$) when the PD is set to 10s and the $winT$ is equal to (a) 1/10 PD and (b) 1/2 PD, respectively

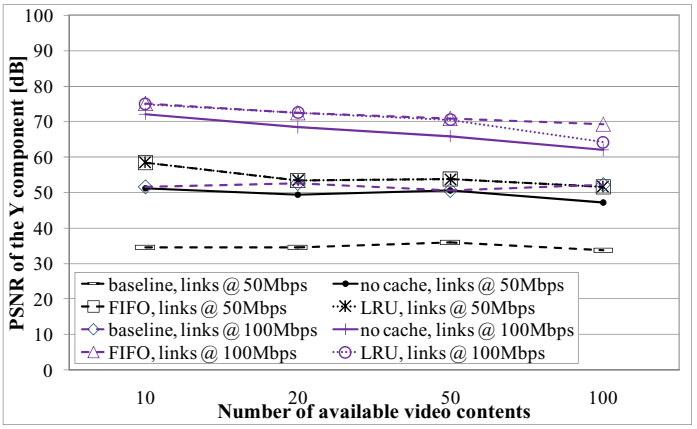


(a)

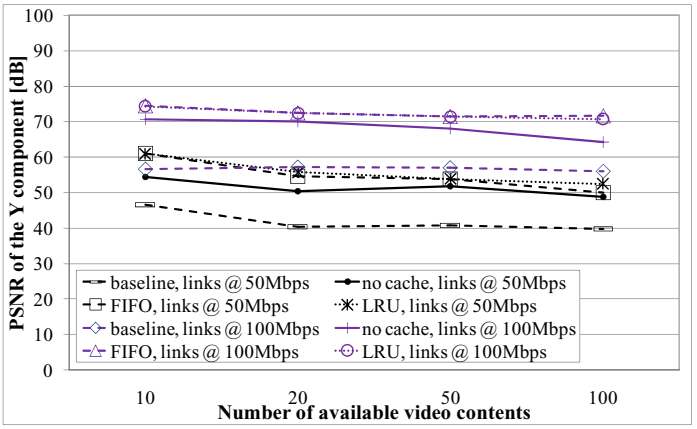


(b)

Fig. 13. PSNR of the Y component (scenario with $\alpha = 1$) when the PD is set to 20s and the $winT$ is equal to (a) 1/10 PD and (b) 1/2 PD, respectively

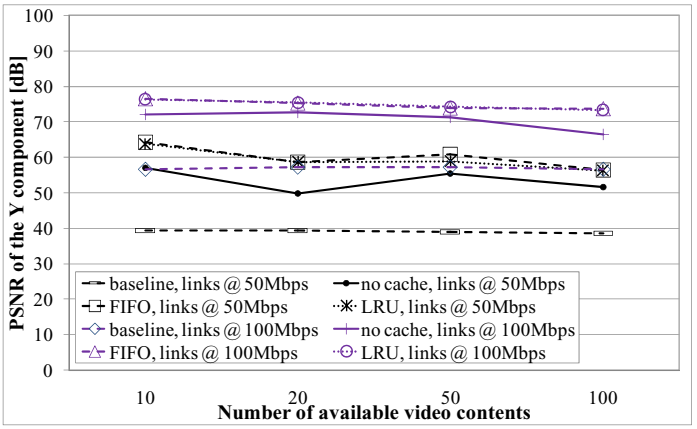


(a)

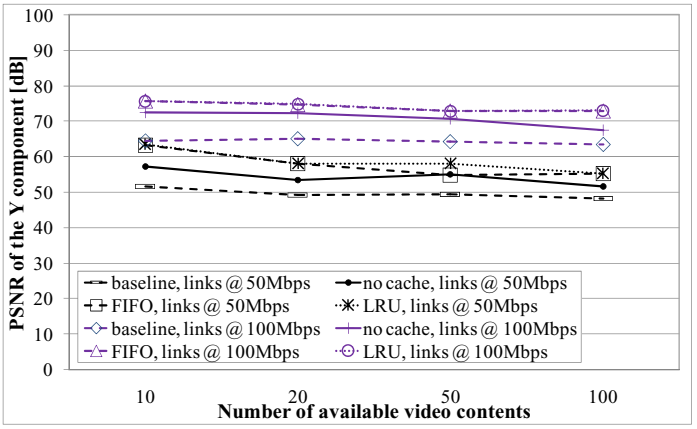


(b)

Fig. 14. PSNR of the Y component (scenario with $\alpha = 1.5$) when the PD is set to 10s and the $winT$ is equal to (a) 1/10 PD and (b) 1/2 PD, respectively



(a)



(b)

Fig. 15. PSNR of the Y component (scenario with $\alpha = 1.5$) when the PD is set to 20s and the $winT$ is equal to (a) 1/10 PD and (b) 1/2 PD, respectively

7 Conclusions and Future Works

In this book chapter we have presented a NDN-based platform providing *crowd-sourced* and real-time media contents within an online social community. It is composed by a community of users that, being into the same place to take part to an event record and broadcast it from their multiple points of view, a number of users interested to see what the aforementioned social community is capturing, a distributed *Event Management System*, which creates events and handles the social community, and a NDN communication infrastructure able to efficiently manage users requests and distribute multimedia contents. With the aim of efficiently enabling *event announcement*, *event discovering*, *media discovering*, and *media delivering* operations, we have properly designed a hierarchical

name-space and an efficient scheme, based on the *sliding-window* technique, to download real-time media contents through NDN primitives. The performances of the conceived solution have been evaluated through computer simulations, carried out with our extended version of the *ccnSim* simulator. The conducted analysis have clearly highlighted the impact that the number of media sources, the content popularity, the amount of network bandwidth dedicated to real-time services, the *playout* delay, and the network configuration have on the quality of service offered to end users. Obtained findings clearly demonstrated the effectiveness of the conceived service architecture, if compared with *baseline scenarios*. In the future, we plan to test the devised service architecture in real-testbeds, thus evaluating its pros and cons under more realistic network settings.

Acknowledgments. This work was partially supported by the PON projects (RES NOVAE, ERMES-01-03113, DSS-01-02499, and EURO6-01-02238) funded by the Italian MIUR and by the European Union (European Social Fund).

References

1. Cisco: Cisco visual networking index: Forecast and methodology, 2012–2017. White Paper (May 2013)
2. Chatzimilioudis, G., Konstantinidis, A., Laoudias, C., Zeinalipour-Yazti, D.: Crowdsourcing with smartphones. IEEE Internet Computing 16(5), 36–44 (2012)
3. Matsubara, D., Egawa, T., Nishinaga, N., Kafle, V., Shin, M.K., Galis, A.: Toward future networks: A viewpoint from ITU-T. IEEE Communication Magazine 51(3), 112–118 (2013)
4. Koponen, T., Chawla, M., Chun, B.G., Ermolinskiy, A., Kim, K.H., Shenker, S., Stoica, I.: A data-oriented (and beyond) network architecture. In: Proc. of the ACM Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM), Kyoto, Japan (2007)
5. Fotiou, N., Nikander, P., Trossen, D., Polyzos, G.C.: Developing information networking further: From PSIRP to PURSUIT. In: Tomkos, I., Bouras, C.J., Ellinas, G., Demestichas, P., Sinha, P. (eds.) Broadnets 2010. LNCS, SITE, vol. 66, pp. 1–13. Springer, Heidelberg (2012)
6. Dannewitz, C., Kutscher, D., Ohlman, B., Farrell, S., Ahlgren, B., Karl, H.: Network of Information (NetInf), An information-centric networking architecture. Computer Communications 36(7), 721–735 (2013)
7. Melazzi, N., Salsano, S., Detti, A., Tropea, G., Chiariglione, L., Difino, A., Anadiotis, A., Mousas, A., Venieris, I., Patrikakis, C.: Publish/subscribe over information centric networks: A Standardized approach in CONVERGENCE. In: Proc. of Future Network Mobile Summit (FutureNetw), Berlin, Germany (July 2012)
8. NDN: Project website (2011), `www.named-data.net/` (accessed: July 08, 2013)
9. Xylomenos, G., Ververidis, C., Siris, V., Fotiou, N., Tsilopoulos, C., Vasilakos, X., Katsaros, K., Polyzos, G.: A Survey of Information-Centric Networking Research. IEEE Communications Surveys Tutorials PP(99), 1–26 (2013)
10. Bari, M., Chowdhury, S., Ahmed, R., Boutaba, R., Mathieu, B.: A survey of naming and routing in information-centric networks. IEEE Communications Magazine 50(12), 44–53 (2012)

11. Rossini, G., Rossi, D.: Large scale simulation of ccn networks. In: Algotel (2012)
12. Myspace, http://www.myspace.com (accessed: January 7, 2014)
13. Facebook, http://www.facebook.com/ (accessed: January 7, 2014)
14. Weibo, S.: http://www.weibo.com/ (accessed: January 7, 2014)
15. Orkut, http://www.orkut.com/ (accessed: January 7, 2014)
16. Hi5, http://www.hi5.com/ (accessed: January 7, 2014)
17. NK, http://www.nk.pl/ (accessed: January 7, 2014)
18. VKontakte, http://www.vk.com/ (accessed: January 7, 2014)
19. Twitter, http://www.twitter.com/ (accessed: January 7, 2014)
20. Thumblr, http://www.thumblr.com/ (accessed: January 7, 2014)
21. Identi.ca, http://www.identi.ca/ (accessed: January 7, 2014)
22. Vine, http://www.vine.com/ (accessed: January 7, 2014)
23. Flickr, http://www.flickr.com (accessed: January 7, 2014)
24. Spotify, https://www.spotify.com/it (accessed: January 7, 2014)
25. Linkedin, http://www.linkedin.com (accessed: January 7, 2014)
26. Ceballos, M.R., Gorricho, J.L.: P2P file sharing analysis for a better performance. In: Proc. of ACM International Conference on Software Engineering (2006)
27. Liu, B., Cui, Y., Lu, Y., Xue, Y.: Locality-awareness in bittorrent-like P2P applications. IEEE Transactions on Multimedia 3(11) (April 2009)
28. Li, J.: Peer-to-Peer multimedia applications. In: Proc. of ACM International Conference on Multimedia (2006)
29. Liu, J., Rao, S.G., Li, B., Zhang, H.: Opportunities and Challenges of Peer-to-Peer Internet Video Broadcast. In: Proc. of IEEE, Special Issue on Recent Advances in Distributed Multimedia Communications (2008)
30. Xiao, X., Shi, Y., Gao, Y.: On Optimal Scheduling for Layered Video Streaming in Heterogeneous Peer-to-Peer Networks. In: Proc. of ACM International Conference on Multimedia (2008)
31. da Silva, A., Leonardi, E., Mellia, M., Meo, M.: A Bandwidth-Aware Scheduling Strategy for P2P-TV Systems. In: Proc. of IEEE International Conference on Peer-to-Peer Computing, pp. 279–288 (2008)
32. Ciullo, D., Garcia, M.A., Horvath, A., Leonardi, E., Mellia, M., Rossi, D., Telek, M., Veglia, P.: Network awareness of P2P live streaming applications: a measurement study. IEEE Transaction on Multimedia (12) (2010)
33. Jacobson, V., Smetters, D.K., Thornton, J.D., Plass, M.F., Briggs, N.H., Braynard, R.L.: Networking named content. In: ACM CoNEXT 2009 (2009)
34. Zhang, L., Estrin, D., Burke, J., Jacobson, V., Thornot, J., Smatters, D., Zhang, B., Tsudik, G., Krioukov, D., Massey, D., Papadopulos, C., Abdelzaher, T., Wang, L., Crowley, P., Yeh, E.: Named data networking (NDN) project. PARC Technical Report TR-2010-02 (October 2010)
35. Lin, W.S., Zhao, H.V., Liu, K.R.: Incentive cooperation strategies for peer-to-peer live multimedia streaming social networks. IEEE Transactions on Multimedia 11(3), 396–412 (2009)
36. Cheng, X., Liu, J.: Nettube: Exploring social networks for peer-to-peer short video sharing. In: Proc. of IEEE INFOCOM 2009, pp. 1152–1160. IEEE (2009)
37. Wang, X., Chen, M., Kwon, T., Yang, L., Leung, V.: Ames-cloud: A framework of adaptive mobile video streaming and efficient social video sharing in the clouds. IEEE Transactions on Multimedia 15(4), 811–820 (2013)
38. Wang, Z., Wu, C., Sun, L., Yang, S.: Peer-assisted social media streaming with social reciprocity. IEEE Transactions on Network and Service Management 10(1), 84–94 (2013)

39. Hoßfeld, T., Seufert, M., Hirth, M., Zinner, T., Tran-Gia, P., Schatz, R.: Quantification of YouTube QoE via crowdsourcing. In: Proc. of IEEE International Symposium on Multimedia (ISM), pp. 494–499 (2011)
40. Recursive fact-finding: A streaming approach to truth estimation in crowdsourcing applications, pp. 530–539 (2013)
41. Hei, X., Liang, C., Liang, J., Liu, Y., Ross, K.W.: A measurement study of a large-scale P2P IPTV system. IEEE Transactions on Multimedia 9(8), 1672–1687 (2007)
42. Magharei, N., Rejaie, R.: Prime: Peer-to-peer receiver-driven mesh-based streaming. IEEE/ACM Transactions on Networking (TON) 17(4), 1052–1065 (2009)
43. Jimenez, R.: Distributed Peer Discovery in Large-Scale P2P Streaming Systems: Addressing Practical Problems of P2P Deployments on the Open Internet. PhD thesis, KTH, Network Systems Laboratory (NS Lab), QC 20131203 (2013)
44. Grieco, L.A.: Emerging topics: special issue on multimedia services in information centric networks (guest editorial). IEEE COMSOC MMTC E-letter, 4–5 (July 2013)
45. Piro, G., Grieco, L.A., Boggia, G., Chatzimisios, P.: Information-centric networking and multimedia services: present and future challenges. ETT, Transactions on Emerging Telecommunications Technologies (2013) (to be published)
46. Jacobson, V., Smetters, D.K., Briggs, N.H., Plass, M.F., Stewart, P., Thornton, J.D., Braynard, R.L.: Voccn: voice-over content-centric networks. In: ACM ReArch 2009 (2009)
47. Zhu, Z., Wang, S., Yang, X., Jacobson, V., Zhang, L.: ACT: audio conference tool over named data networking. In: Proceedings of the ACM SIGCOMM Workshop on Information-Centric Networking, pp. 68–73. ACM, New York (2011)
48. Li, H., Li, Y., Lin, T., Zhao, Z., Tang, H., Zhang, X.: MERTS: A more efficient real-time traffic support scheme for Content Centric Networking. In: Proc. in IEEE Int. Conf. on Computer Sciences and Convergence Information Technology, ICCIT, pp. 528–533 (2011)
49. Han, L., Kang, S.S., Kim, H., In, H.: Adaptive retransmission scheme for video streaming over content-centric wireless networks. IEEE Communications Letters 17(6), 1292–1295 (2013)
50. Park, J., Kim, J., Jang, M.W., Lee, B.J.: Time-based interest protocol for real-time content streaming in content-centric networking (CCN). In: Proc. of IEEE Int. Conf. on Consumer Electronics, ICCE, pp. 512–513 (2013)
51. Kulinsky, D., Burke, J., Zhang, L.: Video streaming over named data networking. IEEE COMSOC MMTC E-letter, 6–9 (July 2013)
52. Pallis, G., Vakali, A.: Insight and perspectives for content delivery networks. ACM Communication Magazine, 101–106 (January 2006)
53. Vakali, A., Pallis, G.: Content delivery networks: status and trends. IEEE Internet Computing 7(6), 68–74 (2003)
54. Ahlgren, B., Dannewitz, C., Imbrenda, C., Kutscher, D., Ohlman, B.: A survey of information-centric networking. IEEE Communications Magazine 50(7), 26–36 (2012)
55. Melazzi, N.B., Chiariglione, L.: The Potential of Information Centric Networking in Two Illustrative Use Scenarios: Mobile Video Delivery and Network Management in Disaster Situations. IEEE COMSOC MMTC E-letter, 17–20 (July 2013)
56. Ciancaglini, V., Piro, G., Loti, R., Grieco, L.A., Liquori, L.: CCN-TV: a data-centric approach to real-time video services. In: in Proc. of IEEE International Conference on Advanced Information Networking and Applications, AINA, Barcelona, Spain (March 2013)

57. Piro, G., Ciancaglini, V.: Enabling real-time TV services in CCN networks. IEEE COMSOC MMTC E-letter, 17–20 (July 2013)
58. Piro, G., Cianci, I., Grieco, L.A., Boggia, G., Camarda, P.: Information centric services in smart cities. Elsevier Journal of Systems and Software 88, 169–188 (2014)
59. Omnet++, `http://www.omnetpp.org/` (accessed: January 7, 2014)
60. Wiegand, T., Sullivan, G., Bjontegaard, G., Luthra, A.: Overview of the H.264/AVC video coding standard. IEEE Transaction on Circuits and Systems for Video Technology 13(7), 560–576 (2003)
61. Chiocchetti, R., Rossi, D., Rossini, G., Carofiglio, G., Perino, D.: Exploit the known or explore the unknown?: hamlet-like doubts in ICN. In: Proc. of ACM ICN Workshop on Information-Centric Networking, pp. 7–12 (2012)
62. Rossi, D., Rossini, G.: Caching performance of content centric networks under multi-path routing (and more). In: Technical report, Telecom ParisTech.s (2011)
63. Piro, G., Grieco, L., Boggia, G., Fortuna, R., Camarda, P.: Two-level Downlink Scheduling for Real-Time Multimedia Services in LTE Networks. IEEE Transaction on Multimedia 13, 1052–1065 (2011)

Linked Open Data as the Fuel for Smarter Cities

Mikel Emaldi, Oscar Peña, Jon Lázaro, and Diego López-de-Ipiña

Deusto Institute of Technology - DeustoTech, University of Deusto,
Avda. Universidades 24, 48007, Bilbao, Spain
{m.emaldi,oscar.pena,jlazaro,dipina}@deusto.es

Abstract. In the last decade big efforts have been carried out in order to move towards the Smart City concept, from both the academic and industrial points of view, encouraging researchers and data stakeholders to find new solutions on how to cope with the huge amount of generated data. Meanwhile, Open Data has arisen as a way to freely share contents to be consumed without restrictions from copyright, patents or other mechanisms of control. Nowadays, Open Data is an achievable concept thanks to the World Wide Web, and has been re-defined for its application in different domains. Regarding public administrations, the concept of Open Government has found an ally in Open Data concepts, defending citizens' right to access data, documentation and proceedings of the governments.

We propose the use of Linked Open Data , a set of best practices to publish data on the Web recommended by the W3C, in a new data life cycle management model, allowing governments and individuals to handle better their data, easing its consumption by anybody, including both companies and third parties interested in the exploitation of the data, and citizens as end users receiving relevant curated information and reports about their city. In summary, Linked Open Data uses the previous Openness concepts to evolve from an infrastructure thought for humans, to an architecture for the automatic consumption of big amounts of data, providing relevant and high-quality data to end users with low maintenance costs. Consequently, smart data can now be achievable in smart cities.

1 Introduction

In the last decade, cities have been sensorised, protocols are constantly refined to deal with the possibilities that new hardware offers, communication networks are offered in all flavours and so on, generating lots of data that needs to be dealt with. Public administrations are rarely able to process all the data they generate in an efficient way, resulting in large amounts of data going un-analysed and limiting the benefits end-users could get from them.

Citizens are also being encouraged to adopt the role of linked open data providers. User-friendly Linked Open Data applications should allow citizens to easily contribute with new trustable data which can be linked to already existing published (more static generally) Linked Open Data provided by city councils.

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_18

In addition, people-centric mobile sensing, empowered by the technology inside actual smartphones , should progress into continuous people-centric enriched Linked Data . Linked Open Data also encourages the linkage to other resources described formally through structured vocabularies, allowing the discovery of related information and the possibility to make inferences, resulting in higher quality data.

Data management is becoming one of the greatest challenges of the 21^{st} century. Regarding urban growth, experts predict that global urban population will double by the year 2050, meaning that nearly 70% of the whole planet's inhabitants will be living in a major town or city [1]. This prediction makes clear the need to deal with the huge amounts of data generated by cities, enabling the possibility to manage their resources in an efficient way. The *Smart Data* term has been coined to address data that makes itself understandable, extracting relevant information and insights from large data sources and presenting the conclusions as human-friendly visualisations.

The problems of managing data are moving to a new level. It is not only a matter of caring about *more* data, but also how we can use it efficiently in our processes. It is about how we can deal with increasing volumes of data (from standalone databases to real *Big Data*) and integrate them to our advantage, making data useful and digestible in order to make better decisions.

In the last few years, the *Smart city* concept has been adopted by cities aware of their citizens' life quality, worried about the efficiency and trustworthiness of the services provided by governing entities and businesses. Smart data (understood as curated, high-quality and digestible data) can help cities promote to a *Smart City* status, analysing the generated data streams and providing useful information to their users: citizens, council managers, third parties, etc.

Although, efficient data lifecycle management processes need to be adopted as best practices, avoiding the provision of input data that can not be used to improve council's services, thus, incrementing the noise around high-quality data.

Our approach is based on an actual review of the state of the art regarding data lifecycle management, proposing our own model as a more refined approach to the existing ones. We also encourage the adoption of Linked Open Data principles[1] to publish both the whole generated data and the processed data, in order to allow further research on the area by third parties and the development of new business models relying on public access data. Similar proposals regarding Linked Data are defended by [2].

2 Background and Definitions

As envisaged by Sir Tim Berners-Lee, the Web is moving from an interlinked documents space to a global information one where both documents and data are linked: The Semantic Web [3]. Related to this Semantic Web , Linked Data is a set of best practices to publish data on the Web in a machine-readable way,

[1] `http://5stardata.info/`

with a explicitly defined semantic meaning, linked to other datasets and allowed to be searched for [4]. In 2006, Sir Tim Berners-Lee described a set of principles to publish Linked Data on the Web:

1. Use URIs as names for things
2. Use HTTP URIs so that people can look up those names
3. When someone looks up a URI, provide useful information, using the standard (RDF, SPARQL)
4. Include links to other URIs, so that they can discover more things

Later, in 2010, he established a five-star rating system to encourage people and governments to publish high-quality Linked Open Data:

⋆	Available on the web (whatever format) but with an **open licence**, to be Open Data .
⋆ ⋆	Available as **machine-readable** structured data (e.g. excel instead of image scan of a table).
⋆ ⋆ ⋆	as (2) plus **non-proprietary format** (e.g. CSV instead of excel).
⋆ ⋆ ⋆ ⋆	All the above plus: Use **open standards** from W3C (RDF and SPARQL) to identify things, so that people can point at your stuff.
⋆ ⋆ ⋆ ⋆ ⋆	All the above plus: **Link your data** to other people's data to provide context.

Over the years, enormous amount of applications based on Linked Data have been developed. Big companies like Google[2], Yahoo![3] or Facebook[4] have lately invested resources on deploying Semantic Web and Linked Data technologies. Several governments from countries like the United States of America or the United Kingdom have published a big amount of Open Data from different administrations in their Open Data portals.

Linked Data is real and can be the key for data management in smart cities. Following these recommendations, data publishers can move towards a new data-powered space, in which data scientists and application developers can research on new uses for Linked Open Data.

3 Data Life Cycle

Through the literature, a broad variety of different definitions of data life cycle models can be found. Although they have been developed for different actuation domains, we describe here some of them which could be applied for generic data, independently of its original domain.

[2] http://www.google.com/insidesearch/features/search/knowledge.html
[3] http://semsearch.yahoo.com/
[4] http://ogp.me/

3.1 Data Documentation Initiative

The first model to be analysed is the model proposed by the Data Documentation Initiative (DDI). The DDI introduced a Combined Life Cycle Model for data managing [5]. As Figure 1 shows, this model has eight elements or steps which can be summarised as follows, according to [6]:

- **Study concept.** At this stage, apart from choosing the research question and the methodology to collect data, the processing and analysis stage of the needed data to answer the question is planned.
- **Data collection.** This model proposes different methods to collect data, like surveys, health records, statistics or Web-based collections.
- **Data processing.** At this stage, the collected data are processed to answer the proposed research question. The data may be recorded in both machine-readable and human-readable form.
- **Data archiving.** Both data and metadata should be archived to ensure long-term access to them, guaranteeing confidentiality.
- **Data distribution.** This stage involves the different ways in which data are distributed, as well as questions related to the terms of use of the used data or citation of the original sources.
- **Data discovery.** Data may be published in different manners, through publications, web-indexes, etc.
- **Data analysis.** Data can be used by others to achieve different goals.
- **Repurposing.** Data can be used outside of their original framework, restructuring or combining it to satisfy diverse purposes.

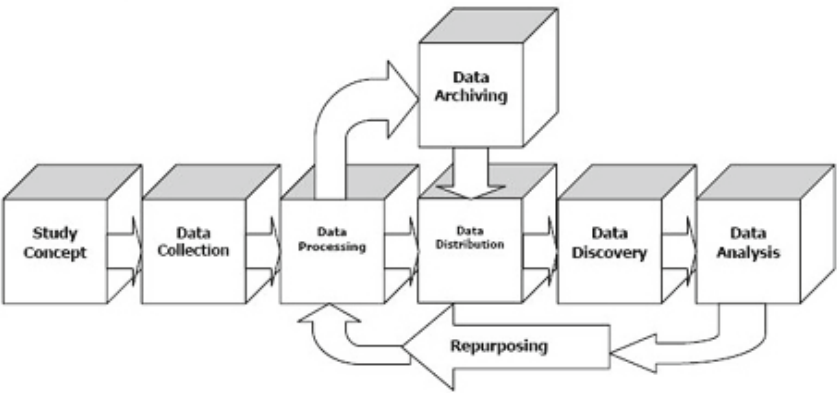


Fig. 1. Combined Life Cycle Model (ownership: DDI Alliance)

3.2 Australian National Data Service

In late 2007, the *Australian National Data Service* (ANDS) was founded with the objective of creating a national data management environment. ANDS established a set of verbs, denominated *Data Sharing Verbs*, that describe the entire life cycle of the data [7]:

- **Create.** *Create* (or *collect* for disciplines with an observational focus) is about the kinds of metadata that could be collected and the tools to fulfil this gathering task.
- **Store.** This *Data Sharing Verb* remarks the need for stable and web-accessible storage, taking care of the appropriate storing of data.
- **Describe.** The more information inside the storage, the more difficult its discovery, access and exploitation is. Annotating the data with the proper metadata solves this issue.
- **Identify.** The application of this verb implies the proper identification of each data resource, assigning a persistent identifier to each of them.
- **Register.** This step pertains to record the descriptions of the different data collections with one or more public catalogues.
- **Discover.** To improve data-reusing, ANDS suggests to enable different discovery services.
- **Access.** To guarantee the appropriate access to data, ANDS advises to provide a suitable search engine to retrieve these data. If data is not electronically available, ANDS recommends to provide contact details to get data in conventional formats.
- **Exploit.** *Exploit*, the final *Data Sharing Verb*, comprises the tools, methodologies and support actions to enable reutilisation of data.

3.3 Ecoinformatics Data Life Cycle

Michener and Jones define in [8] the concept of "ecoinformatics": *a framework that enables scientists to generate new knowledge through innovative tools and approaches for discovering, managing, integrating, analysing, visualising and preserving relevant biological, environmental, and socioeconomic data and information.* To manage these data, the following data life cycle has been defined, as can be seen at Figure 2:

- **Plan.** This step involves the confection of a data management planning.
- **Collect.** This step considers both manual (hand-written data sheets) and automatic (sensor networks) data-gathering methods.
- **Assure.** Quality assurance and quality control (QA/QC), an issue addressed in previously mentioned models, is not taken into account. Michener and Jones proposal is based on developing methods to guarantee the integrity of data. Quality assurance can also include the definition of standards for formats, codes, measurement units, metadata, etc.
- **Describe.** As other data life cycle models, this model remarks the value of the metadata to answer questions about *who, when, where, how* and *why.*

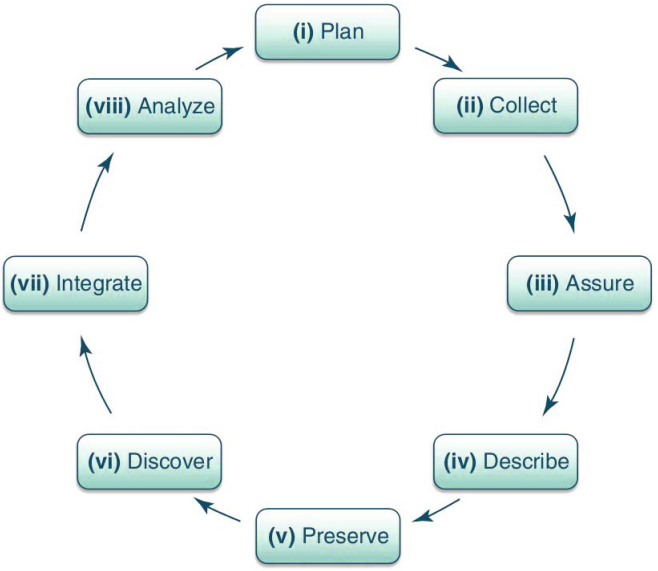


Fig. 2. Data life cycle in ecoinformatics. Taken from [8].

- **Preserve.** Data preservation implies the storage of the data and metadata, ensuring that these data can be verified, replicated and actively curated over time.
- **Discover.** The authors describe the data discovering process as *one of the greatest challenges*, as many data are not immediately available because they are stored in individual laptops. The main challenges to publish the data in a proper way are related to the creation of catalogues and indexes, and about the implementation of the proper search engines.
- **Integrate.** Integrating data from different and heterogeneous sources can become a difficult task, as it requires *understanding methodological differences, transforming data into a common representation and manually converting and recording data to compatible semantics before analysis can begin.*
- **Analyse.** As well as the importance of a clear analysis step, this models remarks the importance of documenting this analysis with sufficient detail to enable its reproduction in different research frameworks.

3.4 UK Data Archive

Another data life cycle model is the one proposed by *UK Data Archive*[5]. This model is oriented to help researchers publish their data in a manner that allows other researchers to continue their work independently. In Figure 3, the following stages can be observed:

[5] `http://www.data-archive.ac.uk/create-manage/life-cycle`

- **Creating data.** Creating the data involves the design of the research question, planning how data are going to be managed and their sharing strategy. If we want to reuse existing data, we have to locate existing data and collect them. Whether data is new or existing, at this stage the metadata has to be created.
- **Processing data.** Like in other models, at this stage the data is translated, checked, validated and cleaned. In the case of confidential data, it needs to be "anonymized". The UK Data Archive recommends the creation of metadata at this stage too.
- **Analysing data.** At this stage, data are interpreted and derived into visualisations or reports. In addition, the data are prepared for preservation, as mentioned in the following stage.
- **Preserving data.** To preserve data properly, they are migrated to the best format and stored in a suitable medium. In addition to the previously created metadata, the creating, processing, analysis and preserving processes are documented.
- **Giving access to data.** Once the data is stored, we have to distribute our data. Data distribution may involve controlling the access to them and establish a sharing license.
- **Re-using data.** At last, the data can be re-used enabling new research topics.

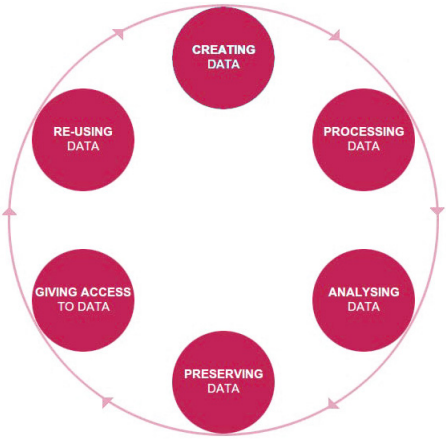


Fig. 3. Data life cycle proposed by UK Data Archive

3.5 The LOD2 Stack Data Life Cycle

The last analysed data life cycle has been developed under the LOD2[6] project. This project proposes a technological and methodological stack which supports

[6] http://lod2.eu/

Fig. 4. Linked Data life cycle from LOD2 stack

the entire life cycle of Linked Data [9]. As Figure 4 shows, the proposed life cycle phases are the following:

- **Storage.** As RDF data presents more challenges than relational data, they propose the collaboration between known and new technologies, like column-store technology, dynamic query optimisation, adaptive caching of joins, optimised graph processing and cluster/cloud scalability.
- **Authoring.** LOD2 provides provenance about data collected through distributed social, semantic collaboration and networking techniques.
- **Interlinking.** At this phase, LOD2 offers approaches to manage the links between different data sources.
- **Classification.** This stage deals with the transformation of raw data into Linked Data. This transformation implies the linkage and integration of data with upper level ontologies.
- **Quality.** Like other models, LOD2 develops techniques for assessing quality based on different metrics.
- **Evolution/Repair.** At this stage, LOD2 deals with the dynamism of the data from the Web, managing changes and modifications over the data.
- **Search/Browsing/Exploration.** This stage is focussed on offering Linked Data to final users through different search, browsing, exploration and visualisation techniques.

3.6 A Common Data Life Cycle for Smart Cities

Based on these data life cycle models, we propose a common data life cycle for managing data in Smart Cities . As can be seen in Figure 5, the different stages of the mentioned models have been aggregated, forming our proposed model.

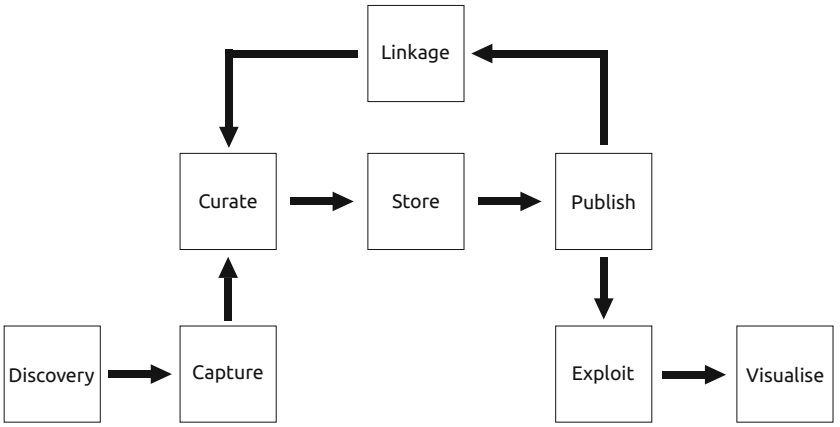


Fig. 5. Proposed model

The different stages of this model, which are going to be explained widely in following sections, are:

- **Discovery:** The first step in our model consists of discovering where data can be taken from, identifying the available datasets which contain the necessary data to accomplish our task. Datasources can either be maintained by us, or by external entities, so the more metadata we can gather from datasets, the easier further steps will become.
- **Capture:** Once datasources are identified, data need to be collected. In a Smart City environment, there are a lot of alternatives to capture data, like sensors, data published by public administration, social networks , mobile sensing, user generated data or in more traditional ways like surveys.
- **Curate:** After the required data are captured, they are prepared to be stored and in need of proper methods to explore them. This processing involves the analysing, refining, cleaning, formatting and transformation of the data.
- **Store:** The storage of data is, probably, the most delicate action in the life cycle. Above the storage all the analysis tools are build, and is the "final endpoint" when someone requests our data. A suitable storage should have indexing, replication, distribution and backup features, among other services.
- **Publish:** Most of the previously mentioned models prioritise the analysing stage over the publication stage. In our model, we defend the opposite approach for a very simple reason: when you exploit your data before their publication, and using different processes as the rest of the people who is

going to use them, you are not making enough emphasis on publishing these data correctly. Everybody has ever met a research paper or an application in which accessing the data was difficult, or, when once the data was collected it became totally incomprehensible. To avoid this issue, we propose to publish the data before their consumption, following the same process the rest of the people do.
- **Linkage:** Before consuming data, we suggest to search for links and relationships with other datasets found in the discovery step. Actual solutions do not allow the linkage with unknown datasets, but tools are developed to ease link discovery processes between two or more given datasources.
- **Exploit:** Once the data is published, we use the provided methods to make use of the data. This data consumption involves data mining, analytics or reasoning.
- **Visualise:** To understand data properly, designing suitable visualisations is essential to show correlations between data and the conclusions of the data analysis in a human-understandable way.

4 Identified Challenges

Taking into consideration the large amounts of data present at Smart Cities , data management's complexity can be described in terms of:

- Volume
- Variety
- Veracity
- Velocity

This four variables can also be found in *Big Data*-related articles (also known as the *Big Data's Vs*) [10, 11], so it is not surprising at all that Smart Cities are going to deal with Big Data problems in the near future (if they are not dealing with them right now).

Data scientist need to take into account these variables, which could overlap in certain environments. Should this happen, each scenario will determine the most relevant factors of the process, generating unwelcome drawbacks on the other ones.

4.1 Volume

The high amount of data generated and used by cities nowadays needs to be properly analysed, processed, stored and eventually accessible. This means conventional IT structures need to evolve, enabling scalable storage technologies, distributed querying approaches and massively parallel processing algorithms and architectures.

There is a growing trend which defends that the minimum amount of data should be stored and analysed without significantly affecting the overall knowledge that could be extracted from the whole dataset. Based on *Pareto's Principle*

(also known as the 80-20 rule), the idea is to focus on a 20% of the data to be able to extract up to the 80% of knowledge within it. Even being a solid research challenge in Big Data, there are occasions where we cannot discard data from being stored or analysed (e.g. sensor data about monitoring a building can be used temporally, while patient monitoring data should be kept for historical records).

However, big amounts of data should not be seen as a drawback attached to Smart Cities. The larger the datasets, the better analysis algorithms can perform, so deeper insights and conclusions should be expected as an outcome. These could ease the decision making stage.

As management consultant Peter Drucker once said: *"If you can not measure it, you can not manage it"*, thus, leaving no way to improve it either. This adage manifests that should you want to take care of some process, but you are not able to measure it or you can not access the data, you will not be able to manage that process. That being said, the higher amounts of data available, the greater the opportunities of obtaining useful knowledge will become.

4.2 Variety

Data is rarely found in a perfectly ordered and ready for processing format. Data scientists are used to work with diverse sources, which seldom fall into neat relational structures: embedded sensor data, documents, media content, social data, etc. As it can be seen in Section 5.2 there are many different sources where data can come from in a smart city. Despite of presented data cycle can be applied for all kind of data, the different steps of the cycle have to be planned, avoiding the overloading of implemented system. For example, data from social media talking about an emergency situation may be prioritised over the rest of the data, to allow Emergency Response Teams (ERTs) react as soon as possible.

Moreover, different data sources can describe the same real-world entities in such different ways, finding conflicting information, different data-types, etc. Taking care of how data sources describe their contents will lead to an easier integration step, lowering development, analytics and maintenance costs over time.

4.3 Veracity

There is also an increasing concern on data trustworthiness. Different data sources can have meaningful differences in terms of quality, coverage, accuracy, timeliness and consistency of the provided data. In fact, [12] conclude that redundancy, consistency, correctness and copying between sources are the most recurrent issues we have to deal with when we try to find trustworthy information from a wide variety of heterogeneous sources.

As pointed out by [13], *data provenance is fundamental to understanding data quality* . They also highlight that established information storage systems may not be adequate to keep semantic sense of data.

Several efforts are trying to convert existing data in high-quality data, providing an extra confidence layer in which data analysts can rely. In a previous research [14], we introduced a provenance data model to be used in user-generated Linked Data datasets, which follow W3C's PROV-O ontology[7]. Some other researches as [15] or [16] also provide some mechanisms to measure the quality and trust in Linked Data.

4.4 Velocity

Finally, we must assume that data generation is experiencing an exponential growth. That forces our IT structure to not only tackle with volume issues, but also with high processing rates. A widely spread concept among data businesses is that sometimes you can not rely on five-minute-old data for your business logic.

That is why *streaming data* has moved from academic fields to industry to solve velocity problems. There are two main reasons to consider streaming processing:

- Sometimes, input data is too fast to store in their entirety without rocketing costs.
- If applications mandate immediate response to the data, batch processes are not suitable. Due to the rise of smartphone applications, this trend is increasingly becoming a common scenario.

5 Linked Open Data as a Viable Approach

In the previous section, we identified some of the challenges Smart Cities will need to face in the following years. The data lifecycle model proposed at Figure 5 relies on Linked Open Data principles to try to solve these issues, reducing costs and enabling third parties to develop new business models on top of Linked Open Data.

Next we describe, how Linked Open Data principles could help in the model's stages:

5.1 Discovery

Before starting any process related with data management, where that data can be found must be known. Identifying the data sources that can be queried is a fundamental first step in any data life cycle. These data sources can be divided in two main groups: *a)* internal, when the team in charge of creating and maintaining the data is the same that makes use of it, or *b)* external, when data is provided by a third party.

The first scenario usually provides a good understanding of the data, as their generation and structure is designed by the same people who are going to use

[7] `http://www.w3.org/TR/prov-o/`

them. Whereas in real applications, its becoming more common to turn to external data sources to use in the business logic algorithms. Data scientists and developers make use of external datasets for analysing them, expecting to get new insights and create new opportunities from existing data. Luckily, some initiatives help greatly whilst searching for new Open Data sources.

- *The Datahub*[8] is a data management platform from the *Open Knowledge Foundation*, providing nearly 11,000 open datasets as of September 2013. The Datahub relies on *CKAN*[9], an open-source software tool for managing and publishing collections of data. The Datahub's datasets are openly accessible, but data formats can vary from CSV files to RDF , going through JSON, XML, etc.
- The *Linking Open Data Cloud* (LOD Cloud)[10] is an Open Data subset whose catalogues are available on the Web as Linked Data, containing links to other Linked Datasets. LOD Cloud is commonly referred as the biggest effort to bring together Linked Open Data initiatives, grouping 337 datasets as of September 2013. The central node of the LOD Cloud is DBpedia [17, 18], a crowdsourced community effort to extract structured information from Wikipedia and make it available on the Web.
- Sindice [19] is a platform to build applications on top of semantically markup data on the Web, such us RDF, RDFa, Microformats or Microdata. The main difference is that Sindice does not keep the found documents, but the URL where semantic data can be found. This makes Sindice the closest approach to a traditional document search engine adapted for the Semantic Web.
- Finally, Sig.ma [20] uses Sindice's search engine to construct a view on top of the discovered data on the Web in an integrated information space.

The projects described above can establish the basis to search for external data sources, on top of which further analysis and refinement processes can be built.

5.2 Capture

Data are undoubtedly the basis of Smart Cities: services offered to citizens, decisions offered to city rulers by Decision Support Systems, all of them work thanks to big amounts of data inputs. These data are captured from a wide variety of sources, like sensor networks installed along the city, social networks, publicly available government data or citizens/users who prosume data through their devices (they crowdsource data, or the devices themselves generate automatically sensing data). In most cases, these sources publish data in a wide set of heterogeneous formats, forcing data consumers to develop different connectors for each source. As can be seen in Section 5.3, there are a lot of different and widely extended ontologies which can represent data acquired from sources found in

[8] `http://datahub.io/`
[9] `http://ckan.org/`
[10] `http://lod-cloud.net/`

Smart Cities, easing the capture, integration and publication of data from heterogeneous domains. In this section, different sources of data which can be found in Smart Cities are shown, while in Section 5.3 the transformation process from their raw data to Linked Data is exposed.

Sensor Networks. A sensor network is composed by low-cost, low-power, small sized and multifunctional sensor nodes which are densely deployed either inside the phenomenon or very close to it [21]. In a smart city, these sensor networks are used for a wide range of applications, from the simple analysis of air quality[11] to the complex representation of public transport services[12], through the sensors embedded in citizens smartphones. For example, the SmartSantander project envisions the deployment of 20,000 sensors in four European cities [22]. Nowadays, due to the existence of open source and cheap hardware devices like Arduino[13] or Raspberry Pi[14], the amount of collaborative and social sensor networks is growing faster and faster. Furthermore, there are software platforms like Xively[15] or Linked Sensor Middleware [23], which allow users to share the captured data from their own sensor networks in an easy way.

Social Networks. Since the adoption of the Web 2.0 paradigm [24], users have become more and more active when interacting with the Web. The clearest example of this transformation of the Web can be found in social networks and the high growth of their users. For example, at the end of the second quarter of 2013, Facebook has almost 1.2 billion users[16], while at the end of 2012, Twitter reached more than 200 million monthly active users[17]. Although users of social networks generate a lot of data, it is hard to manipulate them because users write in a language not easily understood by machines. To solve this issue many authors have worked with different Natural Language Processing (NLP) techniques. For example, NLP and Named Entity Recognition (NER) systems [25] can be used to detect tweets which talk about some emergency situation like a car crash, an earthquake and so on; and to recognise different properties about the emergency situation like the place or the magnitude of this situation [26, 27]. Extracting data from relevant tweets could help emergency teams when planning their response to different types of situations as can be seen at [28–30].

Government Open Data. Government Open Data has gained a lot of value in recent years, thanks to the proliferation of Open Data portals from different administrations of the entire World. In these portals, the governments publish

[11] http://helheim.deusto.es/bizkaisense/
[12] http://traintimes.org.uk/map/tube/
[13] http://www.arduino.cc/
[14] http://www.raspberrypi.org/
[15] https://xively.com
[16] http://techcrunch.com/2013/07/24/facebook-growth-2/
[17] https://twitter.com/twitter/status/281051652235087872

relevant data for the citizens, in a heterogeneous set of formats like CSV, XML or RDF. Usually, data from these portals can be consumed by developers in an easy way thanks to the provided APIs, so there are a lot of applications developed over these data. As citizens are the most important part of Smart Cities, these applications make them an active part in the governance of the city.

To illustrate the importance of Government Open Data, in Table 1 some Open Data portals are shown.

Table 1. Open Data portals around the World

Name	**Public Administration**	**No. of datasets (Sept. 2013)**	**API**
Data.gov	Government of USA	97,536	REST, SOAP, WMS
Data.gov.uk	Government of UK	10,114	REST
Data.gc.ca	Government of Canada	197,805	REST
Open Data Euskadi	Government of Basque Country	2,127	RSS, Java API, REST
Datos Abiertos de Zaragoza	Council of Zaragoza	112	SPARQL

Mobile Sensing and User Generated Data. Another important data source in which Smart Cities can capture data are the citizens themselves. Citizens can interact with city in multiple ways: for example, in [14] an interactive 311 service is described. In this work, authors propose a model to share and validate, through provenance and reputation analysis, reports about city issues published by its citizens. In Urbanopoly [31] and Urbanmatch [32], different games are presented to tourist with the aim of gathering data and photographies from tourist points of interest in the city using their smartphones' cameras. In the same line, csxPOI [33] creates semantically annotated POIs through data gathered by citizens, allowing semi-automatic merging of duplicate POIs and removal of incorrect POIs.

As have been shown, in a Smart City a lot of data sources can be found, publishing an abundant stream of interesting data in a different and heterogeneous manner. In section 5.3, how to transform these data into standard formats is shown.

5.3 Curate

As it can be seen in Section 2, the Linked Data paradigm proposed the Resource Description Framework (RDF) as the best format to publish data and encourage the reuse of widely extended ontologies. In this section we explain **what** is an

ontology, **which** are the most popular ontologies and **how** we can map previously captured raw data to a proper ontology. At the end of this section a set of best practices to construct suitable URIs for Linked Data are shown.

As defined by [34], an ontology is *a formal explicit description of concepts in a domain of discourse, properties of each concept describing various features and attributes of the concept and restrictions on slots*. According to this definition, an ontology has *Classes* which represent the concept, *Properties* which represent different characteristics of *Classes* and *Restrictions* on the values of these properties and relationships among different *Classes*. An ontology allows modelling data avoiding most ambiguities originated when fusing data from different sources, stimulating the interoperability among different sources. As seen in Section 5.2, data may come from a wide variety of sources in Smart Cities, whereby the ontologies seem to be a suitable option to model these data.

The following works use ontologies to model different data sources which can be found in a Smart City. In Bizkaisense project [35], diverse ontologies like Semantic Sensor Network ontology (SSN) [36], Semantic Web for Earth and Environmental Terminology (SWEET) [37] or Unified Code for Units of Measure ontology (UCUM)[18] are used to model raw data from air quality stations from the Basque Country. AEMET Linked Data project[19] has developed a network of ontologies composed by SSN ontology, OWL-Time ontology[20], wsg84_pos ontology[21], GeoBuddies ontology network[22] and its own AEMET ontology, to describe measurements taken by meteorological stations from AEMET (Spanish National Weather Service). In [38] authors extend SSN ontology to model and publish as Linked Data the data stream generated by the sensors of an Android powered smartphone.

Another example of semantic modelling of infrastructures from a city can be found in LinkedQR [39]. LinkedQR is an application that eases the managing task of an art gallery allowing the elaboration of interactive tourism guides through third parties Linked Data and manual curation. LinkedQR uses MusicOntology [40] to describe the audioguides and Dublin Core [41], DBpedia Ontology and Yago [42] to describe other basic information.

LinkedStats project[23] takes data about waste generation and population of Biscay to develop a statistical analysis about the correlation between these two dimensions of the data. It models these statistical data through the RDF Data Cube Vocabulary [59], an ontology developed for modelling multi-dimensional data in RDF. At last, in [43] the authors show how Linked Data enables the integration of data from different sensor networks.

[18] http://idi.fundacionctic.org/muo/ucum-instances.html
[19] http://aemet.linkeddata.es/models.html
[20] http://www.w3.org/TR/owl-time/
[21] http://www.w3.org/2003/01/geo/wgs84_pos
[22] http://mayor2.dia.fi.upm.es/oeg-upm/index.php/en/ontologies/83-geobuddies-ontologies
[23] http://helheim.deusto.es/linkedstats/

The mapping between raw data and ontologies, usually is made by applications created *ad hoc* to each case; Bizkaisense, AEMET Linked Data and LinkedStats have their own Python scripts to generate proper RDF files from raw data. In the case of LinkedQR, it has a control panel where the manager can manually type data and map to a desired ontology. Instead, there are tools designed for transforming raw data into structured data. One of them is Open Refine[24] (formerly Google Refine). Open Refine is a webtool which can apply different manipulations to data (facets, filters, splits, merges, etc.) and export data in different formats based on custom templates. Additionally, Google Refine's RDF Extension allows exporting data in RDF.

Another interesting tool is Virtuoso Sponger, a component of OpenLink Virtuoso[25] which generates Linked Data from different data sources, through a set of extractors called *Cartridges*. There are different Cartridges which support a wide variety of input formats (CSV, Google KML, xHTML, XML, etc.) and vendor specific Cartridges too (Amazon, Ebay, BestBuy, Discogs, etc.).

After modelling data, one of the most important concepts in Linked Data are the URIs or Unified Resource Identifiers. As shown in Section 2, to publish a data resource as Linked Data it has to be identified by an HTTP URI which satisfies these conditions:

- An HTTP URI is unique and consistent.
- An HTTP URI can be accessed from everywhere by everyone.
- The URI and its hierarchy are auto-descriptive.

Designing valid URIs is a very important step into the publication of Linked Data: if you change the URIs of your data, all the incoming links from external sources are going to be broken. To avoid this issue there is a set of good practices, proposed in [44]:

- **Be on the web.** Return RDF for machines and HTML for humans through standard HTTP protocol.
- **Don not be ambiguous.** Use a URL to describe a document and a different URI to identify real-world objects. A URI can not stand for both document and real-world object.

To apply these good practices correctly, authors propose two solutions, which nowadays have been widely adopted by Linked Data community:

- **303 URIs.** Use `303 See Other` status code for redirecting to the proper RDF description of the real-world object or to the HTML document. For example:
  - `http://helheim.deusto.es/hedatuz/resource/biblio/5112` - A URI identifying a bibliographic item.
  - `http://helheim.deusto.es/hedatuz/page/biblio/5112` - The HTML view of the item.

[24] `http://openrefine.org/`
[25] `http://virtuoso.openlinksw.com/`

  - `http://helheim.deusto.es/hedatuz/data/biblio/5112` - A RDF document describing the item.
- **Hash URIs.** URIs contains a *fragment*, a special part separated by the symbol #. The client has to strip-off the fragment from the URI before requesting it to server. Server returns a RDF document in which the client has to search the fragment.

Tools implementing these techniques are described on Section 5.5.

5.4 Store

Once data is mapped to a proper ontology and the RDF files are generated, is time to store them. Due to the big amount of data generated in a city, an appropriate storage has to:

- Support different input data formats.
- Manage and index big amounts of data properly.
- Execute queries over data in an efficient way.
- Offer different interfaces and APIs to users, allowing them to exploit data in a wide variety of formats.

Along this section, the first three points are discussed, while the fourth is discussed in Section 5.5. Before a wide description of each analysed datastore is given, a brief description is presented in Table 2.

Table 2. Summary of characteristics of selected datastores

Datastore	Input formats	API	Output formats
Virtuoso	RDF/XML, N3, Turtle, N-Triples, N-Quads, etc.	SPARQL, SPARQL UPDATE, Java (Jena, Sesame, Redland)	HTML, Spreadsheet, XML, RDF+JSON, JS, NTriples, RDF/XML, CSV
Stardog	NTRIPLES, RDF/XML, Turtle, TRIG, TRIX, N3, N-Quads	SPARQL (CLI, HTTP), Java, JS, Groovy, Spring, Ruby	N-Triples, RDF/XML, TURTLE, TRIG, TRIX, N3, N-Quads
Fuseki	RDF/XML, N-Triples, Turtle, etc.	SPARQL, SPARQL UPDATE, Java	JSON, XML, Text, CSV, TSV

The first datastore to be reviewed is Virtuoso by Openlink, mentioned in Section 5.3. Virtuoso is a hybrid server that manages SQL, XML and RDF in a single server. It is available in both Open Source and commercial versions. As seen in Table 2, it supports a wide variety of input and output formats. It has a SPARQL endpoint, accessible thus by web-interface as by HTTP GET requests,

allowing to web-agents the access to data. Virtuoso supports the SPARQL UPDATE syntax, allowing the update of datastore through HTTP POST requests; and it provides connectors for different Java powered RDF engines, like Jena[26], Sesame[27] or Redland[28]. Further, it supports some OWL properties for reasoning. According to the Berlin SPARQL Benchmark [45] Virtuoso 7 can load one billion triples in 27:11 minutes.

Another datastore which is becoming popular is Stardog[29]. Developed by Clark & Parsia, Stardog is a RDF database which supports SPARQL querying and OWL reasoning. It offers a Command Line Interface to manage the different databases (create, remove, add/remove data, SPARQL queries, etc.), while they can be queried through HTTP too. Furthermore it has its own query syntax which indexes the RDF literals; and its own Java library to manage databases from Java applications. It supports OWL 2 reasoning, supporting different OWL profiles[30] like QL, RL, EL or DL.

The last analysed datastore is Fuseki[31]. Part of Jena's framework, Fuseki (formerly known as Joseki) offers RDF data over HTTP, in a REST style. Fuseki implements W3C's SPARQL 1.1 Query, Update, Protocol and Graph Store HTTP Protocol. It has a web-panel to manage the datastore and can interact with the rest of the Jena components.

As can be seen at Table 2, all analysed datastores are similar in terms of input/output formats or offered APIs. But there are differences in other aspects, like security: Fuseki does not support users nor access roles. Another difference is the installation and executing complexity: while Fuseki and Stardog are launched as a Java JAR file, Virtuoso can be installed through Debian's package system and launched as a UNIX daemon. In the other hand, Virtuoso is more than a "simple" RDF store, Virtuoso is a relational database engine, an application server in which both preinstalled applications and our own applications can be launched, and much more. Furthermore, Virtuoso can be installed in a cluster formed by multiple servers.

Concluding this section, we can say that Fuseki can be used in light-weight installations, when hardware and data are limited; Stardog in more complex systems, due to its fast query execution times. Meanwhile, Virtuoso offers more services like Sponger (described in Section 5.3) or Semantic Wiki, whereby, it can be suitable for environments which need more than the simple storage of RDF triples.

5.5 Publish

Publication stage is one of the most important stages in the life cycle of Linked Data in Smart Cities, because this stage determines how citizens or developers can

[26] http://jena.apache.org/
[27] http://www.openrdf.org/
[28] http://librdf.org/
[29] http://stardog.com/
[30] http://www.w3.org/TR/owl2-profiles/
[31] http://jena.apache.org/documentation/serving_data/index.html

acquire Linked Data to exploit (Section 5.7) through different discovery methods (Section 5.1). As we saw in section 5.4, the three proposed RDF stores include some publication API or SPARQL endpoint, but, sometimes the specifications of the system to be deployed need additional features at this publication stage.

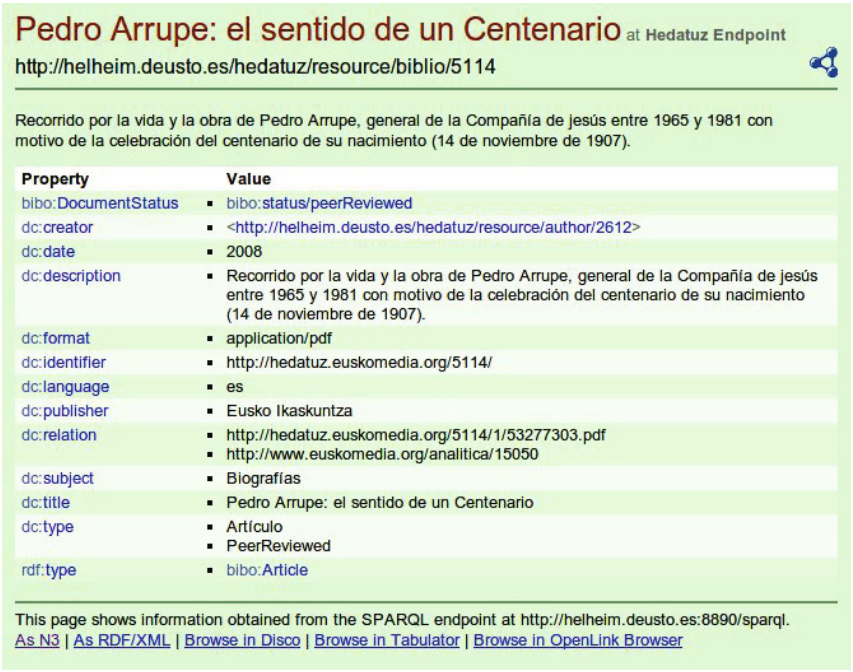


Fig. 6. Example of HTML visualisation of a resource from Hedatuz dataset by Pubby

One of these features can be the `303 Redirection` explained at Section 5.3. Although all datastores mentioned on Section 5.4 offer a SPARQL endpoint to explore data, Linked Data paradigm demands resolvable HTTP URIs as resource identifiers. Fortunately, there are tools which fulfil this demand. One of them, Pubby[32], adds Linked Data interfaces to SPARQL endpoint. Pubby queries the proper SPARQL endpoint to retrieve data related to a given URI and manages the `303 Redirection` mechanism. Depending on the `Accept` header of the HTTP request, Pubby redirects the client to the HTML view of data or to the RDF document describing the resource. Pubby can export data in RDF/XML, NTriples, N3 and Turtle. In Figure 6, an example of the HTML view of a resource is shown.

D2R Server [46] allows the publication of relational databases as Linked Data. At first D2R requires a mapping from the tables and columns of the database to

[32] `http://wifo5-03.informatik.uni-mannheim.de/pubby/`

```
# D2RQ Namespace
@prefix d2rq:        <http://www.wiwiss.fu-berlin.de/suhl/bizer/D2RQ/0.1#> .
# Namespace of the ontology
@prefix : <http://annotation.semanticweb.org/iswc/iswc.daml#> .

# Namespace of the mapping file; does not appear in mapped data
@prefix map: <file:///Users/d2r/example.ttl#> .

# Other namespaces
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

map:Database1 a d2rq:Database;
    d2rq:jdbcDSN "jdbc:mysql://localhost/iswc";
    d2rq:jdbcDriver "com.mysql.jdbc.Driver";
    d2rq:username "user";
    d2rq:password "password";
    .
# ------------------------------------------------
# CREATE TABLE Conferences (ConfID int, Name text, Location text);

map:Conference a d2rq:ClassMap;
    d2rq:dataStorage map:Database1.
    d2rq:class :Conference;
    d2rq:uriPattern "http://conferences.org/comp/confno@@Conferences.ConfID@@";
    .
map:eventTitle a d2rq:PropertyBridge;
    d2rq:belongsToClassMap map:Conference;
    d2rq:property :eventTitle;
    d2rq:column "Conferences.Name";
    d2rq:datatype xsd:string;
    .
map:location a d2rq:PropertyBridge;
    d2rq:belongsToClassMap map:Conference;
    d2rq:property :location;
    d2rq:column "Conferences.Location";
    d2rq:datatype xsd:string;
    .
```

Fig. 7. D2RQ mapping file. Example taken from http://d2rq.org/d2rq-language.

selected ontologies using D2RQ Mapping Language. Once this mapping is done, D2R offers a SPARQL endpoint to query data and a Pubby powered interface.

Besides the publication of the data itself, it is important to consider the publication of provenance information about it. In [47] the authors identify the publication of provenance data as one of the main factors that influence web content trust. At the time of publishing provenance information two approaches can be taken: the first is to publish basic metadata like when or who created and published the data; the second is to provide a more detailed description of where the data come from, including versioning information or the description of the data transformation workflow, for example. Some ontologies help us in the process of providing provenance descriptions of Linked Data. Basic provenance metadata can be provided using Dublin Core terms, like *dcterms*[33]*:contributor*, *dcterms:creator* or *dcterms:created*. Other vocabularies like the Provenance Vocabulary [48] or the Open Provenance Model (OPM) [49] provide ways to publish detailed provenance information like the mentioned before. The W3C has recently created the PROV Data Model [50], a new vocabulary for provenance interchange on the Web. This PROV Data Model is based on OPM

[33] `http://purl.org/dc/terms/`

and describes the entities, activities and people involved in the creation of a piece of data, allowing the consumer to evaluate the reliability of the data based on the their provenance information. Furthermore, PROV was deliberately kept extensible, allowing various extended concepts and custom attributes to be used. For example, the Uncertainty Provenance (UP) [51] set of attributes can be used to model the uncertainty of data, aggregated from heterogeneously divided trusted and untrusted sources, or with varying confidence.

Here we can find an example of how bio2rdf.org -an atlas of post-genomic data and one of the biggest datasets in the LOD Cloud- represent the provenance data of its datasets, using some of the mentioned ontologies in conjunction with VoID, a dataset description vocabulary:

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix void: <http://rdfs.org/ns/void#> .
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix prov: <http://www.w3.org/ns/prov#> .

<http://bio2rdf.org/bio2rdf_dataset:bio2rdf-affymetrix-20121002>
    a void:Dataset ;
    dcterms:created "2012-10-02"^^xsd:date ;
    rdfs:label "affymetrix dataset by Bio2RDF on 2012-10-02" ;
    dcterms:creator <affymetrix> ;
    dcterms:publisher <http://bio2rdf.org> ;
    void:dataDump <datadump> ;
    prov:wasDerivedFrom
        <http://bio2rdf.org/bio2rdf_dataset:affymetrix> ;
    dcterms:rights "restricted-by-source-license" ,
        "attribution" , "use-share-modify" ;
    void:sparqlEndpoint <http://affymetrix.bio2rdf.org/sparql> .
```

In the process of publishing data from Smart Cities as Linked Data, new ontologies are going to be created to model the particularities of each city. The creators of these ontologies have to publish a suitable documentation, allowing the proper reuse of them. A tool for publishing ontologies and their documentation is Neologism[34]. Neologism shows ontologies in a human-readable manner, representing class, subclass and property relationships through diagrams.

5.6 Linkage

Connecting existing data with other available resources is a major challenge for easing data integration. Due to its interlinked nature, Linked Data provides a perfect base to connect the data present in a given dataset.

The linkage stage starts a loop on the model after the publishing step, establishing relationships between existing data and external datasets, in order to provide links to new information stores.

Different frameworks have been developed to deal with class and properties matching. The basis of these frameworks is to provide data discovery features through links to external entities related to the items used in the analysis.

The *Silk - Link Discovery Framework* [52] offers a flexible tool for discovering links between entities within different Web data sources. Silk makes use of *Silk* -

[34] `http://neologism.deri.ie/`

Link Specification Language (Silk-LSL), a declarative language which lets data publishers specify which RDF link types should be discovered providing two related datasets, and the conditions under data items must fulfil to be interlinked. Silk framework's architecture is depicted in Figure 8. As an example, a script in Silk-LSL can be written to match cities between *DBpedia* ontology's *City* or *PopulatedPlace* classes, and *GeoName*'s feature class *gn:P*. As constraints, string similarity metrics can be used to match city names, and take into consideration cities' bounding boxes (i.e. the margins projected on a map) to check overlaps.

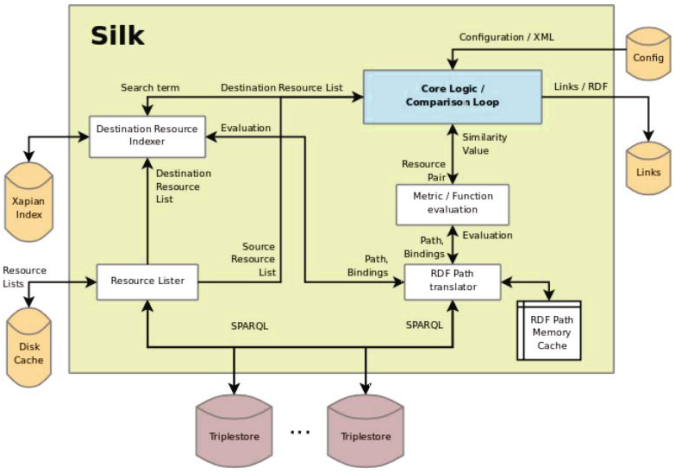


Fig. 8. Silk framework architecture

With a similar approach *LIMES* (LInk discovery framework for MEtric Spaces) [53] can be used for the discovery of links between Linked Data knowledge bases, focusing on a time-efficient approach especially when working with large-scale matching tasks. LIMES relies on *triangle inequality* mathematical principles for distance calculations, which reduce the number of comparisons necessary to complete a mapping by several orders of magnitude. This approach helps detecting the pairs that will not fulfil the requirements in an early stage, thus avoiding spending time in more time-consuming processing. The architecture followed by LIMES framework is depicted in Figure 9.

5.7 Exploit

At this stage, the focus is located on exploiting data for business-logic processes, should they involve data mining algorithms, analytics, reasoning, etc.

Whereas complex processing algorithms can be used independently of the dataset format, Linked Open Data can greatly help at reasoning purposes. Linked Open Data describes entities using ontologies, semantic constraints and

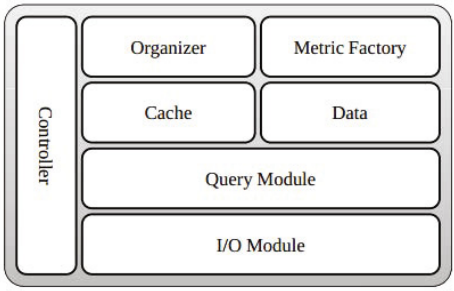


Fig. 9. LIMES framework architecture

restriction rules (belonging, domain, range, etc.) which favor the inference of new information from the existing one. Thanks to those semantics present in Linked Data, algorithms are not fed with raw values (numbers, strings...), but with semantically meaningful information (height in cm, world countries, company names...), thus, resulting in higher quality outputs and making algorithms more error aware (i.e. if a given algorithm is in charge of mapping the layout of a mountainous region, and founds the height of one of the mountains to be *3.45*, it is possible to detect the conversion has failed at some point, as height datatype was expected to be given in *meters*).

As seen in Sections 5.2 and 5.2, sensors and social networks are common data input resources, generating huge amounts of data streamed in real time. The work done in [54] comprises a set of best practices to publish and link stream data to be part of the Semantic Web.

However, when it comes to exploiting Linked Data streams, SPARQL can find its limits [55]. Stream-querying languages such as CQELS's[35] language (an extension of the declarative SPARQL 1.1 language using the EBNF notation) can greatly help in the task. CQELS [56] (Continuous Query Evaluation over Linked Stream) is a native and adaptive query processor for unified query processing over Linked Stream Data and Linked Data developed at DERI Galway.

Initially, a query pattern is added to represent window operators on RDF Stream:

```
GraphPatternNotTriples ::= GroupOrUnionGraphPattern |
OptionalGraphPattern | MinusGraphPattern | GraphGraphPattern |
*StreamGraphPattern* | ServiceGraphPattern | Filter | Bind
```

Assuming that each stream has an IRI as identification, the *StreamGraphPattern* is defined as follows:

```
StreamGraphPattern ::= 'STREAM' '['Window']' VarOrIRIref
'{'TriplesTemplate'}'
Window ::= Rangle|Triple|'NOW'|'ALL'
Range ::= 'RANGE' Duration ('SLIDE' Duration |'TUMBLING')?
Triple ::= 'TRIPLES' INTEGER
Duration ::= (INTEGER 'd'|'h'|'m'|'s'|'ms'|'ns')+
```

[35] https://code.google.com/p/cqels/

An example query could be:

```
PREFIX lv: <http://deri.org/floorplan/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

SELECT ?locName FROM NAMED <http://deri.org/floorplan/>
WHERE {
    STREAM <http://deri.org/streams/rfid> [NOW]
    {?person lv:detectedAt ?loc}
    {?person foaf:name "AUTHORNAME"^^xsd:string }

    GRAPH <http://deri.org/floorplan/>
    {?loc lv:name ?locName}
}
```

Eventually, both batch and streaming results can be consumed through *REST* services by web or mobile applications, or serve as input for more processing algorithms until they are finally presented to end users.

5.8 Visualise

In order to make meaning from data, humans have developed a great ability to understand visual representations. The main objective of data visualisation is to communicate information in a clean and effective way through graphical means. It is also suggested that visualisation should also encourage users engagement and attention.

The *"A picture is worth a thousand words"* saying reflects the power images and graphics have when expressing information, and can condense big datasets into a couple of representative, powerful images.

As Linked Data is based on subject-predicate-object triples, graphs are a natural way to represent triple stores, where subject and object nodes are interconnected through predicate links. When further analysis is applied on triples, a diverse variety of representations can be chosen to show processed information: charts, infographics, flows, etc.[57]

Browser-side visualisation technologies such as d3.js[36] (by Michael Bostock) and Raphaël[37] are JavaScript-based libraries to allow the visual representation of data on modern web browsers, allowing anybody with a minimum internet connection try to understand data patterns in a graphical form.

For developers not familiar with visualisation techniques, some investigations are trying to enable the automatic generation of graphical representations of Linked Data query results. The *LDVM* (Linked Data Visualisation Model) [58] is proposed as a model to rapidly create visualisations of RDF data. *LODVisualisation*[38] is an implemented prototype which supports the LDVM.

Visualbox[39] is a simplified edition of LODSPeaKr[40] focused on allowing people create visualisations using Linked Data. In Figure 10, a SPARQL query to

[36] http://d3js.org/
[37] http://raphaeljs.com/
[38] http://lodvisualisation.appspot.com/
[39] http://alangrafu.github.io/visualbox/
[40] http://lodspeakr.org/

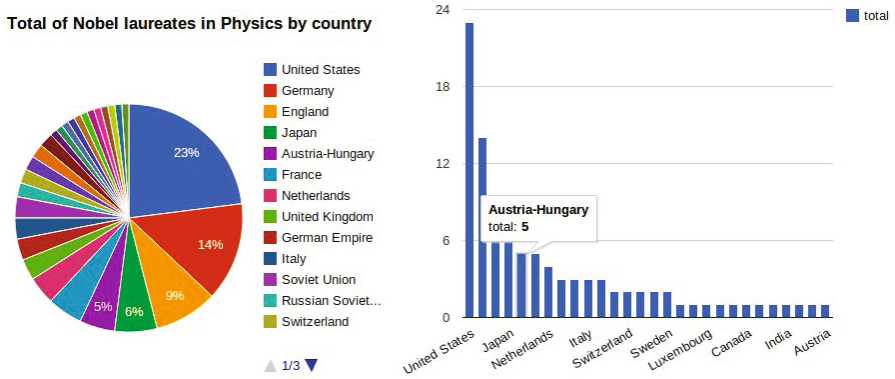


Fig. 10. Visualbox graph example (ownership: Alvaro Graves)

retrieve the number of Nobel laureates in Physics by country is displayed in interactive pie and bar charts.

Independently of how visualisations are generated, they provide a perfect solution to present high-quality, refined data to end-users.

6 Conclusions

In this chapter, we have proposed Linked Data as a suitable paradigm to manage the entire data life cycle in Smart Cities. As can be seen, along this chapter we expose a set of guidelines for public or private managers which want to contribute with data from their administration or enterprise into a Smart City, bringing closer existing tools and exposing practical knowledge acquired by authors while working with Linked Data technologies. The proposed data life cycle for Smart Cities covers the entire travelling path of data inside a Smart City, and the mentioned tools and technologies fulfil all the needed tasks to go forward on this path.

But Linked Open Data is not all about technology. The *Open* term of Linked Open Data is about the awareness of public (and private) administrations to provide citizens with all the data which belong to them, making the governance process more transparent; the awareness of developers to discover the gold behind data and the awareness of fully informed citizens participating on decision making processes: Smart Cities, smart business and Smart Citizens.

Urban Linked Data applications also empower citizens' role of first level data providers. Thanks to smartphones, each citizen is equipped with a full set of sensors which are able to measure the city's pulse at every moment: traffic status, speed of each vehicle to identify how they are moving, reporting of roadworks or malfunctioning public systems and so forth. Citizens are moving from data consumers to data *prosumers*, an aspect data scientists and application developers can benefit from to provide new services for Smart Cities.

Acknowledgments. This work has been supported by research project grants Future Internet II (IE11-316) and SmarTUR (IE12-343) granted by the Basque Government and ADAPTA (IPT-2011-0949-430000) by the Spanish Government. Mikel Emaldi and Jon Lázaro are grateful to University of Deusto for their PhD grants. Oscar Peña holds a PhD grant from the Basque Government.

References

1. World Health Organization: Urbanization and health. Bull World Health Organ. 88, 245–246 (2010)
2. Bizer, C., Boncz, P., Brodie, M.L., Erling, O.: The meaningful use of big data: four perspectives – four challenges. SIGMOD Rec. 40(4), 56–60 (2012)
3. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American 284(5), 28–37 (2001)
4. Bizer, C., Heath, T., Berners-Lee, T.: Linked data-the story so far. International Journal on Semantic Web and Information Systems (IJSWIS) 5(3), 1–22 (2009)
5. Initiative, D.D.: Overview of the DDI version 3.0 conceptual model (April 2008)
6. Ball, A.: Review of data management lifecycle models (2012)
7. Burton, A., Treloar, A.: Designing for discovery and re-use: the 'ANDS data sharing verbs' approach to service decomposition. International Journal of Digital Curation 4(3), 44–56 (2009)
8. Michener, W.K., Jones, M.B.: Ecoinformatics: supporting ecology as a data-intensive science. Trends in Ecology & Evolution 27(2), 85–93 (2012)
9. Auer, S., et al.: Managing the life-cycle of linked data with the LOD2 stack. In: Cudré-Mauroux, P., et al. (eds.) ISWC 2012, Part II. LNCS, vol. 7650, pp. 1–16. Springer, Heidelberg (2012)
10. deRoos, D., Eaton, C., Lapis, G., Zikopoulos, P., Deutsch, T.: Understanding big data: Analytics for enterprise class hadoop and streaming data. McGraw-Hill Osborne Media (2011)
11. Russom, P.: Big data analytics. TDWI Best Practices Report, Fourth Quarter (2011)
12. Li, X., Dong, X.L., Lyons, K., Meng, W., Srivastava, D.: Truth finding on the deep web: is the problem solved? In: Proceedings of the 39th International Conference on Very Large Data Bases, PVLDB 2013, pp. 97–108. VLDB Endowment (2013)
13. Buneman, P., Davidson, S.B.: Data provenance–the foundation of data quality (2013)
14. Emaldi, M., Pena, O., Lázaro, J., Lápez-de-Ipiã, D., Vanhecke, S., Mannens, E.: To trust, or not to trust: Highlighting the need for data provenance in mobile apps for smart cities. In: Proceedings of the 3rd International Workshop on Information Management for Mobile Applications, pp. 68–71 (2013)
15. Hartig, O., Zhao, J.: Using web data provenance for quality assessment. In: Proceedings of the International Workshop on Semantic Web and Provenance Management, Washington DC, USA (2009)
16. Bizer, C., Cyganiak, R.: Quality-driven information filtering using the WIQA policy framework. Web Semantics: Science, Services and Agents on the World Wide Web 7(1), 1–10 (2009)
17. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.G.: Dbpedia: A nucleus for a web of open data. In: Aberer, K., et al. (eds.) ISWC/ASWC 2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007)

18. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: Dbpedia-a crystallization point for the web of data. Web Semantics: Science, Services and Agents on the World Wide Web 7(3), 154–165 (2009)
19. Tummarello, G., Delbru, R., Oren, E.: Sindice.com: Weaving the open linked data. In: Aberer, K., et al. (eds.) ISWC/ASWC 2007. LNCS, vol. 4825, pp. 552–565. Springer, Heidelberg (2007)
20. Tummarello, G., Cyganiak, R., Catasta, M., Danielczyk, S., Delbru, R., Decker, S.: Sig. ma: Live views on the web of data. Web Semantics: Science, Services and Agents on the World Wide Web 8(4), 355–364 (2010)
21. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: A survey on sensor networks. IEEE Communications Magazine 40(8), 102–114 (2002)
22. Sanchez, L., Galache, J.A., Gutierrez, V., Hernandez, J., Bernat, J., Gluhak, A., Garcia, T.: SmartSantander: the meeting point between future internet research and experimentation and the smart cities. In: Future Network & Mobile Summit (FutureNetw 2011), pp. 1–8 (2011)
23. Le-Phuoc, D., Quoc, H.N.M., Parreira, J.X., Hauswirth, M.: The linked sensor middleware–connecting the real world and the semantic web. In: Proceedings of the Semantic Web Challenge (2011)
24. O'reilly, T.: What is web 2.0: Design patterns and business models for the next generation of software. Communications & Strategies (1), 17 (2007)
25. Maynard, D., Tablan, V., Ursu, C., Cunningham, H., Wilks, Y.: Named entity recognition from diverse text types. In: Recent Advances in Natural Language Processing 2001 Conference, pp. 257–274 (2001)
26. Sixto, J., Pena, O., Klein, B., López-de-Ipiña, D.: Enable tweet-geolocation and don't drive ERTs crazy! improving situational awareness using twitter. In: SMERST 2013: Social Media and Semantic Technologies in Emergency Response, Coventry, UK, vol. 1, pp. 27–31 (2013)
27. Martins, B., Anastácio, I., Calado, P.: A machine learning approach for resolving place references in text. In: Geospatial Thinking, pp. 221–236. Springer (2010)
28. Abel, F., Hauff, C., Houben, G.J., Stronkman, R., Tao, K.: Twitcident: fighting fire with information from social web streams. In: Proceedings of the 21st International Conference Companion on World Wide Web, pp. 305–308 (2012)
29. Vieweg, S., Hughes, A.L., Starbird, K., Palen, L.: Microblogging during two natural hazards events: what twitter may contribute to situational awareness. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1079–1088 (2010)
30. Hughes, A.L., Palen, L.: Twitter adoption and use in mass convergence and emergency events. International Journal of Emergency Management 6(3), 248–260 (2009)
31. Celino, I., Cerizza, D., Contessa, S., Corubolo, M., Dell'Aglio, D., Valle, E.D., Fumeo, S.: Urbanopoly – a social and location-based game with a purpose to crowdsource your urban data. In: Proceedings of the 2012 ASE/IEEE International Conference on Social Computing and 2012 ASE/IEEE International Conference on Privacy, Security, Risk and Trust, SOCIALCOM-PASSAT 2012, pp. 910–913. IEEE Computer Society, Washington, DC (2012)
32. Celino, I., Contessa, S., Corubolo, M., Dell'Aglio, D., Valle, E.D., Fumeo, S., Krüger, T.: UrbanMatch - linking and improving smart cities data. In: Bizer, C., Heath, T., Berners-Lee, T., Hausenblas, M. (eds.) Linked Data on the Web. CEUR Workshop Proceedings, CEUR-WS, vol. 937 (2012)

33. Braun, M., Scherp, A., Staab, S.: Collaborative semantic points of interests. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010, Part II. LNCS, vol. 6089, pp. 365–369. Springer, Heidelberg (2010)
34. Noy, N.F., McGuinness, D.L.: Ontology development 101: A guide to creating your first ontology. Stanford knowledge systems laboratory technical report KSL-01-05 and Stanford medical informatics technical report SMI-2001-0880 (2001)
35. Emaldi, M., Lázaro, J., Aguilera, U., Peña, O., López-de-Ipiña, D.: Short paper: Semantic annotations for sensor open data. In: Proceedings of the 5th International Workshop on Semantic Sensor Networks, SSN 2012, pp. 115–120 (2012)
36. Lefort, L., Henson, C., Taylor, K., Barnaghi, P., Compton, M., Corcho, O., Garcia-Castro, R., Graybeal, J., Herzog, A., Janowicz, K.: Semantic sensor network XG final report. W3C Incubator Group Report (2011)
37. Raskin, R.G., Pan, M.J.: Knowledge representation in the semantic web for earth and environmental terminology (SWEET). Computers & Geosciences 31(9), 1119–1125 (2005)
38. d'Aquin, M., Nikolov, A., Motta, E.: Enabling lightweight semantic sensor networks on android devices. In: The 4th International Workshop on Semantic Sensor Networks (SSN 2011) (October/Autumn 2011)
39. Emaldi, M., Lázaro, J., Laiseca, X., López-de-Ipiña, D.: LinkedQR: improving tourism experience through linked data and QR codes. In: Bravo, J., López-de-Ipiña, D., Moya, F. (eds.) UCAmI 2012. LNCS, vol. 7656, pp. 371–378. Springer, Heidelberg (2012)
40. Raimond, Y., Abdallah, S., Sandler, M., Giasson, F.: The music ontology. In: ISMIR 2007: 8th International Conference on Music Information Retrieval, Vienna, Austria, pp. 417–422 (September 2007)
41. Weibel, S., Kunze, J., Lagoze, C., Wolf, M.: Dublin core metadata for resource discovery. Internet Engineering Task Force RFC 2413, 222 (1998)
42. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: Proceedings of the 16th International Conference on World Wide Web, pp. 697–706. ACM (2007)
43. Stasch, C., Schade, S., Llaves, A., Janowicz, K., Bröring145, A.: Aggregating linked sensor data. Semantic Sensor Networks, 46 (2011)
44. Ayers, A., Völkel, M.: Cool uris for the semantic web. Woking Draft. W3C (2008)
45. Bizer, C., Schultz, A.: The berlin sparql benchmark. International Journal on Semantic Web and Information Systems (IJSWIS) 5(2), 1–24 (2009)
46. Bizer, C., Cyganiak, R.: D2r server-publishing relational databases on the semantic web. In: Proceedings of the 5th International Semantic Web Conference, p. 26 (2006)
47. Gil, Y., Artz, D.: Towards content trust of web resources. Web Semantics: Science, Services and Agents on the World Wide Web 5(4), 227–239 (2007)
48. Hartig, O.: Provenance information in the web of data. In: Proceedings of the WWW 2009 Workshop on Linked Data on the Web, LDOW 2009 (2009)
49. Moreau, L., Clifford, B., Freire, J., Futrelle, J., Gil, Y., Groth, P., Kwasnikowska, N., Miles, S., Missier, P., Myers, J., Plale, B., Simmhan, Y., Stephan, E., Van den Bussche, J.: The open provenance model core specification (v1.1). Future Generation Computer Systems 27(6), 743–756 (2011)
50. Belhajjame, K., B'Far, R., Cheney, J., Coppens, S., Cresswell, S., Gil, Y., Groth, P., Klyne, G., Lebo, T., McCusker, J., Miles, S., Myers, J., Sahoo, S., Tilmes, C.: PROV-DM: The PROV data model (2013)

51. De Nies, T., Coppens, S., Mannens, E., Van de Walle, R.: Modeling uncertain provenance and provenance of uncertainty in W3C PROV. In: Proceedings of the 22nd International Conference on World Wide Web Companion, Rio de Janeiro, Brazil, pp. 167–168 (2013)
52. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Silk-a link discovery framework for the web of data. In: Proceedings of the International Semantic Web Conference 2010 Posters & Demonstrations Track. Citeseer (2009)
53. Ngomo, A.C.N., Auer, S.: Limes: a time-efficient approach for large-scale link discovery on the web of data. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, vol. 3, pp. 2312–2317. AAAI Press (2011)
54. Sequeda, J., Corcho, O., Taylor, K., Ayyagari, A., Roure, D.D.: Linked stream data: A position paper. In: Proceedings of the 2nd International Workshop on Semantic Sensor Networks (SSN 2009) at ISWC 2009. CEUR Workshop Proceedings, vol. 522, pp. 148–157 (November 2009)
55. Della Valle, E., Ceri, S., van Harmelen, F., Fensel, D.: It's a streaming world! reasoning upon rapidly changing information. IEEE Intelligent Systems 24(6), 83–89 (2009)
56. Le-Phuoc, D., Dao-Tran, M., Xavier Parreira, J., Hauswirth, M.: A native and adaptive approach for unified processing of linked streams and linked data. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 370–388. Springer, Heidelberg (2011)
57. Khan, M., Khan, S.S.: Data and information visualization methods, and interactive mechanisms: A survey. International Journal of Computer Applications 34(1), 1–14 (2011)
58. Brunetti, J.M., Auer, S., García, R.: The linked data visualization model. In: International Semantic Web Conference (Posters & Demos) (2012)
59. Cyganiak, Richard and Reynold, Dave. The RDF Data Cube Vocabulary (2013), http://www.w3.org/TR/2013/CR-vocab-data-cube-20130625/

Benchmarking Internet of Things Deployment: Frameworks, Best Practices, and Experiences

Franck Le Gall[1], Sophie Vallet Chevillard[2], Alex Gluhak[3], Nils Walravens[4], Zhang Xueli[5], and Hend Ben Hadji[6]

[1] Easy Global Market
Business Pôle
1047 route des Dolines
06901 Sophia Antipolis Cedex, France
franck.le-gall@eglobalmark.com
[2] Inno TSD
Place Joseph Bermond - Ophira 1
06902 Sophia-Antipolis
France
s.valletchevillard@inno-group.com
[3] University of Surrey, Guildford, Surrey, GU2 7XH, United Kingdom
a.gluhak@surrey.ac.uk
[4] iMinds-SMIT, Vrije Universiteit Brussel,
Vrije Universiteit Brussel, Pleinlaan 2, 1050 Elsene, Belgium
nils.walravens@vub.ac.be
[5] Chinese Academy of Telecom Research
No.52, Hua Yuan Bei Road, Haidian District, Beijing, China
Post Code: 100191
zhangxueli@catr.cn
[6] Centre d'Etudes et des Recherches des Télécommunications
Parc Technologique
EL GHAZALA ARIANA 2088
Tunisia
hend.benhji@cert.mincom.tn

Abstract. IoT deployments generate data of the real world in an automated fashion without direct user involvement. With increasing scale of these IoT deployments the extraction of the right knowledge about the real world from a vast amount of IoT data and efficient decisions is a challenging endeavor. While solutions to deal with large amounts of IoT data are slowly emerging, potential users of IoT solutions, or policy makers find it difficult to assess the actual usefulness of investing in IoT deployments or selecting adequate deployment strategies for a particular business domain.

Despite the recent hype generated by consultancy companies and IoT vendors about the IoT, there is still a lack of experience in assessing the utility and benefits of IoT deployments as many IoT deployments are still in their early stages. In order to capture such experience quicker and derive best practices for IoT deployments, systematic tools, or methodologies are required in order to

F. Xhafa et al. (eds.), *Modeling and Processing for Next-Generation Big-Data Technologies*,
Modeling and Optimization in Science and Technologies 4, DOI: 10.1007/978-3-319-09177-8_19

allow the assessment of the goodness or usefulness of IoT deployments and a comparison between emerging IoT deployments to be performed.

This chapter addresses the existing gap and proposes a novel benchmarking framework for IoT deployments. The proposed framework is complementary to the emerging tools for the analysis of big data as it allows various stakeholders to develop a deeper understanding of the surrounding IoT business ecosystem in a respective problem domain and the value proposition that the deployment of an IoT infrastructure may bring. It also allows a better decision making for policy makers for regulatory frameworks.

1 Introduction

Tackling global challenges is considered nowadays as an important driver of Information and Communication Technologies (ICT) development policies, research programs, and technological innovations. Emerging ICT technologies such as the Internet of Things (IoT) provide the promise of alleviating or solving many of our planets problems in areas as resource shortage and sustainability, logistics and healthcare, by an efficient integration of the real world with the digital world of modern computer systems on the Internet. The underlying vision is that they will enable the creation of a new breed of so-called smart services and applications, able to make existing business processes more effective and efficient and the delivery of service more personalized to the needs and situation of individual users and the society at large.

During the recent years, IoT has undergone a vast amount of research and initiatives driven by individual organizations to experiment the potentialities of IoT while trying to gain a 'large' footprint on the envisioned large marker. This conducted in the creation of many experiments either driven by research consortia or large industries. While technologies are becoming mature, recent analyzes demonstrated a number of shortcomings:

- Individual expectations do not fit together, creating conflict of interest between new entrants to the market and former business whose business models are being challenged and need to be reinvented.
- The end expectations of the beneficiaries are not yet known or identified. The balance between perceived value and willingness to pay. The IoT offer will not succeed as long as it is offered as a product or as a technology service and needs to move beyond the technological performance.

The IoT Business ecosystem is, thus, a complex territory in which multiplicity of stakeholders profiles are linked through diversified value chains. The complexity is impacting the large-scale deployment of IoT, not happening today beyond experimentation projects, due to the lack of integration of the stakeholders within a clear and comprehensive ecosystem.

Defining the Internet of Things (IoT) has mobilized many resources over the past years without yet providing one largely adopted definition. Argumentations are still flying around regarding what is or should not be considered as part of the internet of things. The situation and frontiers of IoT is quite diverse in the different parts of the world, and the way IoT is deployed pursues different paths.

Nevertheless, there is consensus on the wideness of the IoT over many of its characteristics:

- **Number of devices**: ranging from some hundreds to thousands, the number of devices to be connected in IoT deployments will explode[1].
- **Technologies**: devices to be connected to the IoT are not only numerous but also encompass many technologies needed to add sensing, actuating, communicating, managing, etc. capabilities to the IoT.
- **Expectations**: IoT is tightly coupled to many concepts including SmartCity, SmartTransport, SmartEnergy, SmartFarming, etc., all being representatives of sectors hoping for increased effectiveness and efficiency by adding ICT supported sensing and actuating capabilities to their environment
- **Stakeholders**: Internet of Things is gathering representatives from diversified sectors, including industries which have not yet been connected to the Internet. In many cases, a move of the ICT industry toward historical sectors is observed in conjunction with a move of historical sectors toward the Internet of Things. This leads to a large increase of involved stakeholders.
- **Data**: Internet of Things relies not only on sensing and actuating capabilities of devices providing physical measures such as temperature, humidity, light, vibration, location, movement, etc. but also on information related to the users's profiles and preferences as well as information related to the local context (such as events agenda in a city). This creates huge amounts of data to be managed.

Several issues hamper large-scale deployment of IoT on both technical and nontechnical dimensions. The table below list some of the issues as they are documented in IoT related literature [2][3][4]:

Table 1. Issues to deploy IoT in large scale

Technical	Non technical
• Communication issues • Protocols • Data exchanges • Data processing and management • Interoperability • Privacy and security issues for the users • Performance • Shelf life of the device • Etc.	• Business model • Governance • Interoperability • Privacy and security issues for the users • New usages • Etc.

[1] "The Internet of Things will include 26 billion units installed by 2020. IoT product and service suppliers will generate incremental revenue exceeding $300 billion, mostly in services, in 2020", according to Gartner analysts [1].

Consequently, moving to larger scale requires a better understanding and assessment of the actual benefits of IoT deployments looking at the whole corresponding business ecosystem.

In order to understand a sector, there is a need to build the overall business ecosystem, highlighting the dependencies and respective positioning of the stakeholders involved. In addition, a business ecosystem needs to include externalities such as the regulatory framework. Investopedia defines the business ecosystem as “The network of organizations – including suppliers, distributors, customers, competitors, government agencies and so on – involved in the delivery of a specific product or service through both competition and cooperation. The idea is that each business in the "ecosystem" affects and is affected by the others, creating a constantly evolving relationship in which each business must be flexible and adaptable in order to survive, as in a biological ecosystem”. There are many ways to represent business ecosystems and each business of an ecosystem is tempted to draw the map of the ecosystem as seen from its perspective.

Consequently, moving to larger scale requires a better understanding and assessment of the actual benefits of IoT deployments regarding several aspects. It is, thus, required to assess if the deployment achieves what it is supposed to, and also if it is the best way to achieve such a goal. In addition, it is also a beneficial experience to learn from other if any unexpected issues appeared in these deployments and if sustainability can be maintained.

Providing answers to these questions is of interest to people involved in IoT deployments to identify good practices, avoid traps, and make right choices. To be efficient, an analysis of deployments based on real field experience is needed, this will be done through examining a multitude of technical and socioeconomic dimensions.

This chapter proposes the development and assessment of a benchmarking framework for IoT deployments which pursues this goal: providing stakeholders involved in an IoT deployment with a decision-making tool, which will provide them with complete vision of existing IoT deployments.

2 Preliminary Concept

2.1 Conceptual Model of Public Policy Evaluation

Smart city concept and the challenges encountered to deploy it in real-field show similarities with a public policy or program as it is a multi-stakeholders process which involves a public decision, operators, and beneficiaries, each may have orthogonal expectations, and various involvement in the different stages of the process. Another element that makes the analogy significant is the long duration required to get realized the impacts corresponding to long-term social and environmental expectations.

The perspective of analysis followed by this work to characterize smart cities deployments aims at encompassing the global dimensions of the smart cities. The main

focus of the analysis relates to the added value produced by the implementation of such a concept for the various stakeholders involved, which also is equivalent to analyse the production of outputs-outcomes-impacts in these cases.

The classical Patton evaluation framework [5] has been adapted for the smart cities case. The following paragraphs explain the main concepts of public policies evaluation and express their mapping with smart cities components.

Public policy is defined through its programming cycle based on three main steps:

- **Policy design** which consists of the identification of the public (or societal) problem that needs to be addressed and defining the objectives of the policy;
- **Policy implementation** which consists of the implementation and operational inputs to achieve objectives. Inputs can come from various sources: resources (human and financial), but also physical facilities, legal aspects, etc.
- **Policy completion** which consists of the production of the outcomes and impacts which are expected to match the original objectives.

The transposition of these concepts to smart cities is straighforward:

- Equivalent to **policy design** is the definition of the general purpose of the smart city. This covers a large number of societal expectations such as energy, health, inclusion, climate change, and all the great challenges the European Union want to fix through its programmes and policy. These objectives are subdivided in several specific objectives. Taking the example of the climate change priority, the overall objective of the policy could be translated in the decreasing of the carbon footprint. The actions undertaken to achieve this are large and cover several parts: energy efficiency, mobility, etc.
- Equivalent to the **implementation policy** is the process of deploying: deployment of sensors, platforms, applications, services. But also nontechnical inputs such as regulations and legal capabilities, human, and financial resources and many more. Following the above example, on the topic of energy efficiency, the implementation could be split in a diversity of actions, each of them implying dedicated realisations:
  - refurbishing public buildings;
  - incentivizing renovation of private buildings where possible;
  - improving public transport and overall urban transport management;
  - increasing energy efficiency of public lighting;
  - etc.

  Defining these actions is the heart of the policy design and is operationalized into specifications that the deployments should answer.
- Finally, equivalent to the **policy completion** is the achievement of impacts expected to be induced by deployment of the smart city components (for instance based on IoT) and its applications and services. As for a public policy or program, an IoT deployment is here considered as a mean to produce intended impacts.

The figure below emphasizes this translation.

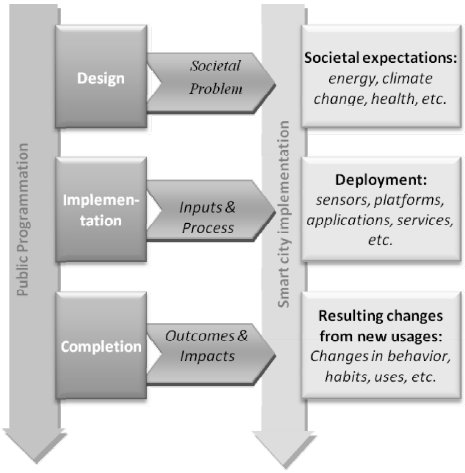


Fig. 1. Smart city implementation as a public policy

Considering a smart city as a public policy enables then to use the classical evaluation tools and methods to observe, monitor, and assess it. Evaluations methods have been proposed and improved over the years. They consist of examining the overall consistence of the policy cycle and the extent of matches between the preliminary logic intervention and the observed outcomes and impacts. They allow highlighting the casual hypotheses done under intervention logic which are often unclear and unshared among stakeholders.

Conceptually, as shown on the Fig. 2, evaluation is based on six evaluation criteria: (1) Effectiveness which consists of analysing to what extent the real observed outcomes and impacts corresponding to the objectives – i.e., the primary expected impacts - and in highlighting the unexpected effects of the implementation of the policy or programme, (2) Efficiency which consists of analysing to what extent the "amount" of observed effects is consistent with the inputs: could the policy or program have the same effects with less input? Or inversely, could the policy have more effects with more inputs? (3) Relevance consists of asking or re-asking the logic of the intervention: Are the done hypothesis on causal relationships consisting in answering the needs and challenges, or some of them are false and should be removed? (4) Utility consists of analysing the nature of the observed effects regarding the social needs and demand. This criterion is seldom considered into evaluations mandated by policy makers, as the existence of the policy itself is seldom questioned, (5) Coherence consists of analysing the consistence of inputs and implementations regarding the objectives of the programme or policy: are the realizations and actions undertaken in line with the objectives or have critical aspects been missed that impede the correct implementation of the programme?

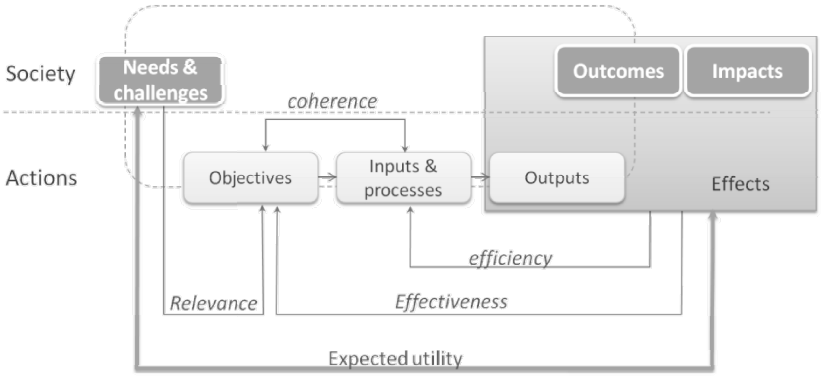


Fig. 2. Evaluation criteria in the context of a public policy/program, adapted from Means [6]

This conceptual framework has been used as a starting point to build our own "benchmarking framework" containing the dimensions, criteria, and indicators (or metrics) needed to characterize the IoT deployments in the context of smart cities. Modeling IoT deployment in such a way has the benefit to include the external context of IoT deployment along its entire lifecycle: (1) upstream regarding the challenges and objectives expected to be achieved, (2) downstream regarding the outcomes and impacts (even unpredicted) it produces.

It is something new in the context of technological deployment where evaluations are mainly based on technological performance and improve the technical characteristics to do better in a more efficient way or at lower cost for instance.

A key element in the development toward smarter cities is that the whole value network of actors and stakeholders is involved in the creation of new service offerings in cities. This cooperation is often referred to as value network or ecosystem analysis [7], but what is actually at stake is the creation of business models for these new services, that incorporate an active role for public bodies, rather than simply assuming a purely commercial logic. Looking at the overall dimensions and not only the technological one offers a new perspective of analysis based on a systemic approach which include the diverse and sometime diverging interest of involved stakeholders, including public authorities seeking for an harmonious and sustainable development of their city. Beyond technological challenges, all of these aspects are of paramount importance to allow a transfer from technology development to market exploitation in a sustainable way.

Therefore, the development of a benchmarking framework for IoT deployments in the framework of smart cities requires to go beyond pure technological approach to include the business ecosystem dimension, including the specific role of public authorities. In order to understand business models related to IoT deployments in smart cities, we need to think about structured approaches to provide answers for the diverse and complex questions companies, citizens, and governments face there. Related considerations are detailed in the following paragraph, starting from the origins of business modeling literature.

3 Methodology

Defining metrics to be captured is the basis of any benchmarking framework which needs comparable dataset to build the comparative analysis. Before realizing the comparative exercise of a benchmark, there is, thus, a need to get the standalone measure of the deployment under evaluation.

In the present framework, the specific interest is on Internet of Things industrial deployments. Such deployments exhibit diverse and broad characteristics. It is, thus, necessary to structure the information to be collected.

The preliminary temptation is to propose a benchmarking framework based on a holistic model that could serve different purposes. Each person willing to develop a benchmark could then select a subset of the proposed metrics, depending on its own intentions. This section proposes a classification for the metrics together with their sublevels.

After defining the overall approach for the benchmarking framework and the nature of targeted indicators (objectives, input, process, and output) relevant to the evaluation, there is a need to identify the areas of interest of the benchmark, later on referred as benchmarking dimensions.

To identify these dimensions, the proposed work started from the STEEPLE analysis framework: the PEST (Political, Economic, Social, and Technological) framework is a traditional approach for the strategic analysis of a macro-environment. One of its recent extensions is known as STEEPLE covering the Social, Technological, Economic, Environmental, Political, Legislative, and Ethical dimensions. This framework is classically used to characterize the external influences of a business system and has been retained as a source of inspiration when building the benchmarking framework. In addition recent work of the 'European Research Cluster on the Internet of Things' acknowledged the contribution of standardization and testing to increase interoperability of IoT technologies and thus, accelerate their industrial deployment.

While defining the framework, after preliminary interviews, it rapidly appeared that the overall benchmarking was closely related to the overall business ecosystem of IoT deployments. Since the general adoption of this concept in the literature related to the rise of internet-based e-commerce [8], the focus of business modeling has gradually shifted from the single firm to networks of firms, and from simple concepts of interaction or revenue generation to extensive concepts encompassing the value network, the functional architecture, the financial model, and the eventual value proposition made to the user [9][10]. In attempt to capture these various elements, one approach has been to consider business modeling as the development of an unambiguous ontology that can serve as the basis for business process modeling and business case simulations [11][12]. This corresponds with related technology design approaches [13] aimed at the mapping of business roles and interactions onto technical modules, interfaces, and information streams. Due to the shifting preoccupation from single-firm revenue generation towards multi-firm control and interface issues, the guiding question of a business model has become "Who controls the value network and the overall system design" just as much as "Is substantial value being produced by this model (or not)" [14].

Based on the tension between these questions, Ballon proposes a holistic business modeling framework that is centered around control on the one hand and creating value on the other. It examines four different aspects of business models: (1) the way

in which the value network is constructed or how roles and actors are distributed in the value network, (2) the functional architecture, or how technical elements play a role in the value creation process, (3) the financial model, or how revenue streams run between actors and the existence of revenue sharing deals, and (4) the value proposition parameters that describe the product or service that is being offered to end users. For each of these four business model design elements, three underlying factors are important, which are represented in Fig. 3.

CONTROL PARAMETERS				**VALUE PARAMETERS**			
Value Network Parameters		**Functional Architecture Parameters**		**Financial Model Parameters**		**Value Configuration Parameters**	
Combination of Assets		Modularity		Cost (Sharing) Model		Positioning	
Concentrated	Distributed	Modular	Integrated	Concentrated	Distributed	Complement	Substitute
Vertical Integration		Distribution of Intelligence		Revenue Model		User Involvement	
Integrated	Disintegrated	Centralised	Distributed	Direct	Indirect	High	Low
Customer Ownership		Interoperability		Revenue Sharing Model		Intended Value	
Direct	Intermediated	Yes	No	Yes	No	Price/ Quality	Lock-in

Fig. 3. Business Model Matrix [12]

Each of the parameters in the business model matrix is explained in more detail in the Table 3.These parameters discuss the various important parts required in understanding ICT-related business models.

Using these inputs, five dimensions have been proposed to characterize the IoT deployments, in addition to two transversal one providing the general description of the deployment and its overall appreciation[2]. These five dimensions are:

- **Technology**: Internet of Thing deployments first rely on technologies to be deployed and to interoperate. These range from the physical layer up to the applications one. Within this dimension, the benchmarking framework aims at characterizing the infrastructure, deployed services, and data-related metrics. In addition, use of interoperable and thus standardized technologies is acknowledged as a key success factor and included within the analysis.
  The main goal of the technology dimension of the benchmark framework is twofold: (1) To measure the technological development level in terms of IoT technologies, (2) To provide metrics that can be useful to identify best practices for IoT

[2] This framework has been produced on behalf of the PROBE-IT project and are documented into project deliverables.

deployments. As the set of technologies related to IoT is broad, the framework categorizes the problem domain into the following four IoT technology dimensions:

- **IoT services and application** - that leverage an underlying IoT infrastructure in order to improve the effectiveness and efficiency of operation of existing services in an organization or the introduction of novel services that can be enabled by leveraging the IoT.

Table 2. Parameters of the Business Model Matrix

Value network *The combination of assets*: Assets include anything tangible or intangible that could be used to help an organization achieve its goals; this element also focuses on the synergetic effects of combining different assets. *The level of vertical integration*: Vertical integration is defined as the level of ownership and control over successive stages of the value chain. *Customer ownership*: This topic looks into the party maintaining the customer relationship and keeping the customer data. Related to customer ownership is the level of openness/lock-in of the case.	**Financial model** *Investment structure*: This topic deals with the necessary investments (both capital expenditures and operational expenditures) and the parties making these investments. *Revenue model*: This topic deals with the tradeoff between direct/indirect revenue models as well as the tradeoff between content-based and transport-based revenue models. *Revenue sharing model*: The revenue sharing model refers to agreements on whether and how to share revenues among the actors involved in the value chain.
Functional architecture *Modularity/integration*: Modularity refers to the design of systems and artefacts as sets of discrete modules that connect to each other via predetermined interfaces. *Distribution of intelligence*: In ICT systems, this refers to the particular distribution of computing power, control, and functionality across the system in order to deliver a specific application or service. *Interoperability*: Interoperability refers to the ability of systems to directly exchange information and services with other systems, and to the interworking of services and products originating from different sources.	**Value proposition** *Positioning*: Positioning of products and services refers to marketing issues including branding, identifying market segments, establishing consumer trust, identifying competing products or services, and identifying relevant attributes of the product or service in question. *User involvement*: Refers to the degree to which users are consumers rather than prosumers (referring to people being both producer and consumer of content at the same time) of content and services. *Intended value*: This item lists the basic attributes that the product or service possesses, or is intended to possess, and that together constitute the intended customer value.

- **IoT infrastructure** – the underlying hardware and software infrastructure that will be deployed in order to support IoT services and applications in a problem domain. Examples of Hardware include sensor and actuator nodes, gateway devices, RFID tags, and readers, etc
- **IoT data** – relates to the underlying information generated by the IoT infrastructure and consumed and further processed by IoT services and applications.
- In addition, a transversal dimension taking into the position toward **standardization** is developed.

Overall, the technology part of the framework defines a broad range of metrics that holistically capture useful insights from the above technology scopes. These indicators can be categorised in the following dimensions: Advancement metrics (process), Accessibility/ Coverage metrics (process), Compliance metrics (output), Performance metrics (output), Metrics for Openness (objective & output), Usage metrics (output).

- **Economy**: This dimension includes all income and expenses metrics related to the IoT deployments. Beyond financial data, local governance model used need to be described. Political factors related to public intervention in the economy are thus included with the discussion. In addition, IoT is expected to drive innovative usages leading to new business models to be captured within the economy dimension analysis.

  Economical and financial aspects are of paramount importance in the context of the IoT. Indeed, as this domain (and more general the Future Internet field) is still in an emerging stage, the question on value creation and the "who is paying for what" question is an open debate. The example of social networks that are still looking for business model illustrates the kind of challenges around this question.
  In the present benchmarking framework perspective, IoT deployments enclose several aspects:
  - The multi-stakeholders and multi-dimensions facets of IoT deployments complicate the share of costs and value creation. Several issues appear regarding for instance the intellectual property, the development and experimentation cost. IoT is developed in an *open innovation* context based on the concept of "innovating with partners by sharing risk and sharing reward". However, the risk perception and expected rewards could differ drastically between stakeholders and make more complex the definition of business model.
  - Regarding the deployment itself, the scale of deployment (for instance in entire cities) could not only be problematic with respect to the cost of infrastructure investment, end devices deployment, and so on, but also the maintenance and exploitation costs.
  - The risk management should also be taken into consideration. In fact, deployments in the scope of PROBE-IT are based on large scale whereas the proof of concept is not certain.
  - Finally, the question of data is nowadays at the heart of concerns. The objectives in this aspects are opposite between private and public word, the second one pushing for opening the data, whereas for the second one, the data access and control are at the heart of the value.

  Regarding the benchmarking framework, interests about economical and financial aspects are thus based on: identifying partnerships and share of cost and revenue, measuring the profitability and economic performance (or prosperity) regarding the

expectations of the stakeholders, providing metrics to identify best practices on these aspects for the different stakeholders.

- **Socio-environmental**: Social and environmental aspects play an important role in IoT deployments because of the strong connexion between IoT and social or environmental challenges.

  In many cases, the link is straightforward: social and/or environmental aspects are an aim in itself of the IoT deployment. It is for instance the case of smartcities, which pursue social achievements (regarding aging population, health, etc.) or environmental goals (like reducing environmental footprint of their activities). It is thus needed to follow and evaluate to what extent the deployments contribute to achieve these promises.

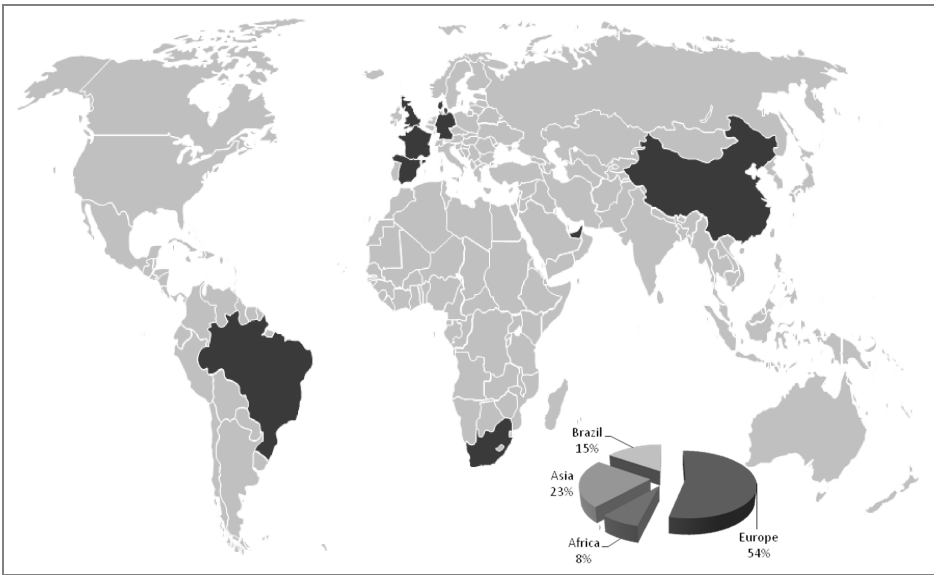
  But these good intentions are still in the stage of causal relationship hypothesis on the way the IoT deployment will produce these expected impacts. The problems IoT deals with are complex and it is hard to implement and foresee all the aspects. Many factors could question these hypotheses insofar as most of the time there are based on or implied changes in user's behavior and user's acceptance: a correct deal has to be find to respect needs and will of the users and the usage that is expected from them, otherwise they will reject the entire system. The recent example of the deployment of NFC technology in the transport system of Paris encounters difficulties because of the lack of transparencies regarding the data collection and the tracking of users.

  Finally, environmental aspects have also to be taken into consideration as negative effects of the deployment for instance in energy consumption, waste production, etc. As an example, one can mention the example of the RFID tags whose copper contained in the antenna adversely polluted the glass recycling streams and imposed specific recycling treatment to RFID tags. Nowadays concerns on climate impose to take into account from the beginning of the lifecycle of a product the question of its destruction and recycling. If smartcities that aim to reduce their environmental impacts are to be coherent, these aspects could become choice criteria.

  The unexpected impact concerning directly the user constitute a dedicated dimension relative to "human factors".
- **Legal and regulation**: IoT deployments mix many technologies, many stakeholders, including citizens as main users and beneficiaries. Any deployment thus needs to comply with a large number of legal requirements including consumer protection, health and safety, radio spectrum management, etc. Going to real field IoT deployment requires paying a close attention to the corresponding legal framework. This framework includes obviously the regulation to be followed and the partnership rules set between the stakeholders of the IoT deployment, including end users agreements. In some cases, certification as a proof to conformance to regulation may be required.
- **Human factors**: Ethical issues, mostly related to data privacy are a major concern when deploying IoT. Beyond users' protection, important in countries such as the European one, these issues relate to the overall user acceptance of the technology that also needs to be evaluated. But on the other side, an increased user's satisfaction can lead to a positive innovation loop where new usages, possibly based on crowd sourcing, are created.

4 Current Situation for IoT Deployments in Smart Cities

A subset of the developped taxonomy has been used to analyze 12 IoT deployments around the world (Europe, Africa, Asia, and Latin America). These deployments have been chosen for their maturity and their smart city orientation. Most of those deployments operate at a citywide scale and provides a variety of smart services (energy savings, light monitoring, etc.). Fig. 4 portrays the smart cities participating in the study, showing their geographical distribution.



Asia:	**Europe:**
Beijing city (China)	Aarhus city (Danmark)
Hengqin city (China)	Barcelona city (Spain)
Abu-Dhabi city (UAE)	Nice city (France)
Africa:	Birmingham city (England)
Johannesburg city (South Africa)	Manchester city (England)
Latin America:	Santander city (Spain)
Aparecida city	Karlsruhe city (Germany)
Rio de janeiro city	

Fig. 4. Geographical distribution of smart cities participating in the study

Based on the maturity of actual deployments and related services and the data available, three areas – technology, economy, and human factors were explored. Socio-environmental dimension and legal aspects could not be in-depth considered as no or poor information was available on these aspects. For each case, literature research and interviews has been conducted. Data collected was synthesized into a common matrix illustrated in the Table 4.

Table 3. Matrix of analysis

	Dimensions metrics			
General	**Country\City**	**Objectives**	**Application domain**	**Services**
		Examples *- Social exclusion* *- Economic re-structuring through the use of IoT technologies.*	*Examples :* *Smart city, Water management, Smart grid*	*Examples :* *City dashboard, air quality monitoring, smart meter in public building, application on public transport system, etc.*
Technology	**IT infrastructure**	**IoT infrastructure**	**IoT data**	**Interoperability & standards**
	Characteristics regarding : - Access Network - Digital hubs. - Backbone networks	**Status:** *Experimentation, living lab, pilot project, production* **Scale:** *from small to large* **Maturity:** *from early stage to actual implementation .*	**Data strategies**: *Example: Open data platform which aims to turn capabilities offered by accessible public data to services and solutions.*	**Interoperability:** *position and strategy* **Standards:** *position and use*
Economy	**Funding**	**Expenses**	**Revenues**	**Risk management**
	Type of funding: *Example: Public-Private fund.* **Funding sources:**	**Nature of expenses** *Example: Investments in new digital infrastructures and services.*	**Business opportunity perceived:** *Example: Opportunity to private sector to create innovative smart city services on public data sharing and create new revenue stream.*	
Human factors	**Data policy**	**User acceptance**	**Expected impacts**	
	Example : Data policy issues, Existence of an Ethical committee		**- For End-user/citizens** **- For Business stakeholders** **- For Government**	

Looking through the 12 smart city cases, it is possible to discern three high-level objectives that articulate the **smart city vision**, each of which addresses specific challenges:

- **Societal improvement**: Some cities would like to make better life for their citizens and they are primarily interested in leveraging IoT technologies to address the frustrations of daily life and improve citizens well being. These cities are increasingly looking to improve the effectiveness and the efficiency of cities public services like education, healthcare, public safety, transportation, utilities, etc.

  This is exemplified by the smart city agenda of Birmingham, where the city's objective is to provide citizens with better quality of life and economic prosperity. Other examples include Barcelona and Johannesburg, where the challenge is the transport congestion, improvement of public transport services is a building block of the smart city agendas.
- **Economic growth:** Other cities create a high quality of life and robust city infrastructure with the aim to be a business hub that attract businesses and employees to their areas and create new business and employment opportunities. The attractiveness and reputation given by this kind of smart city projects help cities move toward global competitiveness.

  This was clearly the case in Johannesburg, Barcelona and Hengqin cities. They are building an advanced ICT infrastructure to attract companies and investors. Other cities, such as Santander, created a testbed for smart city services and IoT technologies focused on open data and made them available to attract businesses and stimulate the innovation in technologies based on real-time data on a city's infrastructure.
- **Environmental sustainability:** Most of cities share a set of challenges related to environmental sustainability such as the increasing needs of energy, pollution, & waste production, etc. They are under the pressure to use energy more efficiently and improve the environment through lower pollution and carbons emissions.

To cope with the environmental challenges, cities are looking to use IoT technologies to increase the efficiency and effectiveness of key municipal services, such as waste and water management and street lighting, and to monitor cities' progress toward climate change mitigation. The best example of this is Amsterdam city, where reducing energy consumption and more efficient energy usage were the key objectives for the initiation of the smart city project.

It is worthwhile mentioning that the three high-level objectives are not exclusive of each other. They are all major reasons behind the initiation of smart city projects. They do not exclude that in a specific smart city context another objective may be present, but considered less important.

4.1 Linking High-Level Objectives of Smart Cities with Stakeholders' Expectations

While many cities share the three high level objectives, presented above, there is no single definitive way in which all stakeholders behave and work together. The priority of smart city projects varies based on the role of each stakeholder.

The study of smart city cases revealed a variety of players involved in the development of smart city projects: the mayor, the city council, the city municipalities, the utilities authority, the service providers, the network operator, the networking equipment suppliers, etc. This large ecosystem of stakeholders can be classified into four main roles: (1) Policy makers (e.g., city council, city municipalities, top level Government officals, etc.), (2) Operator, service providers (e.g., city services, etc.), (3) Technology providers (e.g., big industries, innovative SME, etc.), (4) Users (e.g., citizens), confirming the preliminary assumptions made in the benchmarking framework [15].

Table 4. Linking high-level objectives of smart cities with stakeholders' interests

Stakeholders	High-level objectives					
	Societal		Economic		Environmental	
	Citizen well-being	Good Governance (services, communication, etc.)	Innovation & city attractiveness	Creation of new business / Business transformation	Energy saving	Resources monitoring
Policy makers	✓	✓	✓	✓	✓	✓
Technology/Service providers			✓	✓		
Users	✓	✓			✓	

We draw Table 4 which links each stakeholders' role to the high-level objectives of city. Policymakers generally seek to deliver better life for businesses and citizens with a limited and shrinking budget – they are looking for promoting the use of IoT technologies to effectively provide public services like education, healthcare, public safety, transportation, good governance, etc., while reducing energy savings and monitoring city resources. Also, they play a key role in launching smart city initiatives and attracting sponsors, whereas businesses and technology providers are looking to develop innovative city services and create new business models.

At the same time, citizens are expecting more from their cities. They are looking for high quality of life and optimal conditions for professional development. They are seeking for personalized services that make them more efficient and effective and they are ready to pay for services that remove some of the frustrations of everyday life.

Although the disparity of stakeholders' interests, it worthwhile mentioning that a strong partnership building is a must for the achievement of a common city vision, flagship projects, effective collaboration, and synergy; otherwise competing interests can result in delays and even cancelation of smart city projects.

4.2 Services Opportunities and Application Domains

The continuous evolution of IoT has widened substantially the number of potential smart city services. The next table shows some examples of actual and futuristic services of each smart city case.

Table 5. Services offered by smart cities

Smart city	Smart services
Nice city	Smart metering and urban light management
Manchester city	City dashboard, air quality monitoring, smart meter in public building, application on public transport system, etc.
Barcelona city	More than 100 projects on smart lightning, smart grids, smart metering, electrical vehicles charging, smart water management, smart parking, smart transportation, smart citizen (participatory sensing), open government data, smart waste management, etc.
Santander city	Traffic management and transport (smart parking, traffic monitoring and predictions), environmental impact monitoring (air quality and noise), smart irrigation in parks, smart lightning management, smart tourism (augmented reality tourist guides), improved information from citizens to city council through participatory sensing.
Karlsruhe city	Smart energy (smart grid), smart mobility, smart house (energy efficiency)
Aarhus city	Smart metering for utilities, road and traffic management, environmental monitoring, city dashboard.
Birmingham city	Smart parking, traffic management, congestion and environmental monitoring, multimodal transport, energy monitoring in buildings, digital skills for careers, tele-healthcare, city dashboard.
Aparecida city	Smart energy, smart grid, smart metering and smart city.
Rio de Janeiro city	Smart energy, smart grid, and smart city.
Smart City Infrastructure Update (SCIU)	Utilities monitoring (mobile laser scanning in updating the power, water, and communication city infrastructure networks)
Smart Beijing	City management, emergency and public safety, transportation, water supply, energy saving, agriculture, health care, safe production, smart home, social security, education, etc.
Smart Hengqin	Tour, environment protection, Chinese traditional medicine, public safety, e-commerce.
Johannesburg Broadband Network Project (JBNP)	Intelligent traffic management, smart metering (energy and water), public safety, education.

As shown in table above, a wide range of smart services can be deployed as part of smart city initiatives. These can be grouped in the following domains:

- **Transportation** - to reduce traffic congestion and make travel more efficient, secure, and safe.
- **Environment** – to manage and protect city resources, control costs, and deliver only as much energy or water as is required while reducing waste.
- **Building** – to improve the quality of life in city buildings (e.g., home, commercial buildings, etc.).
- **Education** – to increase access to educational resources, improve quality of education, and reduce costs.
- **Tourism**– to improve access to cultural sites and improve entertainment services.
- **Healthcare** – to improve healthcare services availability, provide wellness and preventive care, and become more cost-effective.
- **Public safety** – to use real-time information to anticipate and respond rapidly to emergencies.
- **Agriculture**– to improve the management of agriculture.
- **City management** – to streamline management and deliver new services in an efficient way.

5 Proposition of Data Value Chain and Ecosystem Framework

The benchmarking of the smart cities highlighted the infant nature of the deployment of smart cities until now. Most of the cities are still in stage of the vision of their initiators, and even some services are provided, in most of the cases, it is still poor in terms of value creation, and far away from the revolution promised by these smart cities.

Bringing IoT solutions to life and making them economically viable requires a proper understanding of the genuine business opportunities that may be created out of this new emerging market. The attempts currently experimented seem not sustainable in time and the bases that begin to appear seem to fail in building a full ecosystem.

Looking at the IoT deployment cases, we may easily notice that manufacturing, deployment and management of smart devices as well as the analytical opportunities arising from big streams of real-time data all represent huge business opportunities. All of these elements alone or combined interest many stakeholders, but we observe a lack of common view about a ‘generic’ value chain. The creation of value is implied and not shared among the various stakeholders involved in IoT deployments and questions such as who is paying for what? who is getting money for what? have no clear answer.

Consequently, there is a need of a conceptual framework which identifies the different elements composing this value chain, the potential actors, their function, and their interactions to organize an overall ecosystem.

Based on the cases studied, Fig. 6 depicted a proposed value chain which identifies what elements create and bring value to the ecosystem.

Within this value chain, three underlying components can be sketched out: (1) infrastructure, (2) data management, and (3) application &services.

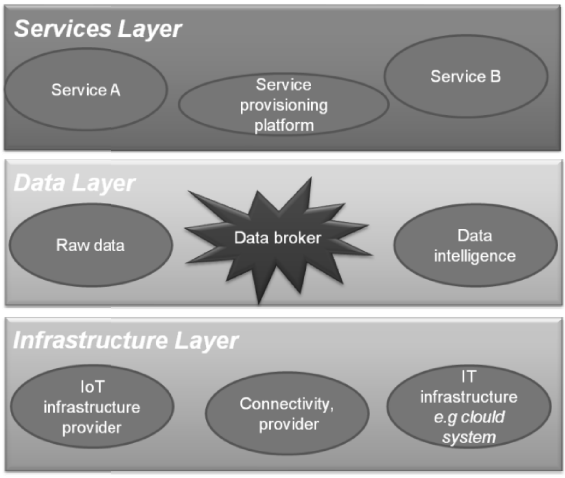


Fig. 5. IoT data value chain

On one hand, the **infrastructure layer** includes the IT and IoT infrastructure which enable to collect, store, and circulate data. Three different kinds of components are distinguished:

- **IoT infrastructure provisioning** covers the wide bunch of providers of smart devices which emerge such as sensors, actuators, cameras, traffics lights, smart meters, etc. The value created by this element into the chain is related to manufacturing and deploying such devices as well as improving their powerful, size (smaller), energy consumption, connectivity capacities for instance.
- **Connectivity provisioning** covers all aspects related to the network infrastructure to connect smart devices. Numerous technologies are available such as Zigbee, Z-Wave, WLAN, WIFI, GPRS and HSPA.DSL, FTT, LTE, MTC, 3/4G. The value created by this element in the chain relies on the transmission of information from devices to servers and between devices. The element is not specific to IoT deployments, and for now, the rules in place follow the scheme of traditional mobile communications. As a consequence, network operators are trying to offer services that go beyond connectivity provisioning to increase their revenues and value share.
- **IT infrastructure** covers the aspects related to the storage and record of the data transmitted (e.g., cloud computing). IoT and mobile services based on real-time communication increase the need of such infrastructures to enable real-time access to data as well as network-based computing services. These infrastructures are gaining in importance based on the new services and opportunities offered by new connectivity capacity and the becoming huge generation of data.

On the other hand, the **services and applications layer** includes the services delivered to users. The connected devices enable (or expect to enable) the creation of applications and services which address the specific needs of verticals market (e.g., supply chain management in manufacturing and logistics management in the transportation industry). But in the case of IoT, business opportunities are perceived as huger, particularly at the crossroad between diverse vertical domains and services (e.g., smart home and smart meter in the Brazil case). This appears as a mean to propel the emergence of new IoT service opportunities from cross-application synergies (as for instance the "smart-life" concept[3]) through platforms supporting multiservices. These kinds of platforms is gaining interest and importance for various stakeholders, and noticeably for big companies which initiate discussion with cities to propose them 'city operating systems' as it is the case between Cisco and Barcelona. But cities themselves want to invest into this for instance to promote usage and accessibility of open data and allow companies (and in particular SMEs) to take benefit from this and propose new services.

The value created by this layer is related to the services delivered to the end user based on sets of data coming from a single or multiples sources. But the business models behind are too complex because among others of the fragmentation of the stakeholders or the data fragmentation.

If we summarize these two layers according to their function, the infrastructure layer enables to generate large amount of data and the services and applications layer uses this data to produce services once they are aggregated, combined, and analyzed. But the link between the two is missing. Based on the analyses of the IoT deployments cases, data management appear to be a "puzzle" as many questions remain unclear such as who owns the data, how to pay the data producers, by whom ? what is the value of raw data ? etc.

Our proposed value chain aims to fill the gap between the infrastructure layer and the services layer introducing a "data layer" which includes three different elements:

- **Raw data** consists in data coming directly from sensors and devices (data producers) which may operate in different sectors. These raw data streams trace information and communication networks formed across a city, record citizens identity and their movements patterns, monitor the behaviors of users, and machines (e.g.; transport systems), etc. They are not necessary usable in this state but their aggregation allow offering services.
- **Intelligent data** consists in handling the amounts of data generated by several sources. Data collation and aggregation is becoming at the heart of value creation. The aim of data aggregation is to deliver only information relevant to specific requests to applications.
- **Data broker** consists in organizing the value flow of data. Such as in a real market, it consists at linking data producers, data managers, and data users, determines the value of data sets, fix the price and conditions of exchange between stakeholders, and so contribute to organize an ecosystem. Based on the interviews carried out, this particular element appears to be the cornerstone which is

[3] Terminology proposed by the European project BUTLER `www.butler-iot.eu`

missing today. In the different cases analyzed, either this function is not covered; either it is covered by different stakeholders that did not share a common view. It is for instance the case in different cities where the city services want to take care of this function in order to foster open data, but have to deal with technological providers which are fostering for their own standard.

This model is a first approach and could be enriched with what might exist in the world of 'big data'. It nevertheless reflects the vision of the current stakeholders and it is noticeable that also stakeholders outside the IT or IoT circles (for instance banks or insurers) might have the opportunity to play a role to organize this value chain and the business ecosystem.

On another side, involvement of public stakeholders introduces additional complexity into the conceptual value chain, as their rules are different from the private actors. The public value concept for IoT deployment is introduced by defining the core principals of a public business model which come down to the questions "Who *governs* the value network?" as well as "Is *public value* being generated by this network?" Governance and public value are then proposed as the two fundamental elements in business models that involve public actors. The governance parameters align with the value network and functional architecture already included in the benchmarking framework, where the public value parameters detail the financial architecture and value proposition.

The identified public value parameters related to the financial architecture are:

- **Return on public investment**: This refers to the question whether the expected value generated by a public investment is purely financial, public, direct, indirect, or combinations of these, and - with relation to the earlier governance parameters – how a choice is justified [16]. A method, which is often used in this respect, is the calculation of so-called *multiplier effects*, i.e., the secondary effects a government investment or certain policy might have, which are not directly related to the original policy goal.
- **Public partnership model**: The organizational parameter to consider in this case is how the financial relationships between the private and public participants in the value network are constructed and under which legal entities they set up cooperation [17][18] One example of such a model is the public-private partnership (PPP).

The public value parameters related to the value proposition are:

- **Public value creation**: This parameter examines public value [19][20] from the perspective of the end user and refers to the justification a government provides in taking the initiative to deliver a specific service, rather than leaving its deployment to the market [21]. One such motivation could be the use of market failure as a concept and justification for government intervention.
- **Public value evaluation**: The core of this parameter is the question whether or not an evaluation [17]is performed of the public value the government sets out to create and if this evaluation is executed before or after the launch of the service.

Now that we have established which parameters are important in a context where a public entity becomes part of the value network in offering a service, and how we interpret the different terms, we proposes to include them in a business model matrix that take into account public value (Fig. 7).

	Value network	Technical architecture	Financial architecture	Value proposition
Business design parameters	Control parameters		Value parameters	
	Control over assets	Modularity	Investment structure	User involvement
	Ownership vs Consortium Exclusive vs other Influence	Modular v integrated	Concentrated v distributed	Enabled, Encouraged, Dissuaded or Blocked
	Vertical integration	Distribution of intelligence	Revenue model	Intended Value
	Integrated v disintegrated	Centralised v distributed	Direct v indirect	Price/Quality Lock-in effects
	Control over customers	Interoperability	Revenue sharing	Positioning
	Direct v mediated Profile & identity management	Enabled, Encouraged, Dissuaded or Blocked	Yes or no	Complements v substitutes Branding
Public design parameters	Governance parameters		Public value parameters	
	Good governance	Technology governance	ROPI	Public value creation
Policy goals	Harmonising existing policy goals & regulation Accountability & trust	Inclusive v exclusive Open v closed data	Expectations on financial returns Multiplier effects	Public value justification Market failure motivation
	Stakeholder management	Public data ownership	Public partnership model	Public value evaluation
Organisational	Choices in (public) stakeholder involvement	Definition of conditions under which and with whom data is shared	PPP, PFI, PC...	Yes or no Public value testing

Fig. 6. Business model matrix including public value concepts

The detailed, qualitative description of all the parameters of this expanded matrix allows for the thorough analysis and direct comparison of complex business models that involve public actors in the value network. Such parameters need to be including in the benchmark of IoT deployments involving public actors such as in the case of smartcities.

6 Conclusion and Further Steps

This paper focused on the analysis of IoT deployments in the context of smart cities based on a benchmarking of twelve smart cities deployments worldwide. The analysis method used consisted of considering IoT deployments as a public policy or programmes which enable the use of classical evaluation tools. This perspective of analysis allowed the consideration of IoT deployments under all their dimensions, not only through their technological characteristics and performance. In particular, this analogy permitted the analysis of IoT deployments through the perspective of production of effects and impacts with respect to their preliminaries objectives. Through this exploratory work, it appears that IoT deployments and smart cities are far away from

delivering their promised objectives particularly to fill end users expectations. The main problem encountered deals with the lack of identification of the value creation along the chain from the infrastructure to the services and applications provisioning. In this chapter, we proposed a model of IoT data value chain organized around data generation process.

The cornerstone of this value chain relies on a data broker that structures, coordinates and manages the flow of data between the data producers (at infrastructure layer) to data users (at services and implications layer) making so the link between raw data and intelligent data. Public stakeholders introduce an additional complexity that should be taken into account by injecting public value concepts.

This work is of course limited by its exploratory nature, the size, and nature of the cases which were studied and the people interviewed. Only two categories of stakeholders were interviewed: policy makers and technological providers. These two categories are the ones who actively act to implement smart cities and deployed IoT-based solutions. They act as initiators of such experimentations and it was perfectly relevant to meet them especially to gather data on the objectives pursued and pitfalls and key success factors encountered to make their vision concrete. But this work could be largely enriched by complementing and confronting their vision (and the observations done through to that prism) with the vision of the other stakeholders involved such as the services and applications providers and the end users. The work reveals that the end-users are for now mainly out of the loop and gathering information about how they perceive the changes that appear in smart cities and the value they give to these new services could also importantly enrich the analyses.

Another limit relies on the scope of IoT deployments in the smart city context. It appears that this scope was not straightforward for everyone and covers various realities in the different cases. IoT deployment and smart cities are still immature. In most of the cases it is still a concept and does not correspond to a concrete reality. In this context, gathering information on objectives and impacts suffers from lack of accuracy and reflects more a vision than a reality. It could be interesting to pursue these observations in time to analyze how smart cities are "becoming real" and evolving with respect to their objectives.

Finally, the proposed models on data value chain and business model should be considered as a first step to foster a way for the different stakeholders to build a comprehensive business strategy (or business strategies) taken into account the divergence of interests among them. Clarifying the value creation at this stage and the value flow among players brings them a new perspective to define their positioning and identify the other players they are competing but also with who they may collaborate. The introduction of public value into the loop introduces also new way to understand the motivations and drivers of the different stakeholders and is helpful to clarify the game rules. Introducing more transparency and building a common and shared vision about the value chain and the ecosystem is in our view the better way to enable sustainable and economically viable IoT deployments and smart cities.

Finally, the proposed model could be further enriched by including Big Data-related considerations which exists independently of the context of smart cities.

References

[1] MacGillivray, C., Turner, V., Lund, D.: Worldwide Internet of Things (IoT) 2013–2020 Forecast: Billions of Things, Trillions of Dollars, Gartnet Market Analysis (2013)

[2] Smith, I.G. (ed.): The Internet of Things 2012, New Horizons (2012) IERC, Casagras2, ISBN 978-0-9553707-9-3

[3] Gauthier, P., Gonzales, L.: L'Internet des Objets... Internet, mais en mieux (2011) ISBN 978-2-12-465316-4

[4] Rifkin, J.: The Third Industrial Revolution: How Lateral Power Is Trans-forming Energy, the Economy, and the World (2009) ISBN 978-0230115217

[5] Patton, M.Q.: Utilization-Focused Evaluation. Sage, ISBN 978-1-4129-5861-5

[6] European Commission, MEANS Collection - Evaluation of socio-economic programmes (1999) ISBN 92-828-6626-2 CX-10-99-000-EN-C

[7] Mazhelis, O., Warma, H., Leminen, S., Ahokangas, P., Pussinen, P., Rajahonka, M., Siuruainen, R., Okkonen, H., Shveykovskiy, A., Myllykoski, J.: Internet-of-Things Market, Value Networks, and Business Models: State of the Art Report, ch. 2 (2013) ISBN 978-951-39-5249-5

[8] Hawkins, R.: The Business Model as a Research Problem in Electronic Commerce, Socio-economic Trends Assessment for the digital Revolution (STAR), IST project, Issue Report No. 4, SPRU – Science and Technology Policy Research, Brighton (2001)

[9] Linder, J.C., Cantrell, S.: Changing Business Models: Surveying the Landscape. Institute for Strategic Change, Accenture, New York (2000)

[10] Faber, E., Ballon, P., Bouwman, H., Haaker, T., Rietkerk, O., Steen, M.: Designing business models for mobile ICT services. In: Proceedings of 16th Bled E-Commerce Conference, Bled, Slovenia (2003)

[11] Pigneur, Y.: An ontology for m-business models. In: Spaccapietra, S., March, S.T., Kambayashi, Y. (eds.) ER 2002. LNCS, vol. 2503, pp. 3–6. Springer, Heidelberg (2002)

[12] Osterwalder, A.: The business model ontology: a proposition in a design science approach. PhD thesis, HEC Lausanne, Lausanne (2004)

[13] Gordijn, J., Akkermans, J.M.: E3-value: design and evaluation of e-business models. IEEE Intelligent Systems 16(4), special issue on E-business, 11–17 (2001)

[14] Ballon, P.: Control and Value in Mobile Communications: A Political Economy of the Reconfiguration of Business Models in the European Mobile Industry. PhD thesis, Department of Communications, Vrije Universiteit Brussel (2009), http://papers.ssrn.com/paper=1331439

[15] Vallet Chevillard, S., Le Gall, F., Zhang, X., Gluhak, A., Marao, G., Amazonas, J.R.: Benchmarking framework for IoT deployment evaluation, Project Deliverable (2013)

[16] Margolis, J.: Benefits, External Economies, and the Justification of Public Investment. The Review of Economics and Statistics 39(3), 284–291 (1957),

[17] http://www.jstor.org/stable/10.2307/1926044

[18] Bovaird, T.: Public-private Partnerships: From Contested Concepts to Prevalent Practice. International Review of Administrative Sciences 70(2), 199–215 (2004)

[19] Bovaird, T.: Developing New Forms of Partnership With the 'Market' in the Procurement of Public Services. Public Administration 84(1), 81–102 (2006)

[20] Moore, M.: Creating Public Value: Strategic Management in Government. Harvard University Press (1995)

[21] Talbot, C.: Measuring Public Value: A Competing Values Approach. The Work Foundation Research Report (2008), http://www.theworkfoundation.com/Assets/Docs/measuring_PV_final2.pdf

[22] Benington, J.: From Private Choice to Public Value? In: Benington, J., Moore, M. (eds.) Public Value: Theory and Practice, pp. 31–39. Palgrave MacMillan (2011)

Author Index

Subject Index

D

E

F

Acronyms

3G/4G	3^{rd}/4^{th} Generation (of mobile telecommunications technology)
6LowPAN	IPv6 over Low Powered Wireless Personal Area Networks
ACID	Atomicity, Consistency, Isolation, Durability
ACK	ACKnowledgement
ADSL	Asymmetric Digital Subscriber Line
API	Application Program Interface
API	Application Programming Interface.
API	Application Programming Interface
BS	base station
C2DM	Cloud to Device Messaging
CAN	Controller Area Network
CAPIM	Context-Aware Platform using Integrated Mobile services
CC	Convergecast Controller
CCN	Content-Centric Networking
CDN	Content Delivery Network
CDS	Connected Dominating Set
CI	Computational Intelligence
CIFS	Common Internet File System
CoAP	Constrained Application Protocol
COMET	COntent Mediator architecture for content-aware nETworks
CPU	Central Processing Unit
CPU	Central Processing Unit
CRUD	Create, Read, Update, Delete
CS	Content Store
CSV	Comma Separated Value
CSV	Comma Separated Values.
DaaS	Data as a Service
DBMS	Distributed Database Management Systems
DFS	Depth First Search
DGUMA	Data Gathering method Using Mobile Agents
DGUMA/DA	DGUMA with Data Aggregation

DGUMA/DAwoRC	DGUMA/DA without Route Control
DHT	Distributed Hash Table
DONA	Data-Oriented Network Architecture
DPG	Data Placement Graph
DRAM	Dynamic Random Access Memory
DSN	Deep Space Network
DSN	Detected Social Network
DTN	Delay-Tolerant Network
Eclat	Equivalence Class Clustering and Bottom-up Lattice Traversal
EDGE	Enhanced Data Rates for GSM Evolution
EIB	European Installation Bus
ESDA	Exploratory Spatial Data Analysis
ETSI	European Telecommunications Standards Institute
ETT	Electronic Triage Tag
EWIDS	Extreme Wireless Distributed Systems
FC	Fiber Channel
FIB	Forwarding Information Base
FIFO	First Input First Output
FL	Fuzzy Logic
FRB	Fuzzy Rule Base
FTP	File Transfer Protocol
GHCN	Global Historical Climatology Network
GIS	Geographic Information System
GPFS	General Parallel File System
GPS	Geographical Positioning System
GPS	Global Positioning System
GSG	Global Serialization Graph
GSOD	Global Summary of Day
HAM	Home Automation Module
HDD	Hard Disk Drive
HDFS	Hadoop Distributed File System
HMI	Human Machine Interface
HPC	High Performance Computing
HPC	High Performance Computing
HTML	HyperText Markup Language.
HTTP	HyperText Transfer Protocol.
HTTP	HyperText Transfer Protocol
HVAC	Heating, Ventilation and Air Conditioning
I/O	Input/Output
ICN	Information-Centric Network
ICN	Information Centric Network
ICNRG	ICN Research Group

ICT	Information & Communication Technology
ICT	Information and Communication Technologies
ID	Identity
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
ILM	Information Life-Cycle Management
IMDG	In Memory Data Grid
IO	Input-Output
IOPS	IO per Second
iOS	iPhone Operating System
IoT	Internet of Things
IoT	Internet of Things
IoT	Internet of Things
IP	Internet Protocol
IRTF	Internet Research Task Force
iSCSI	Internet SCSI
ISO	International Organization for Standardization
ISP	Internet Service Provider
IT	Information Technology
ITS	Intelligent Transport System
JNI	Java Native Interface
JPG	joint photographic expert group
JS	JavaScript.
JSON	JavaScript Object Notation.
KML	Keyhole Markup Language.
LAN	Local Area Network
LarKC	Large Knowledge Collider
LFU	Least Frequently Used
LHC	Large Hadron Collider
LLD	Linked Life Data
LOD	Linked Open Data.
LRU	Least Recently Used
LSG	Local Serialization Graph
LSH	Locality Sensitive Hashing
MAETT	Mobile Agent Electronic Triage Tag
MANET	Mobile Ad hoc NETwork
MANET	Mobile Ad-Hoc Network
MFV	Most Frequently Visited
MLN	Most Likely Next
MMP	massively parallel-processing
MPI	Message-Passing Interface
MR	MapReduce
MWSNs	Mobile Wireless Sensor Networks
N3	Notation3.
NAS	Network Attached Storage
NDN	Named Data Networking
NDO	Named Data Object

NDP	Number of Documents per Peer
NER	Named Entity Recognition.
NFC	Near Field Communication
NFS	Network File System
NLP	Natural Language Processing.
NL-SATA	Near Line Serial ATA
OAuth	Open Authorization
OCP	Open Context Platform
OLTP	Online transaction processing
OMPI	Open MPI
OMPIJAVA	Java bindings for Open MPI
ON	Opportunistic Network
OPAL	Open Portable Access Layer
ORTE	Open Run-Time Environment
OWL	Web Ontology Language.
P2P	Peer-to-Peer
P2P	Peer-to-Peer
P2P	Peer-to-Peer
PaaS	Platform-as-a-Service
PC	Personal Computer
PIT	Pending Interest Table
PMPI	Profiling Interface for Open MPI
PNG	Portable Network Graphics
POI	Point Of Interest.
PSN	Pocket-Switched Network
PSNR	Peak Signal-to-Noise Ratio
PTP	Parcel Transfer Protocol
PURSUIT	Publish Subscribe Internet Technology
QoE	Quality of Experience
QoS	Quality of Service
RAM	Random Access Memory
RAM	Random Access Memory
RDF	Reach Data Framework
RDF	Resource Description Framework.
RDFa	Resource Description Framework in Attributes.
REST	REpresentational State Transfer.
REST	Representational State Transfer
RF	Replication Factor
RF	Radio Frequency
RFID	Radio Frequency Identification Device
RFID	Radio Frequency Identification
RHH	Random Hyperplane Hashing

RP	Replication Percentage
RPC	Remote Procedure Call
RSS	Really Simple Syndication.
RSS	Really Simple Syndication
SaaS	Software as a Service
SAIL	Scalable and Adaptive Internet Solutions
SAN	Storage Area Network
SAS	Serial Attached SCSI
SCADA	Supervisory Control and Data Acquisition
SCF	Store-Carry-and-Forward
SFC	Space Filling Curve
SLA	Service Level Agreement
SLC/MLC-SSD	Single-level cell/ Multi-level cell solid state drive
SN	Social Network
SNC	Saami Network Connectivity
SOA	Service Oriented Architectures
SOAP	Simple Object Access Protocol.
SP	Super-Peer
SPARQL	SPARQL Protocol and RDF Query Language.
SQL	Structured Query Language, computer language to manipulate relational database
SQL	Structured Query Language.
SRP	Scale of Replication per Peer
SRSN	Self-Reported Social Network
SSD	solid state drives/devices
SWIM	Shared Wireless Infostation Model
TECD	Time-Evolving Contact Duration
TMM	Tree Management Module
TSV	Tab Separated Values.
URI	Uniform Resource Identifier.
US	Uniform Social
VANET	Vehicular Ad-Hoc Network
VSM	Vector Space Model
W3C	World Wide Web Consortium.
WLAN	Wireless Local Area Network
WMS	Web Map Service.
WOM	Word-of-Mouth
XML	eXtensible Markup Language.
XML	Extensible Markup Language
XOR	eXclusive OR

Glossary

α**:** The total number of consecutive heartbeats that a server has not received from another server.
$H_{r'}$**:** A hash value stored in the hash pool.
H_r**:** The hash value of log record L_r where *r* ranges from *maxID*+1 to *Z*.
L_v**:** A log record generated by the ReHRS master server whenever a write operation is initiated or completed, where v$\geq$1 is an incremental record ID.
M_{bef}**,** H_{bef}, and W_{bef}**:** They are nodes that act as the master server, HSS, and WSS before a state transition, respectively.
M_{aft}**,** H_{aft}, and W_{aft}**:** They are nodes that act as the master server, HSS, and WSS after a state transition, respectively.
${N_u}^S$**:** It represents node ${N_u}^S$ state (also called role) is *S*, where *u*= 1, 2, 3 and S ε*M*, *H*, *W*, *n/a* in which M, H, W, and n/a respectively stand for the master server, HSS, WSS, and unavailable
T_{early}**:** The time period from the moment when the WSS is requested to warm itself up to the moment when it is requested to take over for the HSS.
T_{remain}**:** The time required by the ReHRS's WSS to finish its warm-up process
T_{total}**:** The total time required by the WSS to finish its warm-up process when it receives a warm-up request from the commander.
$T - metadata$**:** The metadata that is frequently updated.
***K*:** It refers to the largest record ID of the log record currently collected in the HSS's journal
***masID*:** It is the maximum ID of the log records currently collected in the WSS's journal
Analytics: Using software-based algorithms and statistics to derive meaning from data
Anomaly detection: is the task of identifying events or measured characteristics which differ from other data or the expected measurements.
Assurance: the process of obtaining and using accurate and current information about the efficiency and effectiveness of policies and operations, and the status of compliance with the statutory obligations, in order for management to control an organization's activities.
Audit comfort: the support needed to draw conclusions provided by audit procedures.

Audit procedures: the various tests of details, analytical procedures, confirmations and other activities performed to gain audit evidence.
Batch layer: Refer to the corresponding processing layer of the Lambda architecture [38] that processes data in batches.
Batch processing: Refers to processing methods that collect data over a certain amount of time and then process several data sets as a whole.
Big data: the accumulation of datasets from different sources and of different types that can be exploited to yield insights.
Big Data analytics framework: An architectural reference model including architectural principles and a set of tools suitable for analytical challenges that require model learning and near real-time decision making based on large data sets.
Big data: refers to very large datasets such the data cannot be processed using techniques for smaller data collections; large data management and data processing tools have been developed for big data analysis.
Big Data: We use the term *Big Data* both to refer to "datasets whose size is beyond the ability of typical database software tools to capture, store, manage, and analyze" [37].
Business Intelligence: software system to reports, analyzes and presents data. These tools use data previously stored in data warehouse/Data Mart
Cassandra: an open source database management for huge amount of data on a distributed systems. Currently part of apache software foundation
CEP: *See* Complex Event Processing.
Clickstream analytics: The analysis of users' Web activity through the items they click on a page
Cloud computing: a computing paradigm in which highly scalable computing resources, often configured as a distributed system, are provided as a service through a network.
Cold-standby: redundancy When this scheme is employed, the task fails on one node will be executed by another node from the very beginning.
Complex Event Processing: Refers to processing methods that are tailored for application of logic over continuous data flows. Complex event processing derives complex events from more simple events.
Dashboard: A graphical reporting of static or real-time data on a desktop or mobile device. The data represented is typically high-level to give managers a quick report on status or performance
Data center: A physical facility that houses a large number of servers and data storage devices. Data centers might belong to a single organization or sell their services to many organizations
Data cleaning: refers to the process of detecting incorrect data in a dataset and removing or correcting the data in such a manner that it would be useable, such as correcting the data format.
Data Mart: subset of Data Warehouse ready to be used by Business Intelligence tools
Data mining: refers to retrieving data from a dataset, looking for patterns within this data, and displaying it in an intelligible way.
Data mining: The process of deriving patterns or knowledge from large data sets
Data set: A collection of data, typically in tabular form

Data Warehouse: a suite of applications and databases optimized for storing and processing large amount of structured data
DB, DBMS: Database management system. Software that collects and provides access to data in a structured format
Design effectiveness: that internal control procedures are appropriate in respect of the control objective they are intended to achieve and the risk they are intended to address.
Device: Technical physical component (hardware) with communication capabilities to other IT systems. A device can be either attached to or embedded inside a Physical Entity, or monitor a Physical Entity in its vicinity.
E: *E* could be either the HSS or the WSS. If *E* is the HSS, Whenever RTP times out, it checks to see whether it has received a heartbeat from the WSS during the RTP or not. But if *E* is the WSS, then it checks to see whether it has received a heartbeat from the HSS during the RTP or not.
ETL process: Extract, Transform and Load, tools to extract data form sources and transform them for operational needs and load in data store architectures (e.g. data warehouse)
Fully parallel redundancy: It refers to a set of schemes that consists of several servers. Each of them can offer a transparent takeover when another server fails.
Hadoop: An open-source MapReduce implementation developed by Apache.
Hadoop: Open Source framework to process large data set on distributed systems. Currently part of apache software foundation
Hot-standby redundancy: This scheme provides a backup node to maintain an up-to-date copy of the state of a master server. When the master server fails, the backup node can continue the operation of master server.
HSO: A hot-standby-only scheme.
HSS: A hot-standby server employed by the ReHRS.
Hybrid Redundant System: The full name of ReHRS.
In-Stream processing: Refers to methods that process data continuously (cf *batch processing*).
Intelligent Transportation Systems: A distributed architecture based on ICT components (e.g., sensors, actuators, servers, etc.) and M2M communications that is used to manage and control a transportation infrastructure.
Internal controls: the activities performed to ensure that policies and procedures are implemented and operated consistently and effectively, allowing errors and omissions to be prevented or detected.
Internet of Things: A collection of things having identities and virtual personalities operating in smart spaces using intelligent interfaces to connect and communicate within social, environmental, and user contexts.
Internet of Things: A global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on, existing and evolving, interoperable information and communication technologies
Interoperability: The ability to share information and services. The ability of two or more systems or components to exchange and use information. The ability of systems to provide and receive services from other systems and to use the services so interchanged to enable them to operate effectively together.

IOP: It is an initiated operation.
IoT: *See* Internet of Things.
ITS: *See* Intelligent Transportation Systems.
JobTracker: One of the master server of MapReduce. It coordinates all jobs running on the MapReduce clusters, performs task assignment for each job, and monitors the progresses of all map and reduce tasks
Lambda architecture: A high level architecture for Big Data systems defined by Marz and Warren [38].
M2M: *See* Machine-to-machine communication.
Machine-to-machine communication: A communication capacity whereby a set of distributed components (e.g., sensors, actuators, servers, etc.) communicate to support a computational process in realizing its established policy and where the process itself has no human participation (i.e., in terms of neither a provider of inputs to the process nor a consumer of outputs of the process).
Map task: A task of a MapReduce job used to execute the user-defined map function.
Mapper: A machine/worker used to execute a map task
MapReduce: A distributed programming model introduced by Google to process a vast amount of data in parallel on large-scale machines/clusters.
MapReduce cluster: A cluster consisting of a set of commodity machines/workers for conducting MapReduce jobs.
MapReduce: method for distributing tasks across nodes
Mashup: functionality that combines data presentation or presentation from two or more sources to provides new services
Metadata: information that describe content and context of other data like files
Model learning: Summarizes algorithms that characterize large data sets by considerably smaller data sets.
NameNode: Another MapReduce master server, which manages the distributed filesystem namespace and processes all read and write requests.
noSQL: Refers to data storage systems that use non-relational data models.
NR: A No-Redundant scheme, which is similar to the redundant scheme adopted by Hadoop for JobTracker.
NS2: A simulation tool supporting TCP, routing, and multicast protocols over wired and wireless networks.
Operating effectiveness: that internal control procedures are functioning as de-signed, regularly and consistently.
Ordinary parallel: The same to hot-standby redundancy
P-metadata: It is the execution result of an operation and is infrequently or never changed since it is generated.
Predictive modeling: involves the study of data and outcomes from earlier measurements or experiences, and developing a model to predict future data or experiences.
Q: Q could be either the HSS or the WSS. If Q is the HSS, it sends a heartbeat to the WSS whenever STP times out. But if Q is the WSS, it sends a heartbeat to the HSS whenever STP times out.
Reduce task: A task of a MapReduce job used to execute the user-defined reduce function.

Reducer: A machine/worker used to execute a reduce task
Redundancy mechanisms: A set of common methods to improve system reliability.
ReHRS : A reliable hybrid redundant system.
Relational database: data stored in relation-shaped tables
Representational State Transfer: An application model whereby behavior is structured on the basis of elementary operations (e.g., create, update, delete, etc.) upon distinct resources whose identity is represented in a URL notation.
REST: *See* Representational State Transfer.
RR-FOP: It is a response-required finished operation. It means that L_v records information concerning an operation requested by a client/worker, and this operation have been finished by the master server.
RTP: It is a predefined receiving time period.
RU-FOP: It is a response-unrequired finished operation. It means that L_v records information concerning an operation both initiated and finished by the master server
Scalability: The ability of a system or process to maintain acceptable performance levels as workload or scope increases
Schema: The structure that defines the organization of data in a database system
SCL: *See* Service Capability Layer.
Security triad: the three key information security objectives of Confidentiality, Integrity and Availability.
Semi structured data: data not resides in fixed structures but that contain markers to organize data elements
Service Capability Layer: A set of services offered to applications utilizing M2M capacities that is organized through a layer of software (e.g., on top of middleware platform).
SIM: See Subscriber Identity Module.
SQL: Standard Query Language. Refers to the standardized query language for relational database management systems.
STP: It is a predefined sending time period. Every STP, the HSS and WSS in the ReHRS mutually send a heartbeat to each other.
Stream processing: Refers to technologies that are tailored for applying processing logic over continuous streams of linearly ordered data with a given schema.
Streaming layer: Refers to the corresponding processing layer of the *Lambda architecture* [38] that uses *stream processing* for processing data continuously.
Structured data: data resident in statically dimensioned structures like matrixes or tables
Subscriber Identity Module: An embedded component that provides cryptographic mechanisms (e.g., authentication, authorization, etc.) of accessing the identity information of a subscriber to a service (e.g., a cellular mobile data service) and the profile information associated to that subscription.
Takeover delays: The time period from the moment when a server fails to the moment when another node acts as the server. It comprises the server failure detection time, IP address reconfiguration time, P-metadata retrieval time, and T-metadata retrieval time.
Time series: refers to a sequence of observations which are ordered in some way according to time or space. For example, temperature samples collected over time could

represent time series data, if the temperatures were gathered in sequence, over time.
Unstructured data: data sitting outside structured and organized data repositories such as relational databases.
Unstructured data: data that not resides in fixed structures like matrixes, tables etc.
Visualization: tools for providing a synoptic view of information
Warm-standby redundancy: When this scheme is employed, the states of the master server are periodically replicated to a warm-standby node. When the master server fails, the state replica can be used to restart the operation of the master server.
WSO: A warm-standby-only scheme.
WSS: A warm-standby server employed by the ReHRS
***x*:** A pre-defined threshold to request the WSS to warm itself up.
***y*:** A pre-defined threshold of taking over the failed server.
***Z*:** The largest ID of the log records collected in the journal of the commanders
ZFS: open source distributed file system solution implemented by Sun Microsystems