Hao Chen · Shuang Peng ·
Chun Du · Jun Li

Earth Observation Satellites

Task Planning and Scheduling

國防工業出版社
National Defense Industry Press

Springer

Earth Observation Satellites

Hao Chen · Shuang Peng · Chun Du · Jun Li

Earth Observation Satellites

Task Planning and Scheduling

國防工業出版社
National Defense Industry Press

Springer

Hao Chen
National University of Defense Technology
Changsha, China

Shuang Peng
National University of Defense Technology
Changsha, China

Chun Du
National University of Defense Technology
Changsha, China

Jun Li
National University of Defense Technology
Changsha, China

ISBN 978-981-99-3564-2 ISBN 978-981-99-3565-9 (eBook)
https://doi.org/10.1007/978-981-99-3565-9

Jointly published with National Defense Industry Press
The print edition is not for sale in China (Mainland). Customers from China (Mainland) please order the print book from: National Defense Industry Press.

The English translation of this book from its Chinese original manuscript was done with the help of artificial intelligence. A subsequent human revision of the content was done by the author.

Translation from the English language edition: "Dui Di Guan Ce Wei Xing Ren Wu Gui Hua Yu Diao Du Ji Shu" by Hao Chen et al., © National Defense Industry Press 2021. Published by National Defense Industry Press. All Rights Reserved.

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

Preface

From the first artificial satellite "Sputnik 1" launched by the Soviet Union in 1957 to the early twenty-first century, aerospace technology has achieved significant progress in the transition from the laboratory simulation to the practical application. As one of the most important categories in the spacecraft family, earth observation satellites mainly use spaceborne sensors to observe the earth's surface and lower atmosphere in order to obtain relevant information. It has irreplaceable merits such as wide coverage, long duration, little spatial and boundary limitations, and high security on stuff. Therefore, the EOSs are playing a significant role in the aspects of remote sensing, disaster prevention and control, environmental protection, territorial mapping, urban planning, agriculture assessment, and meteorology. With the rapid development of the world space industry, the number of EOSs has gradually increased, the types are increasingly rich, and the capacity and complexity of satellite platforms and payloads have increased significantly, but it is still difficult to meet the increasing and diversified earth observation requirements of various sectors. How to efficiently and reasonably develop daily satellite earth observation programs and make full use of satellite earth observation capabilities, so as to enhance the overall effectiveness of earth observation systems is a key issue that needs to be addressed in the space field.

The book is divided into eight chapters. Among them, the Chaps. 1 and 2, provide an overview of the key technologies and research status of task planning for earth observation satellites and analyze the challenges posed by various elements in the earth observation satellite system for planning and scheduling. The Chaps. 3–6 introduce the centralized EOSs task scheduling models and algorithms under deterministic conditions, dynamic scenarios of EOSs task rescheduling methods, distributed EOSs task scheduling models and algorithms, and EOSs onboard autonomous task scheduling models and algorithms; The Chaps. 7 and 8 introduce the architecture, main functions and human–computer interaction interfaces of typical satellite task scheduling and planning systems from the perspective of practical applications, and look into future technology trends.

The division of work in writing this book is as follows: Chap. 1: Hao Chen, Shuang Peng, Chun Du and Jun Li; Chap. 2: Hao Chen; Chap. 3: Hao Chen and Chun Du;

Chap. 4: Chun Du and Hao Chen; Chap. 5: Shuang Peng; Chap. 6: Shuang Peng and Hao Chen; Chaps. 7 and 8: Hao Chen and Shuang Peng. Hao Chen and Shuang Peng were responsible for the organization, unification and review of this book. The preparation of this book was also supported by our research group. Thanks to Prof. Wei Xiong, Prof. Ning Jing, Prof. Luo Chen, and Prof. Zhinong Zhong.

Relevant literature from recent years has also been consulted in the preparation of this book and is gratefully acknowledged.

Due to the limited level, it is inevitable that there are mistakes and irregularities in the book. I sincerely hope that readers can criticize and correct them.

Changsha, Hunan, China
September 2022

Hao Chen
Shuang Peng
Chun Du
Jun Li

Brief Introduction to This Book

Based on the actual needs of the earth observation satellite (EOS) operation control center, this book analyzes and introduces the development of the EOS task scheduling technology. Firstly, the state-of-the-art achievements and advances in this area are summarized. Secondly, for both static and dynamic scenarios, generic models, algorithms, and systems in centralized task scheduling technology, distributed task scheduling technology and onboard autonomous task scheduling technology are illustrated. Finally, the outlook on the expected technologies of the EOS task planning and scheduling for the future is summarized.

This book is suitable for engineers, post-graduate students, PhD students, and researchers engaged in aerospace task planning and scheduling.

Contents

Chapter 1
Introduction

1.1 Background of Earth Observation Satellite Task Scheduling

The past twentieth century is an extraordinary era with significance historical revolutions and great progress in science. For the first time, the human beings made a step out of the earth, where they lived and raised for hundreds of thousands of years, and made a step into the magnificent "Space age." From the first artificial satellite "Sputnik 1" launched by the Soviet Union in 1957 to the early twenty-first century, aerospace technology has achieved significant progress in the transition from the laboratory simulation to the practical application. Meanwhile, with the developments of the spacecraft design, the applications of the aerospace technology have permeated to different societies including scientific research, economical events, and security operations.

The earth observation satellites (EOSs) are one of the most important spacecrafts. The EOSs utilize the spaceborne sensors to detect the earth's surface and the lower atmosphere to obtain the information of the ground targets. It has irreplaceable merits such as wide coverage, long duration, little spatial and boundary limitations, and high security on stuff. Therefore, the EOSs are playing a significant role in the aspects of remote sensing, disaster prevention and control, environmental protection, territorial mapping, urban planning, agriculture assessment, and meteorology. The categories of the EOSs are typically divided into the visible light EOS, the infrared EOS, the multispectral EOS, the hyperspectral EOS, the ultra-broad spectrum EOS, the synthetic aperture radar (SAR) EOS, the surface electromagnetic detection (SED) EOS, etc., according to different sensors. All these techniques are of great interest to those powerful countries in the spaceflight with great advances in these years, which are the USA, Russia, China, France, and Germany. Specifically, the aerospace industry in China is arisen rapidly by the benefits of the achievements on the projects like "High-resolution Earth Observation System" and "Manned Spaceflight and Lunar Exploration." In this instance, the task scheduling of the EOSs that play a critical

H. Chen et al., *Earth Observation Satellites*,
https://doi.org/10.1007/978-981-99-3565-9_1

role in the aerospace technology, is of great and increasing interest to the academic research and industrial applications.

1.1.1 Summary of the EOSs Task Scheduling

The EOSs surround the earth in the specific orbits to provide observations for different targets on the ground according to different task requests. The obtained remote sensing image data is transmitted to the receiver on the ground station through a radio by the real-time transmission or the post-event transmission. Then the obtained data is preprocessed and recognized by the ground data processing center (DPC) to gain the effective information for the tasks.

In practical application scenarios, the satellite-based ground target observation is typically carried on the following steps: First, the users propose the observation requests, and the Earth Observation Satellite Operation Center (EOSOC) then generates the corresponding ground target observation task based on the related analytical and the computational operations. Next, a Satellite Earth Observation Programme (SEOP) will be produced through the Satellite Tasks Scheduling (STS), with respect to the task-specific information, the attributes of satellites (orbits prediction, available devices), and the relevant constraints (energy constraints, side viewing angle constraints, solar altitude angle constraints, cloud cover constraints, sensor switching time constraints, side viewing times constraints, spaceborne memory capacity constraints, etc.). After that, the control command for the satellite platform and the loaded devices will be allocated through the measurements and control facilities on the EOSOC based on the SEOP, and assign the observations and the data transmission operations to the satellites. The remote sensing image data obtained by the satellites will then be distributed to the ground receiving station (GRS) and will be processed by the related data enhanced and analytical systems before dispatching to the users. The generic process is depicted in Fig. 1.1.

It can be seen from Fig. 1.1 that the STS is essential to the satellite control and ultimate performance. In the early stage of the earth observation satellite technique development, the number of the earth observation tasks is affordable to the EOSOC as limited loaded devices and primary aerospace technology do not require complicated disciplines for action controlling and task planning. Consequently, the observation time and angle are typically fixed and the satellite control is simple. However, with the developments on the aerospace technology, the EOSs are capacity of sideways viewing, which allows satellites performing observation task through adjusting the sideways viewing angle of the observation sensors. Recent advances on the Agile Earth Observation Satellite (AEOS) achieve better flexibility on the earth observation with respect to the Yaw, Roll, Pitch three-axis-angle maneuverability. Meanwhile, it is necessary to take more challenging practical constraints into account, as well as searching the best STS strategy in a larger solution space when dealing with the earth observation task scheduling problem.

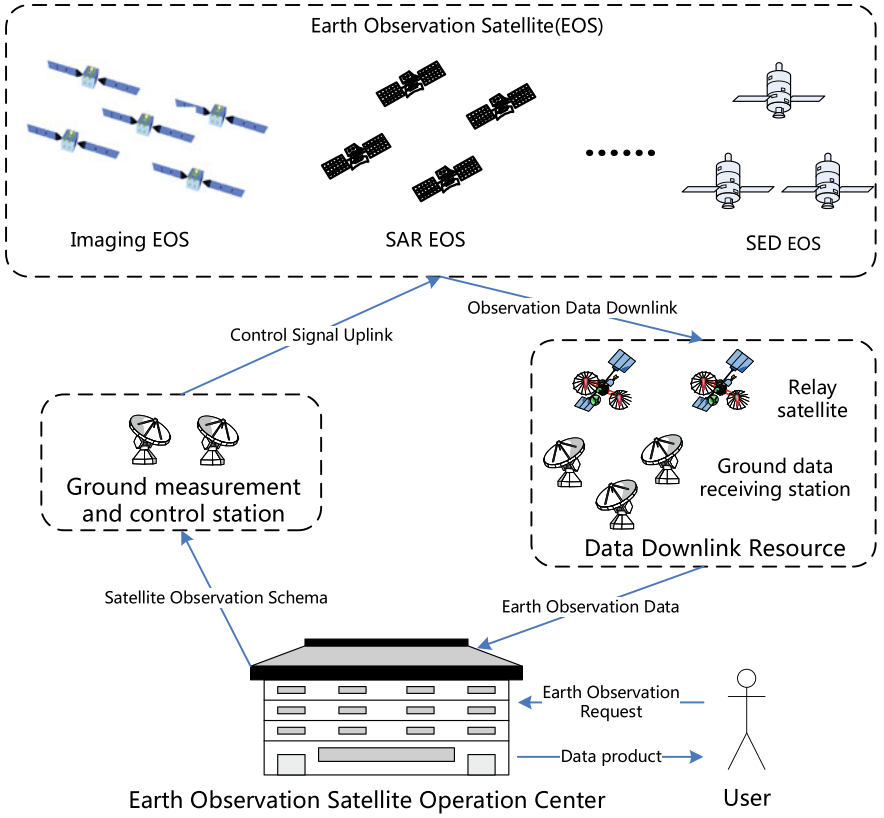


Fig. 1.1 Typical operation control flow of the EOSs

Priority is a fatal issue of the STS problem, task or target with higher priority should be sorted in prior. The value of the priority measures the significance of the observation data or the necessity of task. With the fact that the EOSs surround in the low-earth orbit with superior speed, the targets in the task list can only be very temporal observed toward the EOSs observation devices due to the earth curvature. The observed period is named access time window or observation time window, restraining the STS strategy. Besides, satellite observation sensors can only adjust to restricted angles in the specific observation time window, the power supply for the sensors adjustment, and the onboard storage memory is also limited, the EOSs are essentially constrained by the complicated conditions in practical scenarios. In this instance, satellites can barely complete all the earth observation tasks and only satisfy a part of the users' requests in one STS period. Therefore, the completed tasks in one STS period are merely a subset of the total tasks.

The STS aims at searching for the best subset from the whole satellite observation task set. This global optimal subset should maximize the total benefit (e.g., targets with the higher priority should be observed in prior) while satisfying all

the constraints. The relevant aspects of the STS include satellite platforms, satellite payload, earth observation requests, earth target environment and scenarios, which are interfered and interacted, resulting in the complexity and difficulty of the STS problem.

1.1.2 Theoretical Significance and Application Value

(1) The STS problem is a typically NP-hard problem referring to the previous evidence in the literature [1], and thus no polynomial temporal algorithm can achieve accurate estimation for this problem. The complexity of the STS problem is further augmented while taking the agnostic multiple satellite resource, the different observation requests, and the fruitful target environment and scenarios into account. In sum, studies on solving the STS problem are of great sensation on the theoretical and practical advances.
(2) The modern society has increasing demands on the remote sensing data, specifically, the EOSs and the related technique have been widely applied to industry, agriculture, economy, and military areas with significant role. At the same time, with the increments of the number and categories of the in-orbit EOS and the sustainable growth on the users' demand, it is imperative to propose an equilibrium and comprehensive STS strategy to assign the different categories of satellites and their related devices and resources, satisfying the user demand as well as improving the level of resources utilization. The EOS resource planning and scheduling, as a fatal technology in the field of the satellite control and practical applications, has become an important research direction in aerospace technology. A mutual and optimal STS strategy is of great industrial value in terms of supplying key technical support in the practical applications.
(3) Large amount of commercial aerospace companies spring up in China following the implementation of the national aerospace technology development strategy in these years. With the incremental demands and the explosion of the related patents, the STS, as one of the core technologies in aerospace application, has obtained more and more attentions with respect to the merits of stimulating the domestic demand, the promotion of the prosperity in all the related industry chain, and the contributions to the national economic development.

1.2 Research Status of EOS Task Scheduling

1.2.1 Centralized EOSs Task Scheduling

Ground-based centralized EOS task scheduling is applicable to the most traditional STS scenario, in which the ground-based centralized EOS task scheduling algorithm centrally schedules and assigns the earth observation tasks of all satellites. Generally, the ground-based centralized EOS task scheduling algorithm is deployed on a

high-performance computing cluster of the satellite operation and control center. It performs complex scheduling calculations to arrange the reasonable and optimized solutions for multiple satellites with a bird's eye on the observation requirements and satellite constraints. At present, there have been huge progress in this field and most of the current works are modeling for specific satellite task scheduling problems or transforming them into classical scheduling problems and are solved in the traditional schemes. Some scholars have also studied the models and methods of STS for the complex earth observation targets, such as the area targets and the moving targets on the ocean surface. Detailed information is introduced below.

1. Satellite task scheduling model

Satellite Earth Observation Programme (SEOP) involves various disciplines including computer science, operations research, artificial intelligence, and the performance indicators (control accuracy, control mode, and storage characteristics, etc.). In addition, the systems and their operating modes vary among different countries, leading to significantly different constraints, where the researchers have proposed their own modeling schemes from different perspectives. The majority of categories include the constraint satisfaction problem (CSP) models, the graph theory-based models, and the satellite-specific models.

(1) Constraint satisfaction problem model for EOS scheduling

The satellite earth observation process is subject to various constraints, and a large number of researchers across the world have established the CSP models by describing the various satellite constraints and the scheduling objective functions through mathematical expressions. Lemaitre established a constraint satisfaction problem model for the dexterous satellite mission scheduling and further compared the results of the greedy algorithms, the dynamic programming methods, the constraint propagation algorithms, and the local search algorithms [2]. Globus modeled the constraint satisfaction problem for multi-satellite scheduling, with a glimpse of the priority of mission requirements, as well as the constraint that each satellite has multiple remote sensing devices. These problems are typically solved by the simulated annealing algorithm, the genetic algorithm, the random hill-climbing algorithm, and the iterative sampling algorithm [3]. Bianchessi [4] modeled the constraint satisfaction problem for each of the two constellations and solved it using the column generation algorithm, the stochastic greedy algorithm, and the Lagrangian relaxation algorithm. Li considered data storage and downlink, applying the idea of hybrid modeling of constraint, to build a hybrid constraint planning model, and solved it by using the variable neighborhood forbidden search and the guided forbidden search methods [5]. Jun studied the integrated task scheduling problem of the imaging satellites in global optimization mode, established a constraint satisfaction problem model, introduced the multi-objective dominance relationship into the model, and proposed a multi-objective mission scheduling algorithm based on the SPEA2 genetic algorithm framework [6]. Jin established a constraint satisfaction problem that integrates satellite resources and data downlink resources and proposed a comprehensive scheduling method for star-ground resources based on the steady-state evolutionary algorithm and the Lagrangian relaxation method [7]. Sun established the

mechanism of single-objective optimization and the multi-objective optimization co-evolution for the agile satellite resource scheduling problem with multi-user control and proposed a multi-objective co-evolutionary algorithm for multi-objective joint mission scheduling [8]. Wu et al. proposed an improved non-dominated ordering based on the synthetic aperture radar satellite formation for the imaging scheduling problem, considering three objective functions simultaneously to meet the needs of different types of targets NSGA-III imaging the scheduling method [9]. Li designed a preference-based multi-objective optimization solution framework for the agile satellite earth observation task scheduling problem and proposed two preference-based multi-objective optimization algorithms based on the reference points and the target regions, respectively [10].

The CSP model can explicitly describe the relevant constraints and the scheduling evaluation criteria of the satellite earth observation process. It guides the algorithm to search for the optimal solution of the problem. However, the CSP model is established for specific satellite constraints and scheduling evaluation criteria, with close bound to the satellite scheduling problem. The constraints usually vary dramatically among the EOSs, and thus different CSP models are needed for the different EOSs planning and scheduling problems.

(2) Graph theory-based model for EOS scheduling

Some researchers have developed graph theory-based models by mapping ground targets to a series of graph vertices and constraint relations to sets of edges. For example, Gabrel studied the imaging paths of acyclic directed graphs to represent the way satellites transition between multiple tasks, which were solved using the shortest path algorithm [11]. Zhang analyzed the single-satellite scheduling problem and established the shortest path model for multiple targets of satellite earth observation, which is solved by the idea of marker update [12].

The biggest advantage of the graph theory-based model is that it intuitively represents the temporal and conflict relationship among multiple earth observation tasks. However, it lacks capability to express the relevant constraints and optimization criterion relationships in the process of multi-satellite joint earth observation. Therefore, the graph-theoretic models are widely implemented for the single-satellite scheduling problems or the multi-satellite scheduling problems that have been decomposed to the single-satellite level.

(3) Model and system for specific types of EOS

In order to improve the scheduling computing efficiency, some researchers have proposed their specific planning and scheduling models and built some scheduling systems for specific types of EOSs. When studying the observation task scheduling problem of the Landsat-7 resource satellite, Potter et al. established a scheduling model based on an idea called “multi-pass.” They completed the scheduling in multiple steps according to the priority of the task [13]. Yamaguchi [14] and Muraoka [15] studied the observation scheduling method for the AM-1 satellite with ASTER by calculating the priority of alternative imaging segments based on the shape and

distribution of the observation area, as well as scheduling the imaging tasks in order of priority. Chien [16] and Rabideau [17] designed the Automated Scheduling and Planning Environment (ASPEN) system for the National Aeronautics and Space Administration's (NASA) multiple-EOS task scheduling, which is solved by adopting an iterative domain knowledge-based repair mechanism. Frank [18] and Dungan et al. [19] from the NASA described the multi-satellite imaging scheduling problem as a constrained optimization problem and established a scheduling model based on the constraint-based interval (CBI) framework, which is solved by a greedy algorithm with random search. Based on the above approach, they have implemented the EUROPA scheduling system.

The design of the dedicated model is closely associated to the specific onboard equipment and scheduling scenarios. The model is simplified to improve the scheduling efficiency in a specific strategy, which is usually not general and generic.

2. Reducing the EOS scheduling problem to classical scheduling problems

Classical scheduling problems usually have more widely adopted modeling and solution methods. In order to utilize these models and solution methods, some scholars have reduced the EOS planning and scheduling problem to certain classical planning and scheduling problems before modeling and solving them. Vasquez et al. mapped the daily scheduling of the SPOT5 satellite observation task to a multi-dimensional 0–1 backpack problem and constructed the constraint satisfaction problem model. Then, they adopted a forbidden search for the solution and obtained the upper bound of the problem using a relaxation method [20]. Wolfe et al. reduced the single-satellite scheduling problem to a single-machine scheduling problem, established the corresponding integer programming model, and designed and compared eight heuristic algorithms from the perspectives of time window price and opportunity cost of a task, respectively [21]. The literature showed that the combination of these heuristics can obtain a solution close to the upper bound of the problem [1]. Based on the observation characteristics of ROCSAT-II, Lin treated it as a workshop scheduling problem with time window constraints, which decomposes the main problem into several subproblems using the Lagrangian relaxation methods and forbidden search methods, and solved it using linear programming techniques [22]. He studied the multi-EOS scheduling problem without considering the satellite data downloading and reduced it to a multi-machine scheduling problem with time window constraints. He established two models of mixed integer programming and constraint satisfaction, and solved it using the taboo search and column generation algorithms [23]. Li [5] and Guo [24] mapped the integrated task scheduling problem of satellites with multiple payload types, such as visible, synthetic aperture radar (SAR), infrared (IR), and multispectral, into a vehicle loading and unloading problem with time windows. They adopted the immune genetic algorithm and simulated annealing algorithm to solve the above models.

Although the classical scheduling problem can be modeled and solved by drawing on a large number of research results, the classical problem usually has a more rigorous formal definition, so the satellite-related constraints may need to be simplified in the normalization process to fit the expression form of the target classical

problem. The classical problem scheduling model is less scalable and difficult to express the satellite earth observation process under complex conditions. For example, it is difficult to express the nonlinear constraints in the satellite earth observation process when the EOS task scheduling problem is mapped to an integer programming problem model.

3. Algorithms for EOS scheduling

Whether building a mathematical model or reducing it to a classical problem, it is necessary to design the corresponding optimization algorithm. The selection, coding, and design of the optimization algorithms are directly related to the adopted models, while different planning and scheduling models take the most significant account on the design of solution algorithms. Optimization algorithms applied to this field can be essentially divided into two categories: the conventional optimization algorithms and the heuristic optimization algorithms.

Typical conventional optimization algorithms applied to the field of the satellite task scheduling include the constraint programming method [2], the label-setting algorithm [11], the column generation algorithm [23], and so on. Since the satellite task scheduling is a typical NP-hard problem, conventional optimization algorithms can only solve small-scale earth observation satellite task scheduling problems [18, 22].

At present, the vast majority of research works have adopted the heuristic optimization algorithms for solving the problem. For example, Frank et al. utilized a heuristic-based stochastic search algorithm and designed several heuristic rules [18]. Wang et al. [25] designed a priority-based heuristic search algorithm that utilizes a combination of conflict resolution, finite backtracking, and on-demand downloading rules, which can produce satisfactory scheduling results in a short time. Chen et al. [26] designed several priority-based task conflict resolution heuristics for agile satellites, which can effectively resolve the conflict between observation tasks with overlapping observation time windows.

However, the rule-based heuristic search algorithms usually perform ineffective when compared to the meta-heuristic optimization algorithms. The simulated annealing (SA), the tabu search (TS), the genetic algorithm (GA), the ant colony algorithm (ACA), the fireworks algorithms (FA) [27], the adaptive large neighborhood search (ALNS) [28, 29], and other metaheuristic optimization algorithms have shown stronger capabilities in solving the combinatorial optimization problems and have been widely used in the earth observation satellite task scheduling.

4. EOS task scheduling for specific targets

Observation targets can be divided into the point targets, the area targets and the moving targets according to their types. The size of the point targets is relatively small comparing to the width of the onboard sensors. They can be completely covered by the field of view of the imaging satellite sensors. The size of the area targets is much larger, and the sensor of EOS cannot completely cover the targets in a single shoot, so it is necessary to be shot more than one time by a single satellite or a group of multiple satellites. The moving targets refer to the targets in motion in the ocean

area of concern, which have a certain degree of uncertainty. This section introduces the current status of the EOS task scheduling research for area targets and moving targets.

(1) EOS task scheduling for area targets

In the area target-oriented satellite observation task scheduling, research work mainly adopts the research idea of "decomposition first, then scheduling." Specifically, the area target satellite observation scheduling problem can be decomposed into two subproblems: the area target decomposition and the satellite observation scheduling. For the area target decomposition, the area target is divided into multiple subareas (strips) that can be fully covered by one shot of the onboard sensors, and the operating parameters of the satellite payload (switching time, satellite sway angle, etc.) are determined within each time window when the satellite visits the subareas. The satellite observation scheduling problem for area targets focuses on the observation strips after target decomposition, and its scheduling strategy follows the traditional observation scheduling method for point targets.

Lemaître et al. [30] made an early exploration of the observation scheduling problem for area targets. The area target to be observed is a large polygon area, which usually cannot be completely observed by a single shoot. Lemaître et al. proposed a parallel segmentation approach to divide the whole polygon into several strips, ensuring that the area corresponding to each of these strips can be completely observed by the satellite in one shoot. Cordeau et al. [31] and Bianchessi [32] utilized the standard taboo search algorithm to solve area target observing scheduling problem and verified the feasibility of the proposed approaches by simulation experiments. Tangpattanakul et al. [33] proposed a genetic algorithm-based area target observing task scheduling task scheduling method, which can do multi-objective local search based on priority of area targets.

Wang et al. [34] designed a parallel partitioning method to convert the area target into small observation strips according to the observation requirements and designed a priority-based heuristic algorithm with a conflict avoidance and finite backtracking search, which was able to find a satisfactory solution to the problem. Yang [35] established a CSP model for area target observation scheduling of the EOSs by analyzing the characteristics of the area target observation and proposed a multi-satellite area target observation task scheduling algorithm based on a solution iterative repair strategy. He conducted some experiments to verify the practicality and effectiveness of the proposed algorithm. Xu et al. [36] proposed a three-phase solution framework, including the area discretization, the target decomposition, and the task scheduling to solve the EOSs large area observation problem. Zhu [37] analyzed various specific situations of the area target coverage optimization problem of multi-satellite cooperative observation and abstracted the multi-satellite cooperative area target observation problem into three basic types of problems, namely the maximum coverage area problem under the resource-limited situation, the minimum completion time problem under the resource-sufficient situation, and the minimum coverage

cost problem under the resource-sufficient situation, and proposed the corresponding solutions for each type of problems respectively.

(2) EOS task scheduling for moving targets

The moving targets in this book refer to the targets that move with the low-level speed on the ocean surface. Unlike the task scheduling for observation of stationary ground targets, the task scheduling for moving targets usually includes three processing steps: search and discovery, relay observation, and dynamic adjustment of the satellite observation programme. The difficulty and challenge of the problem solution is underlying how to accurately predict the target location by combining relevant information, i.e., the way of effectively performing search and discovery, relay observation for moving targets, and the dynamic adjustment of the observation programme, respectively, can be handled by the general dynamic rescheduling methods (see related demonstration in Sect. 1.2.2).

Berry [38] of the Defense Science and Technology Organization (DSTO), Australia, treated the observation scheduling problem of moving targets at sea as a sensor resource scheduling problem and established a generic framework for solving the problem. This method divides the observation area into several grids and implemented a probabilistic update strategy based on Bayesian criterion, in terms of establishing a Gaussian Markovian motion model for predicting the trajectory of the moving targets. Ci [39] developed a partially observable Markov decision process (POMDP)-based moving target searching model for the "online" EOSs moving target observation task scheduling. Guo [24] proposed a stochastic model based on the dynamic update of target distribution probability, as well as an adaptive and interactive multimodality target prediction method; however, whose sensor optimization strategy with the maximum cumulative discovery probability is hard to accurately estimate the target state. Lu [40] reduced the complexity of the problem by transforming the task scheduling problem of the ocean motion targets into a potential area target observing task scheduling problem. However, the uncertainty of the motion targets could lead to ineffective target discovery. Xu et al. [41] proposed a multi-model motion prediction method for the discontinuous satellite observation, which integrates conventional uniform motion prediction, track-based prediction, and potential area prediction. Zhang [42] studied the problem of multi-satellite cooperative search for sea surface moving targets in a multi-obstacle sea surface environment and proposed a multi-obstacle-oriented sea surface moving target motion prediction method for searching the targets on ocean surface. In terms of sensor planning and scheduling for moving targets, Yuan [43] decomposed the observation task scheduling problem of moving targets into two parts: the motion target state prediction and the sensor scheduling. He proposed a traceless particle filter-based sea surface motion target state prediction algorithm for target trajectory prediction, and a Rényi scatter-based satellite sensor scheduling method. Li et al. [44] proposed an onboard sensor scheduling method for moving targets based on reinforcement learning methods. Mei et al. [45] proposed a sensor scheduling algorithm based on the Kullback–Leibler (KL) divergence and the target detection probability, which significantly improved the capability of discovering moving targets for satellites.

1.2.2 EOSs Task Rescheduling for Dynamic Scenarios

Current methods for the EOSs task scheduling are mainly based on the deterministic scheduling, which is assumed that the observation tasks and satellite resources involved in scheduling keep static once the scheduling process starts. In practical, the satellites are working in a dynamic environment, where satellite resources may fail temporarily (or repair from failure) and new observation tasks may arrive randomly. If the satellite task scheduling process cannot adapt to these changes, it will definitely lead to a reduction of the satellite resource utilization. In view of this, the study of dynamic rescheduling of the satellite resources has become a hot topic of research for scholars all over the world.

Pemberton et al. [46] of Veridian, France, were the first to analyze the requirements of the multi-satellite dynamic rescheduling and divided the reasons for multi-satellite dynamic rescheduling into four cases: the changes in the state of satellite resources, the arrival of new tasks, the selection of task opportunities, and the influence of environmental uncertainty. They pointed out that the problem has the general characteristics of a continuous planning and scheduling problem, requiring that the changes between two consecutive EOSs observation plan should be small enough. However, no specific scheduling strategies and methods are given in the paper. Verfaillie et al. analyzed the characteristics of the dynamic and uncertain scheduling problems in general and classified the current processing methods into the reactive processing strategies and the proactive processing strategies [47]. The reactive processing strategy is mainly a method that reacts quickly to changes in the environment and reschedules according to a predesigned strategy, while the application of proactive processing strategy requires the acquisition of some prior knowledge of the changes in the environment, and the scheduling system will adjust the initial scheduling results autonomously based on this priori knowledge without waiting for the changes to occur before reacting. Obviously, according to the possible dynamic change factors in the satellite scheduling process summarized by Pemberton et al. [46], we cannot predict the information of the new observation tasks, and it is difficult to obtain any prior knowledge that the satellite will fail, so the dynamic rescheduling of the EOSs can only adopt the reactive processing strategy.

In the dynamic rescheduling process, if the initial scheduling results are directly discarded and the deterministic satellite task scheduling algorithms are used again for the new scenarios (new task and new satellite resources), the real-time scheduling requirements are hard to be satisfied when the problem size is with large scale. Furthermore, if the rescheduling results are too different from the initial scheduling results, the rescheduling results may be difficult to be applied, while if the newly arrived tasks are not processed or the old tasks which cannot be performed by failed satellites are directly discarded, the optimization of the satellite observation plan is difficult to be guaranteed. Therefore, the rational application of the initial scheduling results becomes the main strategy to deal with the dynamic rescheduling problem of the earth observation satellites at present.

When studying the SPOT satellite scheduling problem, Verfaillie et al. of the ESA proposed a dynamic treatment idea for the case of new tasks arrival [48]: A sufficient condition for a new observation task can be inserted into the scheduling programme is that when other tasks in the initial scheduling programme are changed by the task insertion, the changed tasks must be able to be inserted into another position of the scheduling programme and meet the scheduling deadline. Bensana et al. [49] studied the scheduling problem of a single earth observation satellite and used a mathematical method based on a Markov decision process framework to deal with the uncertainty of cloud cover and achieved an optimization of satellite scheduling results. Liao et al. analyzed the effect of the cloud coverage on the imaging process of the ROCSAT-II, modeled the problem as a stochastic integer programming problem, and used a rolling adjustment strategy to adjust the generated ROCSAT-II satellite task scheduling programme in real time according to the latest weather conditions [50], but could only handle the linearly constrained case. Billups et al. of Colorado University proposed various dynamic rescheduling methods based on the greedy algorithms, the genetic algorithms, the integer programming methods, and the graph-theoretic methods for the single-satellite dynamic rescheduling problem under the constrained simplifying conditions [51], but they cannot solve the satellite model containing nonlinear constraints.

All of the above works can only handle the single-satellite dynamic rescheduling problem, and with the advancements on the related research works, some scholars have extended these works and applied them to the field of the multi-satellite dynamic rescheduling.

Yang et al. established a dynamic constraint satisfaction problem (DCSP) model for the changes of the satellite resource state and the arrivals of the new tasks, respectively. They proposed a corresponding reactive scheduling algorithm combined with heuristic rules to minimize the observation programme adjustment [52]. The core idea of this work is the task-and-priority-based iterative repair strategy; i.e., when a task conflicts, only the higher priority task is allowed to replace the lower priority task. Kramer et al. proposed the task swapping algorithm based on an iterative repair strategy for the general dynamic rescheduling problem [53, 54], which is based on the Max-Flexibility, the Min-Conflicts, and Min-Contention heuristic rules to iteratively select scheduled tasks, then replace them with tasks that are not currently scheduled, and then reschedule the replaced tasks, saving the new solution if the scheduling is successful or returning to the original one if the scheduling fails. Zhang et al. [55] conducted a study for the scenario of the resource failure and the emergency observation task arrival and proposed a heuristic search algorithm containing five task replacement strategies, such as the interval pruning, the task pruning, the minimum conflict part for a single task, the minimum conflict set, and the maximum flexibility. Zhu et al. [56, 57] designed a task fusion strategy based on group partitioning, where they proposed a dynamic insertion algorithm based on the task backoff and repair. Wang et al. [58] designed the insert-delete-re-insert (IDI) algorithm with three operations: insert task, delete task, and reinsert task, and the insert-shift-delete-re-insert (ISDR) algorithm with four operations: insert task, shift task, delete task, and reinsert task, respectively. The experiments show that the optimality of ISDR

is better than that of the IDI algorithm. Jian [59] designed a variable neighborhood search method with four operations of "insert-redistribute-replace-delete." All of the above studies used a task replacement strategy based on heuristic rules, which can be regarded as an improvement of the priority-based iterative repair strategy. The iterative process decides whether the original task is replaced by the new task through certain heuristic rules, instead of considering only the task priority.

The task priority-based iterative repair strategy is a greedy scheduling strategy, which is difficult to guarantee global optimality. The heuristic rule-based task replacement strategy relies more on the task distribution characteristics in the scheduling problem, and the degree of optimization of the scheduling results will vary with different task distribution characteristics, which will lead to insufficient stability of the scheduling results in some extreme cases. Liu [60] applied dynamic rescheduling technology to multi-satellite observation of forest resources, and for perturbation situations such as the cloud cover occlusion and the resource failure. Firstly, a multi-objective rescheduling model was constructed with the objectives of small deviation from the original observation programme, large completion observation benefit, and load balance. And then a multi-objective particle swarm scheduling algorithm was proposed. The performance of the multi-objective rescheduling algorithm is significantly improved by designing a local search and global search balancing control strategy, an adaptive parameter adjustment strategy, and a population diversity maintenance strategy. Zhang et al. [61] designed an event-driven strategy based on the trigger rules and constructed a reactive scheduling multi-satellite multi-objective optimization module with the objective function of maximizing the scheduling gain and minimizing the perturbation measure for the observation of sudden events such as earthquakes and fires. In this work, the authors considered dynamic uncertainties such as the satellite resource failure and the emergency mission addition and integrated the task constraints, time constraints, satellite energy, and storage constraints. The number of triggers, the task completion rate, and the response time are also taken into account. Hu et al. [62] proposed a heuristic sliding time window insertion algorithm for emergency observation tasks, which integrates the task urgency and task conflict degree.

1.2.3 Distributed EOSs Task Scheduling

Under distributed conditions, the core research of multi-satellite collaborative task scheduling not only focuses on the resource assignment and task scheduling, but also focuses on how multiple satellites can autonomously perform task assignment through the information interaction and negotiation. The main idea is to model the satellite entities as autonomous and collaborative agents, where each agent only plans its own earth observation solution for the corresponding satellite (without knowing the global information of the whole constellation), and then generates the overall multi-satellite task scheduling solution through double-way collaboration among multiple agents [63]. The usage constraints of satellites are all encapsulated inside

the agent and separated from the collaborative framework. When satellite resources are added or withdrawn, only the agent representing the satellite needs to be registered or canceled in the existing multi-agent system, which ensures the high scalability of the system. The "independent scheduling and computation plus collaboration" approach of the multiple agents owns inherent parallel computing structure and benefits from multiple blade servers, which can adapt to the requirements of task on-demand processing.

Typical works include: Gao [64] proposed an extended contract network protocol for the distributed satellite system task collaboration based on the belief–desire–intention (BDI). The author then improved the basic contract network technology from three aspects of the task bidding and evaluation. Wang et al. [65, 66] used the historical information of collaborative scheduling to guide the subsequent collaborative scheduling and proposed a distributed collaborative task scheduling algorithm based on the multi-agent hybrid learning strategy and reinforcement learning. Li et al. [67] proposed an efficient collaborative mechanism for parallel multi-satellites, which significantly improved the efficiency of multi-satellite agent collaboration. Feng et al. [68] added operators such as the single-satellite scheme clustering and evolutionary computation to the contract network algorithm to improve the optimality of multi-satellite earth observation programme. Bonnet et al. [69] designed a multi-agent system with adaptive and self-organizing capabilities for the dynamic task scheduling problem of the earth observation satellite constellation, which can process the newly arrived earth observation requests in real time and improve the system's reaction capacity. Du et al. [70] integrated various mechanisms such as clustering-based task preprocessing, contract network protocol-based task assignment, and dynamic insertion-based task rescheduling, and proposed a multi-dimension multi-agent cluster collaboration model to solve the problems of inflexible interaction patterns, the low negotiation efficiency, and the poor dynamic responsiveness. Zheng et al. [71] introduced the idea of game theory into the multi-satellite collaborative task scheduling and designed various negotiation mechanisms such as the utility-based regret game, the smoke signal game, and the broadcast-based game.

1.2.4 EOSs Onboard Autonomous Task Scheduling

Depending on the task scheduling approach, the satellite onboard autonomous task scheduling [72] can be divided into batch scheduling [73, 74], rolling scheduling [75, 76], and sequential decision-making [77], which are described below.

1. Batch scheduling

As shown in Fig. 1.2, in the batch scheduling mode, the onboard task scheduling system periodically plans the daily or weekly tasks and generates the executable earth observation plan. The satellite executes each satellite action sequentially according to the earth observation plan, and before its execution ends the onboard task scheduling system plans again for the task in the next scheduling period and generates the

corresponding earth observation plan, and so on. Usually, there is no temporal overlap between two adjacent scheduling periods. The batch scheduling approach has been used in the early onboard autonomous task scheduling systems. In 1998, NASA validated the Remote Agent Experiment-Planning Scheduling (RAX-PS) technology using the Deep Space-1 probe in its New Millennium Program [73, 74]. The RAX-PS is composed of scheduling engine and knowledge base. The heuristic search algorithm is adopted to solve the satellite task scheduling problem and generate the satellite action sequence scheme for each onboard subsystem. Zhang et al. [78] introduced the heuristic rules and simulated annealing strategy into the genetic algorithm as a way to improve the quality and efficiency of solving satellite autonomous task scheduling problem. Miao et al. [79] proposed a hierarchical merit-based task scheduling algorithm for the multi-observation target imaging task scheduling problem in the hotspot regions, which generates observation schemes by optimizing observation targets at three levels of importance. Xue et al. [80] established a satellite integrated mission model for three types of the joint task scheduling problems including target observation, data transmission, and orbital maneuvering under emergency conditions. They decomposed the problem into two subproblems of the sequence planning and time scheduling for processing, and proposed a satellite autonomous task scheduling algorithm based on heuristic search and improved plan review techniques, etc. Batch scheduling is the mainstream method of the on-ground satellite task scheduling system and there has been huge advance in this area, which can be referred to Sects. 1.2.1 and 1.2.2.

In the batch scheduling approach, the task scheduling algorithm coordinates the observation tasks from a global perspective and usually searches for the optimal solution or near-optimal solution, but suffers from low efficiency. During the autonomous task scheduling process, the satellite is always flying at a high speed, and if the onboard autonomous task scheduling algorithm has not yet given an executable observation solution at the start time of observing a target, it means that the autonomous satellite cannot make a decision whether the current target should be observed, and the onboard task scheduling fails; if the intermediate results of the task scheduling process are used, the optimization of the observation solution is difficult to be guaranteed. Therefore, the batch scheduling method is suitable for application scenarios with the low timeliness requirements.

2. Rolling scheduling

As shown in Fig. 1.3, in the rolling scheduling approach, the satellite onboard autonomous task scheduling system plans the observation tasks in the next few

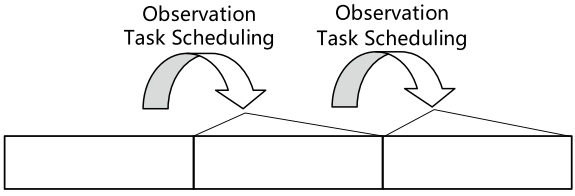


Fig. 1.2 Satellite onboard batch scheduling for the EOS tasks [76]

minutes to several orbital passes (called the rolling scheduling window) based on the status of the satellite's resources and equipment including energy, storage, attitude, etc., and the rolling scheduling window will keep moving forward with the time. When the states of observation tasks or resource in the rolling scheduling window change (e.g., the randomly arriving observation tasks, the task cancelation due to unsuitable observation conditions, the inconsistency of energy, the storage, and other resource states with program expectations, etc.), the onboard task scheduling system automatically and iteratively fixes and updates the existing observation program based on the variation of the observation tasks and energy storage resources in the rolling scheduling window. In the rolling scheduling approach, there is a temporal overlap between rolling scheduling windows, which ensures that the onboard task scheduling system can make timely and rapid adjustments based on the existing observation plan in response to the changes on the satellite. The rolling scheduling can be regarded as a compromise between the optimality and efficiency of the onboard autonomous task scheduling algorithm.

Currently, both of the domestic and foreign aerospace agencies have applied rolling scheduling methods to the autonomous task scheduling systems on satellites. In 2000, the NANA conducted an Autonomous Sciencecraft Experiment (ASE) experiment [75] on the Earth Observing One (EO-1) satellite to validate the Continuous Task scheduling Program Execution and Replanning technique [81] (CASPER, the core component of onboard autonomous mission planning and scheduling), considers task scheduling as an incremental process that uses a heuristic-based approach based on the current resource status, the undertaken observation tasks, and the resource and task changes, an iterative repair technique based on heuristic rules is used to achieve the continuous planning of onboard observation tasks, which improves the responsiveness of the planning system to dynamic environment [76, 82].

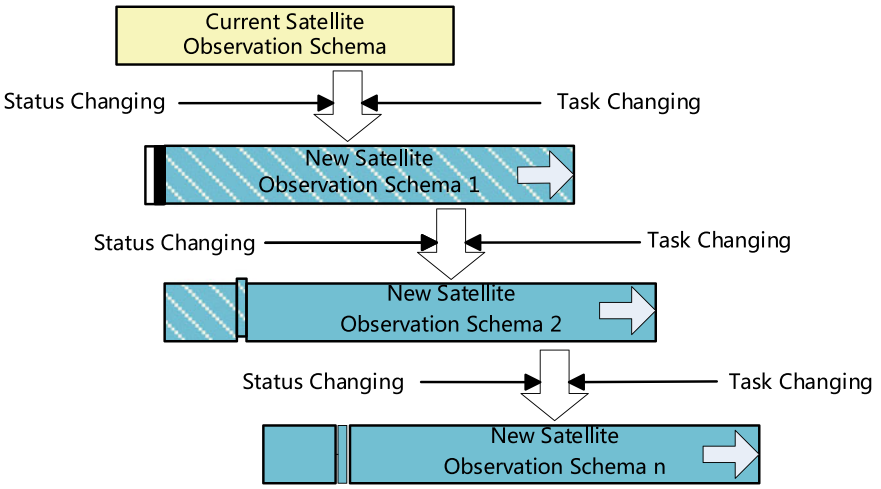


Fig. 1.3 Satellite onboard rolling scheduling for EOS tasks [75]

The promising performance of CASPER has led to its wide application in subsequent satellite onboard task scheduling experiments. For example, in the Three Corner Sat (3CS) experiment [83] conducted in 2002, NASA successfully applied CASPER to three nanosatellites to complete a cloud imaging test mission. In 2006, NANA applied CASPER technology to the "TechSat21." In 2006, NANA applied CASPER technology to the formation flight experiment of "TechSat21" program and conducted Autonomous Sciencecraft Constellation (ASC) experiment using three formation flight satellites [84, 85], which adopted the TeamAgent/ObjectAgent collaboration approach, and the TeamAgent used CASPER to perform the satellite formation mission. In 2013, NANA successfully applied CASPER technology to CubeSats and conducted the Intelligent Payload Experiment (IPEX) payload experiment [86]. In addition, scientists have also tried to apply CASPER to the deep space exploration activities [87, 88].

In 2001, the European Space Agency (ESA) conducted the PROBA-1 experiment [89, 90] to the validate onboard resource management and monitoring, the mission scheduling and execution, and the scientific data collection. In 2007, the ESA conducted the PROBA-2 experiment, which validated the technology of the relevant satellite autonomous operation platform and launch technology. In 2014, the ESA conducted the PROBA-3 experiment, in which two satellites in the orbit formed a coronal observer to observe solar activity and managed to validate the satellite formation flight technology.

Centre National d'Etudes Spatiales (CNES) and Office National d'Etudes et de Recherches Aerospatiales (ONERA) of France have conducted research on the framework, models, and algorithms for autonomous satellite onboard task planning techniques for spacecraft, respectively. The research work has been applied to (Autonomy Generic Architecture—Test and Application) project [91, 92]. Maillard and Verfaillie et al. [93, 94] proposed a joint satellite onboard-ground autonomous task scheduling framework considering the uncertainty of onboard energy and storage resource usage, in which the ground system generates an executable observation plan based on the satellite task scheduling model and uploads it to the satellite, and the onboard autonomous task scheduling system dynamically adjusts the observation programme by greedy strategy according to the use of onboard energy, storage, and other resources, so as to give priority to the execution of the high-priority tasks and the data transmission. Pralet and Beaumet et al. [77, 95] proposed a reactive/deliberative planning framework structure considering different types of satellite actions such as the earth observation, the data transmission, the attitude adjustment, the solar orientation, and the earth orientation. The reaction planning module determines the next satellite action based on the decision rules, and the deliberative planning module searches for an optimal solution based on the rolling scheduling approach using an iterative greedy random search algorithm. The reaction/deliberative framework can respond to the uncertain dynamics on board (e.g., the uncertainty in resource usage, the dynamic changes in the missions, the uncertainty in the cloud environment, etc.) in a timely manner, improving the rapid response capability of the satellite.

The German Aerospace Center (Deutsches Zentrum für Luftund Raumfahrt, DLR) conducted the Verification of Autonomous Mission Planning Onboard a Spacecraft

(VAMOS) autonomous task scheduling experiment in the FireBIRD mission [96], which is capable of reacting quickly to the onboard states or events.

In addition to the above-mentioned satellite onboard autonomous task scheduling systems, the rolling scheduling-based algorithms have also been widely studied. Li et al. [97] combined the supervised learning methods with the heuristic search algorithms. Neural networks were used to calculate the priority of each task, and then the heuristic search algorithm inserted the candidate tasks into the observation programme sequentially according to the priority of the tasks, enhancing the optimization-seeking capability. Liu et al. [98] discussed three issues such as the time window selection, the task-processing strategy, and the resource usage principles in the rolling task scheduling framework. Xing et al. [99] considered the influence of dynamic environment on the satellite onboard task scheduling process and analyzed the task update range and rescheduling timing, but did not give a specific rescheduling method. He et al. [100] considered the timeliness requirements of the random arrival observation task and the task execution deadline, and proposed various heuristic search algorithms such as the arrival time first, waiting time first, and deadline first. Xi et al. [101] designed a variety of the task ordering rules to address the timeliness requirements of dynamic demands and inserted tasks into the observation programme in a specified order. Liu et al. [102] combined the roulette idea with task ordering rules and proposed an iterative greedy search algorithm, where each iteration of the solution starts from zero, randomly selects a task ordering rule, determines the task order using the roulette idea, and then inserts the tasks into the observation programme sequentially to form a new observation plan. The algorithm terminates when the cumulative gain of the observation programme no longer improves. Li et al. [103] modeled the earth observation satellite as an intelligent agent and proposed two heuristic search algorithms, time utilization first and resource utilization first, whose main idea is to rank the observation tasks according to their evaluation indexes in a descending order, and then insert the observation tasks into the observation programme sequentially until the satellite can no longer undertake any task. He et al. [104] considered the influence of cloud cover on the satellite imaging and divided the task scheduling process into three stages: preassignment, coarse scheduling, and fine scheduling. However, this method is only suitable for dealing with the cloud occlusion in real time and can hardly cope with the dynamically arriving observation tasks. It can be seen that the above research mainly adopts the heuristic search algorithm to solve the problem and incrementally update the existing observation programme by choosing a reasonable observation task insertion order. However, the design of the heuristic strategy often faces difficulties in taking advantage of the domain knowledge, and the optimization and stability of the algorithm.

To address the shortcomings of the heuristic search algorithms in optimality, some researchers have tried to use deterministic search algorithms to calculate the optimal solution or combine relaxation techniques to search for the approximate optimal solution. Chen et al. [105] established a dynamic topological structure using a directed acyclic graph model for the autonomous task scheduling problem of electromagnetic detection satellites. They defined the approximate dominance relation of paths, reduced the number of paths that need to be retained at the vertices in the graph by path

dominance relaxation, and proposed an onboard autonomous task scheduling method based on label updating shortest path search. Chu et al. [106] introduced an application scenario of the dual-satellite cluster for the sea surface search and rescue, containing wide target discovery and high-resolution target identification. A search method based on the branch and bound algorithm was then proposed.

Compared with the batch scheduling, the rolling scheduling method decomposes the complex optimization problem into several overlapping optimization subproblems, which largely reduces the difficulty of problem and speeds up the computing process. However, the performance of task scheduling algorithm is easily affected by the length of rolling scheduling window. If the window is too long, the number of the observation tasks involved in scheduling will increase, and the computational cost of the task scheduling algorithm will increase as well. If the rolling scheduling window is too short, the optimization performance of the satellite observation schema may deteriorate sharply due to the "short-sightedness." Secondly, the rolling scheduling algorithm is mainly based on the rule-based heuristic search algorithm, and the optimality needs to be further improved.

3. Sequential decision-making

Sequential decision-making is an optimal decision-making method for uncertain dynamic systems. The process can be summarized as follows: starting from the initial state, making a decision at each moment, observing the change of the system's state, making the next decision according to the new state, and repeating until the end. The process of sequential decision-making is shown in Fig. 1.4. The onboard autonomous task scheduling system iteratively decides the actions that the satellite should perform at each moment (stage) according to the satellite's current states such as the energy, storage, and attitude. The major difference between the search algorithm used in the batch scheduling and rolling scheduling approach is that the sequential decision algorithm only needs to decide the next action based on the current status (including satellite status and the observation task status), without considering the future observation plan in advance, and thus greatly reduce the search space. Therefore, the sequential decision-making approach can react in real time to uncertainties such as random arrival of observation tasks, inconsistency between the resource status and observation programme expectations, thus greatly enhance the rapid response capability of the satellite.

Currently, there are very few studies using the sequential decision to solve the satellite onboard task planning problem, which is still in the exploratory stage and usually relies on the human-defined rules for decision-making. However, the optimization performance is not satisfying yet. Chien et al. [87] constructed an onboard ad-hoc response system, which determines the next satellite action to be performed according to the predefined response rules. Beaumet et al. [77] used a priority-based randomized decision algorithm for the satellite actions to determine the actions to be performed by the satellite, taking into account various types of satellite actions such as the to-earth data transmission, the earth observation, the sun orientation, and the earth orientation during the satellite operation. With the rapid development and wide

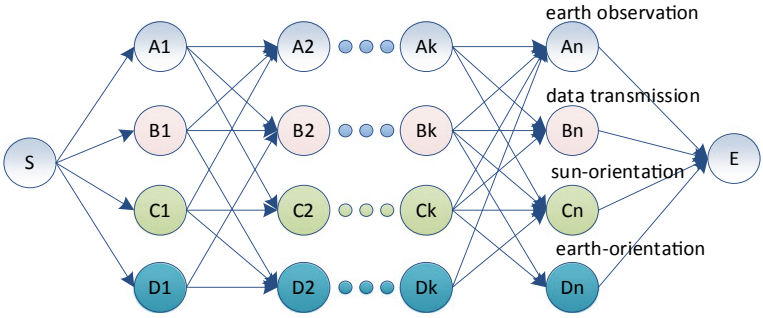


Fig. 1.4 Satellite onboard sequential decision-making for EOS tasks

application of machine learning technology, the knowledge mining of historical planning data can be used to guide the decision-making process, and help to improve the performance of sequential decision-making methods with rapid response capability. Currently, a small number of researches have been conducted using machine learning methods to solve the EOS scheduling problems [66, 107–109], which mainly employ case learning, supervised learning, and reinforcement learning to learn historical earth observation programme to further enhance the degree of decision optimization.

1.2.5 Satellite Data Downlink Scheduling

Satellite data downlink scheduling, which can also be called satellite range scheduling, refers to the task scheduling for the ground data receiving station. When the satellite enters the coverage area of the ground station antenna, it will establish a communication link with the ground station and downlink the earth observation data temporarily stored in the onboard memory to the ground, thus forming an information product. Usually, a set of receiving equipment can only provide data receiving service to one satellite at a time. Therefore, when multiple satellites appear in the same ground station reception range at the same time, the situation of multiple satellites competing for the ground station's data downlink service occurs, which leads to the data downlink conflicts and requires rational arrangement of the data downlink resources via optimization methods. A typical data downlink resource scheduling problem is the Air Force Satellite Control Network (AFSCN) scheduling problem, a ground station network including 16 sets of data transmission receiving antennas deployed worldwide. The network needs to handle more than 500 data downlink requests from different satellites every day. For the AFSCN scheduling problem, Barbulescu compared the performance of the heuristic construction and correction algorithm, the hill-climbing algorithm, and the evolutionary algorithm on the basis of the previous research experience, and experimentally showed that the evolutionary algorithm achieved the best results [110]. Clement et al. studied the Deep Space Network

(DSN) scheduling problem and proposed a hybrid algorithm combining local search operator and iterative repair operator based on heuristic strategy [111]. Jin et al. constructed a generalized function model for fixed-priority number downlink tasks and proposed a heuristic method for arranging downlink resources [112]. Wu established a satellite data real-time/playback downlink scheduling model for the characteristics of incremental tasks in the emergency scheduling scenarios and proposed an improved particle swarm genetic algorithm to solve the problem [113]. Chen et al. considered the dynamic change of priority, constructed a conflict constraint graph model, and proposed a solution using genetic algorithm. On this basis, they adopted improved genetic algorithm and discrete the particle swarm optimization algorithm to investigate the new emerging problems in the process of data downlink resource scheduling, such as the problem of scheduling data downlink resources under the condition of incremental update of observation tasks, the problem of data transmission conflicts resolving among member satellites of the same cluster and the problem of satellite data real-time/playback downlink adaptive decision [114–116]. Xhafa et al. studied the effectiveness of the heuristic hill-climbing algorithm [117], the simulated annealing algorithm [118], and the genetic algorithm [119] for solving the data downlink resource scheduling problem and concluded that the meta-heuristic optimization algorithm is more suitable for the given problem. Yao et al. employed a genetic algorithm with heuristic information to solve satellite data downlink resource problem with mixed observation data including both point targets and area targets [120]. Xing et al. optimized two objectives simultaneously including the failure rate of the data downlink task and the balance of ground station usage and proposed a method based on the Multi-Objective Evolutionary Algorithms (MOEAs) [121]. Furthermore, Zhang et al. combined support vector machines with the MOEAs to improve the computational efficiency of the multi-objective optimization process [122]. Du et al. [123]. proposed a particle swarm optimization algorithm for the topic-oriented satellite data for the downlink scheduling problem.

Chapter 2
Description and Analysis of the EOS Task Scheduling Problem

In order to analyze the essence of the EOSs task scheduling problem, this chapter first describes the working process and mechanism of EOSs and data transmission resources. Then, the elements of the EOSs resource scheduling problem, including the elements of earth observation tasks, satellite resources, data transmission resources, and the elements of the optimization objective function, are clarified. On this basis, the main characteristics and challenges of the EOSs task scheduling problem are analyzed, and the differences and connections between the EOSs task scheduling problem and the classical task scheduling problem are discussed.

2.1 Analysis of EOSs Operational Processes

While earth observation satellites (EOSs) may differ in their imaging principles and sensor parameters, they share many common characteristics that are relevant to the task scheduling of these satellites [6]. Understanding these commonalities is essential for conducting effective EOSs task scheduling studies. Generally, an earth observation satellite follows a similar process: It carries a high-resolution earth observation payload (the sensor), orbits in near-earth orbit at high speeds, and when it passes over a ground target, it adjusts its attitude to capture images of the target using the sensor. The satellite then temporarily stores the observation data onboard before transmitting it back to the ground through a satellite-ground communication link when it enters the reception range of a ground station. To facilitate this process, it is necessary to define and understand the concepts related to the satellite operation process.

H. Chen et al., *Earth Observation Satellites*,
https://doi.org/10.1007/978-981-99-3565-9_2

2.1.1 The Observation Process of EOSs

The EOSs' flying around the earth and the rotation of the earth make it possible for the satellite to observe certain ground areas, and the range of these areas can be determined by the parameters of the onboard sensors and the trajectory of the subsatellite points.

Definition 2.1: Trajectory of subsatellite points
The projection point of a satellite's position on the ground is called the subsatellite point, which can be expressed in terms of the geographical longitude and latitude. The motion of the satellite and the rotation of the earth make the subsatellite point move on the surface of the earth, forming the trajectory of subsatellite points, as shown in Fig. 2.1. Therefore, EOSs take advantage of the earth's rotation to pass over a large area of the earth surface and achieve imaging coverage of a large area. The significance of the subsatellite points trajectory is that it can be used to determine the coverage of the ground by an EOS.

Definition 2.2: Ground coverage of an earth observation satellite
The ground coverage of an earth observation satellite is the effective visible area of the satellite to the ground.

In the EOSs task scheduling problem, the ground coverage of an earth observation satellite is determined by the satellite's subsatellite trajectory, the effective observation range of the satellite and the maximum roll angle. When an EOS is in orbit, the ground coverage of the EOS is a band area with the subsatellite point trajectory as

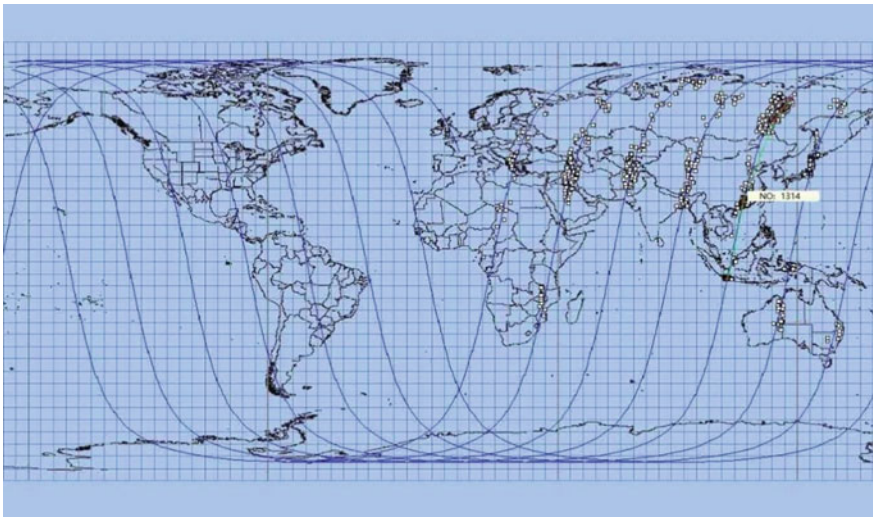


Fig. 2.1 Schematic diagram of the trajectory of subsatellite points

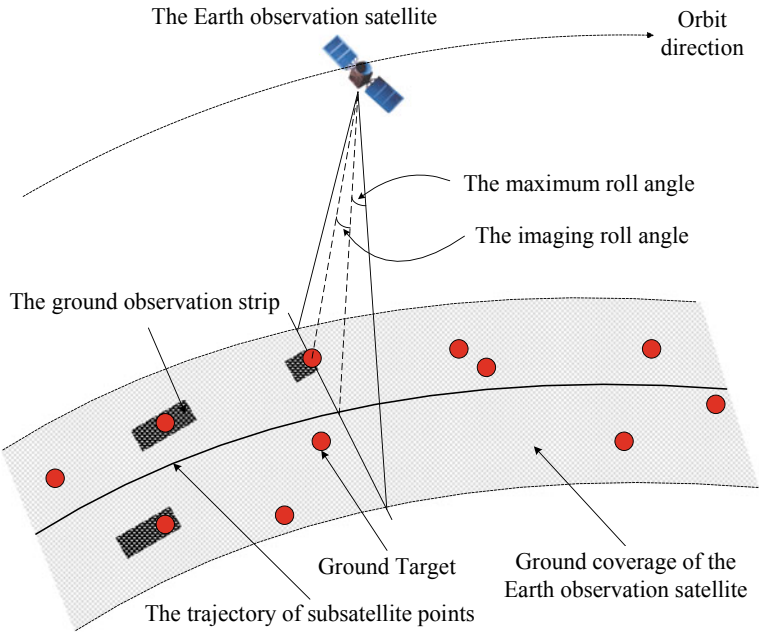


Fig. 2.2 Schematic diagram of satellite earth observation

the midline, and any single ground target within the band area can be observed by the EOS, shown as shown in Fig. 2.2.

Different EOSs have different orbital designs, which correspond to different imaging coverage areas, and these banded coverage areas may overlap during satellite operation, and ground targets in the overlapping areas can be imaged by multiple EOSs.

Definition 2.3: Ground observation strips
When an EOS conducts earth observation, it is moving at high speeds, and the onboard sensors have a limited field of view. As a result, the observation data generated by each earth observation is a strip with a specific width, referred to as the ground observation strips (or observation strips). The width of the observation strip is determined by the flight altitude of the EOS and the field of view of the sensor. If the observation strip covers a ground target, it indicates that the satellite has successfully observed the target. Typically, the ground observation strip of a satellite is relatively narrow, as illustrated in Fig. 2.2.

Definition 2.4: Earth observation requirement
The requirement for earth observation is initiated by the user. It usually consists of an area of the earth's surface and detailed observation demands for that area. A certain surface area of the earth, also known as the ground target, is the object of the satellite's earth observation; the detailed observation demands are some special

demands of the user for the observation data acquired by EOSs, and the common types of observation demands are the following.

- Demands for sensor utilization: The user requires that the observation should be a product imaged by a certain type of sensor, for example, demand for visible sensor, demand for IR sensor, demand for SAR sensor, etc.
- Demands for sensor resolution: The resolution of the data products generated by satellite observations should be greater than or equal to the sensor resolution demands proposed by the user.
- Demands for stereo imaging: The user requires multiple observations of this ground target at different angles to be able to accomplish the task of reconstructing this target in three dimensions.
- Demands for observation time period: The user requires that the ground target should be observed during a certain specified time period (e.g., 6:00–11:00 a.m.).
- Demands for observation timeliness: The user requires that the ground target observation data be available by a certain deadline.
- Demands for target group observation: The current target is strongly associated with one or more targets, constituting a target group that needs to be observed together, and the profit of these observations will be significantly reduced if observations are not available for all targets of the target group.
- Demands for time resolution: The demand is to observe a ground target at regular intervals to update the environmental situation periodically. For example, for the observation requirement "To observe No. 1 barrier lake in Wenchuan earthquake-stricken area every 6 hours," the time resolution is "not more than 6 hours."

Definition 2.5: Satellite imaging roll angle and satellite visible window
For the ground target that is usually not on the subsatellite point trajectory, the satellite needs to adjust the satellite platform or sensor attitude to observe it at a certain angle, and this angle is called the satellite imaging roll angle, as Fig. 2.2 shown. The satellite's imaging roll angle cannot exceed its maximum roll angle.

Based on the parameters of satellite orbit, the satellite's orbit can be forecasted for a future period of time, and thus the set of visible time periods between the EOS and the ground target can be fixed based on the geographic location of the ground target. The visible time period between the satellite and the ground target is called the satellite visible window to the target (satellite visible window for short). Obviously, the satellite can only observe the ground targets (with a certain imaging roll angle) within a certain satellite visible window.

Definition 2.6: Earth observation task
The earth observation requirements, along with the corresponding satellite visible windows, are the earth observation tasks. Since the satellite visible window already specifies the visual relationship between the satellite and the ground target, an earth observation task can be referred to as a satellite earth observation task. When an EOS performs a satellite earth observation task, it means that the satellite will observe the specified ground target within a certain satellite visible time window.

Definition 2.7: Earth observation meta-task
Typically, an earth observation satellite (EOS) has multiple visible windows to observe a ground target. Each visible window specifies three factors: a satellite, a ground target, and a candidate access window. These factors constitute an earth observation meta-task, or simply, a meta-task. The satellite is the subject of the earth observation meta-mission, the ground target is the object to be observed, and the candidate access time window is the period during which the satellite is visible to the ground target. A meta-task is considered executed when the satellite observes the ground target within the access time window. The set of meta-tasks is usually the input to the satellite task scheduling algorithm, which decides which meta-tasks will be executed.

Definition 2.8: Satellite earth observation programme
The output of a satellite task scheduling algorithm is a satellite earth observation programme. Typically, a satellite earth observation programme contains the meta-task to be performed and a satellite data transmission programme.

Definition 2.9: Multi-satellite observation conflict region
The overlapping coverage area of different earth observation satellites within one scheduling time horizon is called the multi-satellite observation conflict area, as shown in Fig. 2.3. Here the scheduling time horizon is the time period considered for satellite task scheduling. The ground target within the multi-satellite observation conflict region is called multi-satellite conflict observation target.

Definition 2.10: Conflict observation task
If the corresponding ground target of an earth observation task is a multi-satellite conflict observation target, the earth observation task is called multi-satellite conflict observation task (conflict observation task for short).

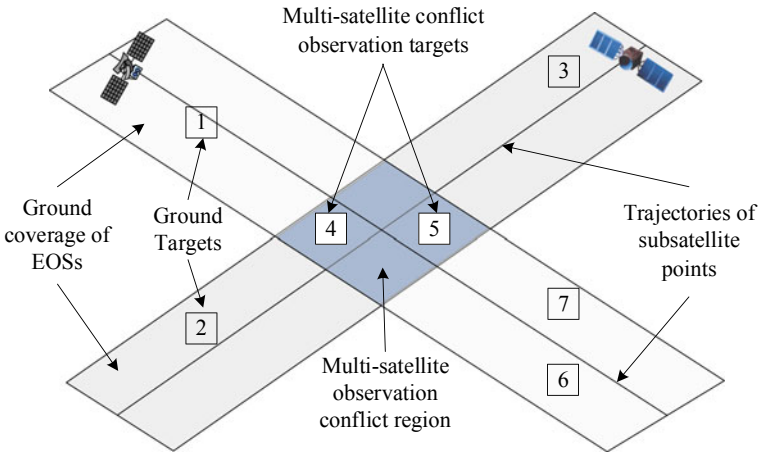


Fig. 2.3 Schematic diagram of the multi-satellite observation conflict area and the multi-satellite conflict observation targets

It is worth noting that for the earth observation requirement of only one observation, more than one observation to target will waste valuable satellite resources. However, for requirements with multiple observation requirements (e.g., stereo imaging requirements, etc.), tasks in overlapping observation regions are not conflict observation tasks.

In addition, earth observation satellites pass over the same ground target at certain intervals, a situation that we call revisits of ground targets. The main difference between revisits of ground targets and multi-satellite conflict observation targets is that revisits are for a single earth observation satellite, while the conflict observation targets are for multiple EOSs.

2.1.2 EOSs Data Transmission Process

Usually, the satellite data transmission resource includes two kinds of resources, ground station resources and relay satellite resources, and their data transmission processes are described below, respectively.

1. Ground station data transmission process

Ground data receiving stations, or simply ground stations, are situated on the earth's surface and mainly consist of antennas, servo equipment, and receiving equipment, among other things [7]. When an EOS enters the receiving range of a ground station, its antenna captures the satellite, and a satellite-ground communication link is established. The data stored temporarily in the onboard memory of the satellite is transmitted to the ground station through this communication link, and the onboard memory storage space is released once the transmission is complete. When the satellite leaves the receiving range of the ground station, the satellite-ground communication link is broken, and the ground station completes the data reception while the satellite completes the data transmission. It is important to note that each satellite data transmission activity corresponds uniquely to one ground station data reception activity. Unless stated otherwise, this text will not differentiate between satellite data transmission activities and ground station data reception activities. The process of the ground station data reception activity is illustrated in Fig. 2.4 [124].

Normally, a ground station has only one set of receiving equipment, and if multiple sets of receiving equipment exist at a ground station, each set of receiving equipment can be treated as a logical ground station. In the satellite data transmission resource scheduling process, the ground station with multiple sets of receiving equipment can be treated as if it were several co-located logical ground stations with one set of receiving equipment.

Before the start of data reception, some preparations for the ground station are required, before the implementation of the reception of the satellite data. And after the data transmission process, the ground station also needs some time to release the link.

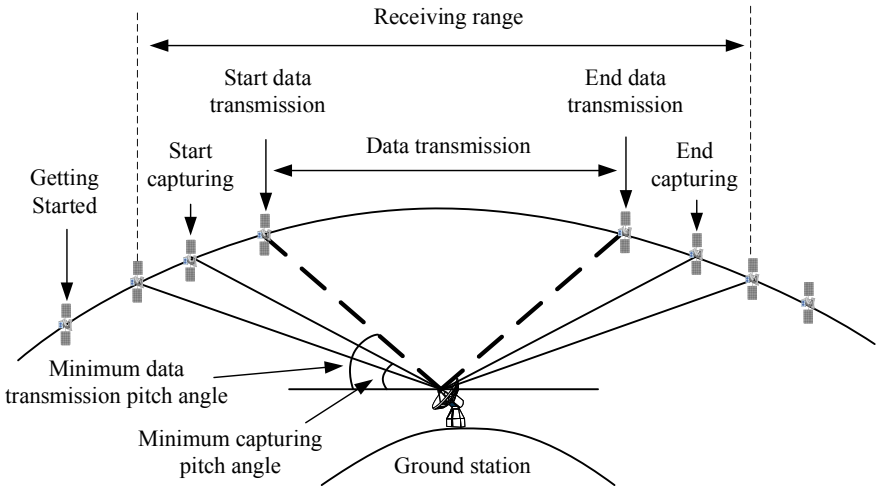


Fig. 2.4 Schematic diagram of the ground station data reception activity process

Therefore, the ground station reception equipment needs a certain reception preparation time and a resource release time [125]. In the scheduling process, the minimum time interval between two data reception processes of a ground station is called the ground station data reception switching time.

When implementing data reception, a set of equipment can only receive data from one satellite at a time, called the ground station resource capacity constraint. When multiple EOSs enter the reception range of a ground station simultaneously, the ground station resource capacity constraint leads to multi-satellite data transmission conflicts since the ground station can only serve one satellite for data transmission.

2. Relay satellite data reception process

Relay satellites are positioned in geosynchronous orbit to provide high coverage Telemetry, Track, and Command (TT&C) as well as data relay services for medium- and low-orbiting spacecraft. The use of relay satellites has the potential to significantly improve the communication coverage of such spacecraft. Essentially, relay satellites indirectly extend the visible time window between spacecraft and ground stations, thus increasing the amount of time available for spacecraft TT&C and data transmission. Proprietary agencies manage relay satellite systems, which offer TT&C and data relay services to multiple spacecraft operated by various departments and companies. These departments request relay satellite resources as needed, and the relay satellite management agency plans and schedules them in a unified manner. This ensures that available relay satellite time windows are efficiently allocated to different spacecraft of each department.

When a relay satellite switches from providing data relay service for one spacecraft to providing data relay service for another spacecraft, a certain amount of switching time is required, called the relay satellite data reception switching time. The ground

station data reception switching time and the relay satellite data reception switching time are collectively referred to as the data reception switching time of the data transmission resources.

Similar to how ground stations are handled, we assume that a relay satellite can only serve one spacecraft at a time, called relay satellite resource capacity constraint. If a relay satellite has the ability to provide data relay service to multiple spacecrafts at the same time (called concurrent service capability), we logically treat it as if it were multiple relay satellites without concurrent service capability.

This book assumes that relay satellite scheduling is completed and that relay satellite available time windows are known prior to the planning and scheduling of earth observation satellite resources.

2.2 Description of EOS Task Scheduling Problem

Task scheduling for EOSs refers to the problem of determining which satellites, at which time and in which operating mode to perform earth observation tasks, and at which time and with which data transmission resources to transmit observation data to the ground, in order to maximize the comprehensive benefits, addressing the large number of earth observation requirements. It can be seen that the EOSs task scheduling problem can be abstracted as a complex combined optimization problem composed of earth observation tasks, satellite resources, data transmission resources, and other elements. In the following, the elements will first be introduced, then the relationship between each element in the scheduling process will be analyzed, and finally the characteristics and difficulties of the EOSs resource scheduling problem will be analyzed.

2.2.1 Elements of Earth Observation Tasks

The earth observation requirements proposed by users (see Definition 2.4) is transformed into a number of earth observation tasks after the satellite visible time window is calculated (see Definition 2.6). From the point of view of the earth observation satellite task scheduling process and applications, the earth observation task elements should contain the following information.

1. Target geographic location and target category

The geographic location of a ground target is represented by a series of latitude and longitude points. If the target size is negligible compared to the width of the satellite-ground observation strip (see Definition 2.3), then the target can be represented as a point with no size range. For satellite task scheduling purposes, the target is represented by a single latitude and longitude point. However, if the target size is significant, it is considered an area target. An area target can be represented as a series

of latitude and longitude points arranged in a certain order, and the geographical range of the area target is defined if the first latitude and longitude points are connected with the last one in the sequence to form a closed polygon area. Usually, a single observation by an EOS cannot cover the entire area target, and multiple satellites are required to coordinate the observation. The spatiotemporal information (geographic locations with time labels) of the target determines its visible time window with different satellites.

2. Target priorities

Target priority is an evaluation of the importance or urgence of earth observation requirements (see Definition 2.4), with a higher priority indicating that the task is more important or urgent. Earth observation tasks with a high priority should be given priority in response.

3. Detailed observation demands

In Definition 2.4, several common detailed observation demands are outlined. These demands vary and must be considered individually in the satellite task scheduling process. From the viewpoint of EOS task scheduling, detailed observation demands can be classified into two categories: detailed observation demands for filtering and detailed observation demands for scheduling. Typical detailed observation demands for filtering include sensor utilization, sensor resolution, observation time period, and observation timeliness. In terms of task scheduling, fulfilling filtering-type demands is relatively straightforward and can be accomplished by utilizing a rule-based expert system [126] to filter out EOS resources that are not capable of carrying out earth observation tasks. As an example, consider a detailed observation demand with a sensor utilization requirement of "visible." In this case, it is sufficient to filter out all satellite visible windows that are accessed by non-visible sensors (e.g., IR sensors, SAR sensors, etc.). Typical detailed observation demands for scheduling include stereo imaging, target group observation, and time resolution. These demands require special considerations in the scheduling algorithm. For instance, for a stereo imaging demand, the observation must be carried out from different angles to the ground target, necessitating the scheduling algorithm to consciously schedule multiple EOSs with the same type of sensors to observe the target. Moreover, the scheduling algorithm must make a prudent decision that, if a satellite's observation capability falls short of the stereo imaging demand, the algorithm should redirect the satellite's observation toward other ground targets instead of the target with the stereo imaging demand.

2.2.2 Satellite Resource Elements

1. Satellite orbit

Satellites operate at high speed along a fixed space orbit, in contrast to aircraft that can freely navigate the sky. The characteristics of a satellite's orbit determine its

relative geometric relationship with the earth during its movement in orbit. Earth observation satellites typically reside in low earth orbit, limiting their visibility of ground targets and ground stations to specific visible windows. Satellites can only perform earth observation tasks, such as capturing images using onboard sensors or transmitting data, during these visible windows.

2. Satellite platforms

A satellite platform is a general term for the subsystems that ensure the proper functioning of the payload. Based on their maneuverability, earth observation satellites (EOSs) can be categorized as Agile (AEOS) and Non-Agile (NEOS).

Compared to traditional NEOS, AEOS has swift three-axis attitude maneuvering capabilities of Yaw, Roll, and Pitch on their platforms, which significantly increase their visible window on a target. Figure 2.5 provides a schematic diagram highlighting the differences between agile and non-agile EOSs [10]. The diagram presents three candidate observation tasks numbered ①, ②, and ③. It is evident that the visible window of the AEOS is larger than that of the NEOS, and it is sufficient to observe a ground target. As depicted in Fig. 2.5, the AEOS can perform all three observing tasks, while the NEOS can only perform two. The flexibility of an AEOS considerably enhances its observational abilities, but it also increases the complexity of the observation task scheduling problem. To schedule tasks for AEOS, one must decide not only which target to observe but also when to initiate the observation.

In contrast to agile earth observation satellites, non-agile ones typically possess limited capabilities, often only allowing for rolling (side-looking) observations. Electromagnetic environment detection satellites, on the other hand, typically lack attitude

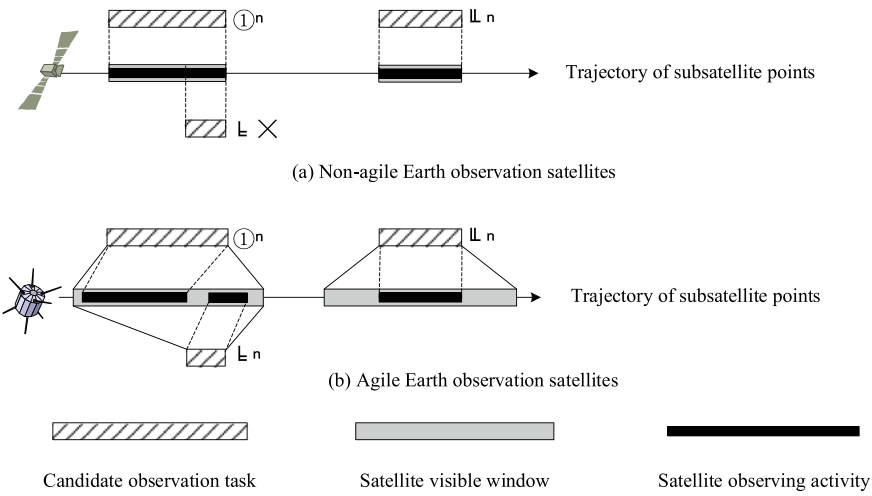


Fig. 2.5 Comparison of agile earth observation satellites with non-agile earth observation satellites

adjustment capabilities and are only capable of collecting data on the electromagnetic environment along the subsatellite trajectory.

3. Satellite observation payload constraints

The primary payload of an earth observation satellite is an earth observation sensor, with each type of sensor payload possessing unique characteristics. However, from a satellite task scheduling perspective, every switch-on and switch-off of a sensor can be considered a distinct working mode of the sensor. Constraints on payload usage can be summarized as follows:

- Sensor observation range constraint

Each working mode of a satellite sensor corresponds to a specific observation spectrum, resolution, and range of view. The working mode set of a sensor ultimately determines the observation capability of the satellite payload. Therefore, the satellite can only fulfill earth observation requests that fall within its observation capability. For example, an optical satellite with a resolution of 3 m cannot meet a user's observation requirement for a 1-m resolution optical image.

- Working mode observation condition constraints

Certain satellite payloads may encounter limitations in performing observations under specific conditions. For instance, optical satellites are particularly sensitive to light conditions and cloud cover, impeding their ability to observe ground targets in situations of poor lighting or overcast skies.

- Working time length constraint

In order to safeguard satellite sensors, the maximum on/off time of each working mode is typically restricted. However, to ensure the acquisition of dependable observation data from ground targets, the switching on/off time length of a satellite must also satisfy a certain duration known as the minimum on/off time.

- Working mode switching constraints

When performing observations of consecutive ground targets, a satellite requires a certain duration for switching between working modes. As such, the minimum duration time between working modes creates a constraint on the switching time.

- Action constraints

Restricted by the payload characteristics, satellites are limited to performing a single action at any given time, and there is no temporal crossover permitted between two observation actions that employ different working modes by the same satellite.

4. Satellite energy constraints

Satellites rely on a power supply to facilitate their routine operations, including those of subsystems such as the onboard computer, attitude control, satellite system thermal control, sensors, and communications. Typically, the power for earth observation satellites (EOSs) is derived from solar energy. To ensure uninterrupted operation

even in the shadow area, a combination of the "solar array + battery" power supply method is adopted [24]. However, the accumulated working time of the sensors is often restricted due to limitations in the satellite's power supply capabilities.

5. Satellite onboard memory capacity constraints

Satellites performing earth observation tasks temporarily store acquired data in their onboard memory. However, this process reduces the available capacity of the onboard memory. To restore this capacity, the satellite transmits the data to a ground station directly or via a relay satellite. The inability to transmit the data effectively results in a zero available capacity for storing earth observation data.

2.2.3 Elements of Data Transmission Resources

Data transmission resources are comprised of ground station resources and relay satellite resources. When either of these resources provides data downlink services to a satellite, the satellite is then capable of transmitting observation data.

1. Start time and end time of the data transmission

For a satellite's data transmission to be successful, it must occur within the visible window between the satellite and the ground station. Specifically, once the satellite enters the receiving range of the ground station and the antenna captures and tracks the satellite signal, the transmission process can begin. When the satellite moves out of range of the ground station, the data transmission process comes to an end (as illustrated in Fig. 2.4).

In contrast, the start time and end time of data transmission via relay resources are determined by the scheduling results of the relay satellite management department. If a relay satellite is available during a certain period, all EOSs can theoretically use it to transmit data.

2. Data reception switching time constraint for data transmission resources

The switching time for ground station data reception and relay data reception is collectively known as the data transmission resource switching time. This term refers to the time it takes to switch from providing data downlink services to one satellite to providing these services to another. This switching time constraint requires that the time required to switch between satellites be greater than the data transmission resource switching time.

However, due to this constraint, conflicts may arise in cases where multiple satellites enter the data reception range of a ground station, as depicted in Fig. 2.6. These conflicts occur due to multi-satellite data transmission access, which is limited by the data transmission resource switching time constraint.

Figure 2.6 is a schematic diagram that illustrates a conflict in multi-satellite data transmission access for a single data transmission resource. The horizontal axis indicates data transmission time, while the vertical axis distinguishes the transmission

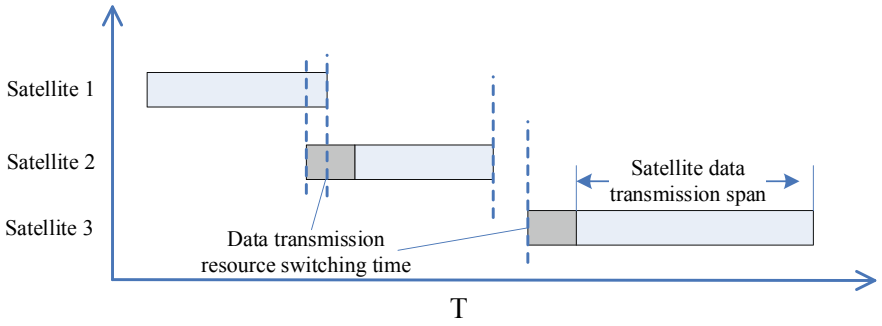


Fig. 2.6 Schematic of multi-satellite data transmission access conflict

periods of different satellites. As depicted in the figure, the conflict arises because the interval between the data transmission slots of satellite 1 and satellite 2 is less than the data transmission resource switching time. However, the data transmission slot of satellite 3 does not conflict with the data transmission slots of either satellite 1 or satellite 2.

From Fig. 2.6, it is evident that the switch time of the data transmission resource can be added to the satellite data transmission slot by preprocessing. As such, scheduling processes need only consider the constraint of relay satellite resource capacity. For the subsequent satellite and data transmission scheduling process presented in this book, we assume that the data transmission resource switching time has been added to the satellite data transmission time slot.

2.2.4 Optimization Objectives

Satellite task scheduling is a multi-objective optimization problem that researchers have approached by proposing different optimization objectives based on various application scenarios. These objectives share certain commonalities and can be summarized into four categories.

1. Task importance optimization objective

The task importance optimization objective requires the prioritization of more important earth observation tasks over less important ones if not all tasks can be completed. The importance of an earth observation task is reflected by its task priority, which is determined by the ground target priority. Therefore, the task importance optimization objective can be expressed as maximizing the total value of completed tasks.

2. Objectives for optimizing the timeliness of the tasks

The task timeliness optimization objective has two implications. First, it requires the prioritization of important earth observation tasks to be performed as early as

possible, and their observation data should be transmitted to the ground station as quickly as possible. Second, tasks with specific observation timeliness demands (as defined in Definition 2.4) should be scheduled and completed before their deadline as far as possible.

3. Task completion optimization objective

For an earth observation requirement with a demand for target group observation (as defined in Definition 2.4), if the EOSs observe only six out of eight targets included in the target group, the task for the target group is not considered 100% complete. Similarly, for an earth observation requirement with a demand for time resolution (as defined in Definition 2.4), if the demand is "at least one observation every six hours" and there is a 6-h interval between two consecutive observations, the task is also not considered 100% complete. The task completion optimization objective is to achieve 100% completion for as many earth observation tasks as possible.

4. Resource utilization optimization objective

The resource utilization optimization objective aims to minimize the consumption or occupation of resources by EOSs during the performance of earth observation tasks.

2.3 Difficulties and Challenges in the Scheduling of EOSs Task

Compared with the general planning and scheduling problem, the EOS task scheduling problem has the following notable characteristics.

2.3.1 Oversubscribed Problem Characteristics

The term "overconstrained" refers to a situation in which there are too many conflicting constraints, leading to an infeasible solution to the problem. This is the opposite of the phenomenon of "underconstrained," in which there are not enough constraints between variables, resulting in a problem with more than one feasible solution [5]. The earth observation satellites (EOSs) scheduling problem falls into the category of "overconstrained." On the one hand, this is due to the fact that the development of EOS resources is far from meeting the growing demand for remote sensing data for various applications in different sectors. The demand for satellites by various departments is always relatively excessive, making the EOS resources always in a relatively scarce state. On the other hand, the operation of earth observation satellites is strictly limited by various constraints, leading to the overconstrained phenomenon.

Of course, the overconstrained problem is a relative concept, and if certain constraints of the original problem are relaxed, the resulting new problem may

have a feasible solution. However, for the earth observation satellite resource scheduling problem, the constraints related to both satellite resources and data transmission resources are physical constraints that cannot be relaxed. Therefore, only the constraint that all tasks must be completed can be relaxed; i.e., some of the earth observation requirements can be dropped, although this relaxation usually leads to an underconstrained situation again. At this point, it is necessary to employ a feasible solution with some optimization objectives such that the scheduling result is an optimal or user-satisfactory solution.

Barbulescu et al. have identified the concept of oversubscribed problem (OSP), which is characterized by a scarcity of resources and a larger number of tasks than the carrying capacity of the resources [110]. In such problems, tasks have priorities and optional time windows, and the objective function usually relates to task priorities. Many applications, such as space telescope planning, earth observation satellite scheduling, ground station scheduling, aircraft loading scheduling, and space shuttle payloads scheduling, exhibit oversubscribed scheduling characteristics. The EOSs task scheduling problem is a prime example of the oversubscribed scheduling problem, where the aim of scheduling is to choose a subset of earth observation tasks that meet all the satellite constraints while maximizing overall benefits.

2.3.2 Non-fully Fungible Feature of Resources

In general scheduling problems, an activity can be accomplished by multiple alternative resources. The resource scheduling problem for earth observation satellites also shares this feature. For instance, a particular earth observation task can be performed by either the sensors carried by satellite A or the sensors carried by satellite B. A specific satellite can transmit data to either ground station α or to ground station β. However, the difference lies in the fact that two satellites cannot have identical orbits, and two ground stations cannot have identical geographic locations. Therefore, the available resources in the EOSs scheduling problem are not entirely fungible. For example, if satellite A cannot perform an observation during a specific visible time window between a given ground target and satellite A due to various reasons such as performing another earth observation task or low energy, we cannot simply replace satellite A with satellite B. This is because there may not be a visible time window between satellite B and the ground target at that moment. Additionally, the performance of the sensors carried by satellite A and satellite B may not be identical, since there could be differences in their payload types and resolution. This feature also contributes to the non-fully fungible nature of resources.

2.3.3 Scheduling for Heterogenous Satellite Resources

The satellites involved in task scheduling usually include both agile and non-agile satellites, and the sensors they carry include: optical, infrared, multispectral, hyperspectral, ultra-wide spectrum, SAR, electromagnetic detection, and many other types. Different types of satellites have different capabilities and are constrained by different orbits, satellite platforms, and satellite payloads. The EOSs task scheduling system needs to coordinate various heterogeneous satellite resources so as to maximize the overall observation benefits. The scheduling process exhibits typical multi-modal, nonlinear, and nonconvex optimization characteristics, and requiring multi-modal optimization techniques.

2.3.4 Diverse Types of Earth Observation Demands

As the application level of earth observation satellites deepens, the requirements for satellite observation are becoming diverse. In addition to the traditional observation targets, complex observation requirements such as area target coverage, ocean moving target search and tracking, ground target 3D reconstruction (stereo imaging demands), a group of targets observation, and regular refresh of the situation (temporal resolution demands) have gradually emerged. These factors need to be comprehensively considered in planning and scheduling, which also brings new challenges to EOSs task scheduling.

2.3.5 Diversity of Data Transmission Modes

Satellites can transmit data to ground stations through two modes: real-time transmission and playback transmission [24]. The real-time transmission mode allows for instantaneous data transfer, with the satellite obtaining observation data through its sensor and transmitting it directly to the ground station via an onboard data transmission antenna. In this mode, the satellite must be in simultaneous view of both the ground station and the target, and the process does not occupy onboard memory. On the other hand, in the playback mode, the satellite first observes the ground target, records the data in onboard memory, and then transmits the corresponding data to a ground station when the satellite passes over the receiving range. This is a delayed transmission mode that requires onboard memory usage. Figure 2.7 illustrates the process of real-time and playback transmission modes.

Observation tasks using the real-time transmission mode generally have higher priority due to their greater timeliness. The use of real-time transmission can significantly improve the timeliness of observation data. However, real-time transmission and playback transmission are contradictory, as in the real-time transmission mode

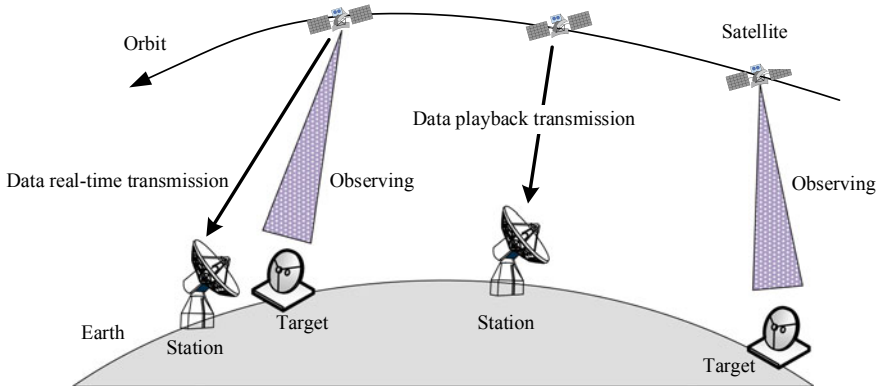


Fig. 2.7 Schematic diagram of the real-time transmission and playback transmission process

the data stored in onboard memory cannot be transmitted to ground at the process, resulting in reducing the timeliness of the observation data in the onboard memory. Overuse of real-time transmission mode can occupy onboard memory for extended periods, increasing the risk of the satellite running out of memory and being unable to perform earth observations. Therefore, scheduling of real-time transmission and playback transmission actions needs to be reasonably determined based on earth observation task requirements and their distribution.

2.3.6 Onboard Memory Occupation and Release

During earth observation, a satellite records observation data on its onboard memory and transmits it to a ground station when it enters the receiving range. The onboard memory occupancy is released through playback transmission [7]. Figure 2.8 illustrates the relationship between onboard memory occupancy/release process and satellite actions.

The challenge lies in predicting the amount of data transmitted in a single data transmission activity, which depends on the scheduling results of observation and data transmission activities. Since it is uncertain which observation tasks will be performed until the scheduling computation is completed, accurately predicting satellite memory occupancy is difficult, adding complexity to problem resolution.

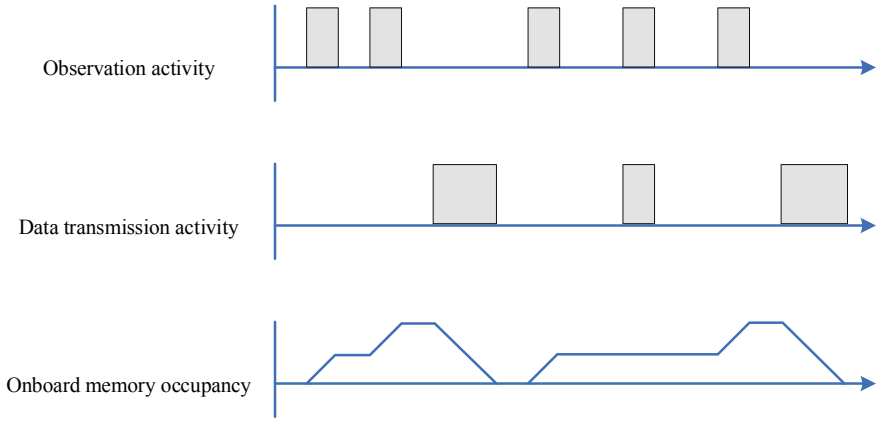


Fig. 2.8 Schematic of the onboard memory occupancy and release process

2.3.7 Uncertainty Included in EOSs Task Scheduling

The scheduling of earth observation satellites (EOSs) is subject to certain uncertainties that can lead to additional difficulties in the scheduling process. These uncertainties are mainly related to the characteristics of the observation tasks and the dynamics of satellite resources.

1. Earth observation task uncertainty

The scheduling of earth observation satellites (EOSs) is subject to various uncertainties, including the new tasks random emerge, changes in task attributes, and cancelations [125]. For instance, when a new batch of tasks arrives, the existing earth observation programme must be adapted to accommodate them, ensuring maximum overall benefits. This requires effective coordination between new and existing tasks, with new tasks integrated into the original observation programme.

In other words, when new tasks are added to the programme, it is essential to optimize the programme to incorporate these changes, maximizing its effectiveness. Effective coordination can help to manage uncertainties in the scheduling process, ensuring that new tasks are integrated seamlessly and maximizing the overall benefits of the EOSs.

2. Characteristics of the dynamics of satellite resources

The scheduling of earth observation satellites (EOSs) is subject to various uncertainties, including temporary failures due to satellite malfunction in the complicated space environment where they operate. Typically, a failed satellite is only part of the resources of multiple EOSs, resulting in a temporary inability to perform earth observation tasks during the failure period. This situation requires effective coordination of available satellite resources to minimize the impact of the failure on the scheduling process.

When a failed satellite is recovered, a unified approach should be taken in considering existing and new satellite resources to maximize the fulfillment of user observation requirements. This involves optimizing the scheduling process to ensure the best use of available resources, taking into account the specific needs of the observation tasks. Effective coordination can help to minimize the impact of temporary failures and ensure that the EOSs continue to operate efficiently in the challenging space environment.

3. Available capacity agnostic of onboard memory for some special satellites [125]

Certain types of earth observation satellites, such as electromagnetic detection satellites, have unique characteristics that pose challenges in the task scheduling process. Specifically, the amount of data that sensors acquire during a single switch-on activity is highly dependent on the characteristics and intensity of the ground electromagnetic environment. Given that the electromagnetic environment is usually time-varying or random, it is not possible to precisely determine the amount of data acquired by the sensors during an observation, making it impossible to estimate the available capacity of onboard memory at a given moment.

In contrast, general earth observation satellites (e.g., optical satellites) acquire a predetermined amount of data based on the switch-on duration and working mode of the sensors, providing deterministic information for ground scheduling. As a result, the uncertainty factor related to onboard memory capacity is a significant difference between the task scheduling of electromagnetic detection satellites and general earth observation satellites. This uncertainty factor presents a new challenge in maximizing the use of onboard memory capacity, which requires advanced scheduling algorithms that can adapt to the variability of the electromagnetic environment.

2.3.8 Complicated Optimization Objectives

The optimization objective in a general planning and scheduling problem is relatively simple. For example, in the single machine scheduling problem, the objective is usually to complete the maximum number of artifacts per unit of time or to complete all artifacts in the shortest total time. In the multi-dimensional backpack problem, the objective is to maximize the value of the items placed into the backpack. However, the earth observation satellite task scheduling problem is a typical multi-objective optimization problem with potentially conflicting objectives that need to be optimized simultaneously. This adds to the difficulty of the optimization computation. In engineering practice, to reduce the computational burden, a weighting approach is usually used to transform multiple optimization objectives into a single optimization objective. However, this approach introduces another challenging problem: how to set the weights of each optimization objective. Although aerospace experts have found weight vectors that yield user-satisfactory solutions based on their experience, explicit rules for setting the weights of each optimization objective function have not been established [6].

Chapter 3
Model and Method of Ground-Based Centralized EOS Task Scheduling

Ground-based centralized satellite task scheduling is a well-studied problem in the field of satellite operations. This method assumes that a single operation and control center manages all satellite resources and establishes a mathematical model for earth observation satellites (EOSs) and their resource scheduling. A centralized optimization algorithm is then used to solve the problem. The ground-based centralized satellite task scheduling problem can be divided into two main subproblems: the earth observation task scheduling problem and the satellite data transmission scheduling problem. The former involves determining when the satellite should use its sensors to observe which targets, while the latter involves deciding when the satellite should transmit observation data to which ground station. The two problems have essential relationship due to constraints on the use of satellite sensor and resource constraints such as satellite energy and onboard memory storage.

In addition to these fundamental subproblems, more complex scenarios, such as area targets and ocean moving targets, require additional consideration when developing satellite observation task scheduling methods. Finally, a learnable centralized satellite task scheduling method is proposed, which can continuously improve the performance of the optimization algorithm through online learning of historical scheduling results.

3.1 Problem Description and Analysis

3.1.1 Centralized EOS Task Scheduling Problem

Earth observation satellites are intended to orbit the earth, and their onboard sensors are programmed to capture data related to specific ground targets as they pass over them. This observation data is temporarily stored in onboard memory and then transmitted to a ground station using either real-time or playback transmission mode, depending on the availability of a ground station's reception range.

H. Chen et al., *Earth Observation Satellites*,
https://doi.org/10.1007/978-981-99-3565-9_3

The ground-based centralized satellite task scheduling problem is a fundamental issue in satellite operations that can be mathematically formulated as follows:

1. Given a scheduling time horizon $w_{\text{schedule}} = [t_{\text{B}}, t_{\text{E}}]$. t_{B} and t_{E} are the start time and end time of the scheduling time horizon. All scheduling elements are limited to the given scheduling period.
2. Given a EOSs set SAT. For $\forall s \in \text{SAT}, s \equiv \langle \text{MODE}^s, \text{memy}^s, \text{trans}^s_{i,j}, \text{pre}^s, \text{post}^s, \Delta T^s_{\text{m}}, \Delta T^s_{\text{l}}, \Delta T^s_{\text{lc}}, \Delta T^s_{\text{ld}}, \omega^s \rangle$, where MODE^s is the set of observing modes of satellite s. For an optical satellite, MODE^s is the imaging roll angle of the satellite s. For a synthetic aperture radar (SAR) satellite, MODE^s is the set of parameters of SAR sensors; for electromagnetic detection satellites, the working mode is the parameters of the onboard electromagnetic signal receiver. memy^s is the total capacity of the onboard memory of satellite s. If the onboard memory capacity is full (also known as onboard memory overflow), the satellite cannot continue its earth observation task using the playback transmission mode. $\text{trans}^s_{i,j}$ is the transition time from working mode i to working mode j, in which $i, j \in \text{MODE}^s$. pre^s is the power-on preparation time of the satellite s. post^s is the power-off stabilization time of the satellite s. ΔT^s_{m} and ΔT^s_{l} denote the minimum and maximum single observing time of satellite s. ΔT^s_{lc} is the longest cumulative working time of the satellite s in a single circle. ΔT^s_{ld} is the longest cumulative working time of the satellite s in a single day. ω^s is the data acquiring/transmission ratio of the satellite s, i.e., the ratio of data generated per unit time from onboard earth observation sensor to the data transmitted per unit time of the data transmission payload. ω^s characterizes the data transmission capability of the satellite.
3. Given an observation target set TARGET. For $\forall \text{tar} \in \text{TARGET}$, $\text{tar} \equiv \langle \text{lon}_{\text{tar}}, \text{lat}_{\text{tar}}, \text{rot}_{\text{tar}} \rangle$ in which, lon_{tar} and lat_{tar} are the target's longitude and latitude. rot_{tar} is the maximum number of effective observations to the target tar. There is no furthermore reward if the observation times of tar are more than rot_{tar}. For most of the targets, $\text{rot}_{\text{tar}} = 1$, which means the target only needs to be observed once. For targets with stereo imaging demands, the target should be observed from different angles, and $\text{rot}_{\text{tar}} > 1$.
4. An EOS fly along a certain predefined trajectory and can only roll around and observe targets that are vertically distributed along its subsatellite trajectory. Therefore, the performed observation event will exactly fill its visible time window (VTW). We consider each VTW as a meta-task (short for task, denoted as TASK), which is the minimal unit for scheduling. For $\forall k \in \text{TASK}, k \equiv \langle s_k, \text{mod}^s_k, \psi_k, t^k_{\text{b}}, t^k_{\text{e}}, \text{tar}_k, \text{circle}_k \rangle$. Where, $s_k \in \text{SAT}$ denotes the satellite that performs task k in the VTW. $\text{mod}^s_k \in \text{MODE}^s$ denotes the working mode that satellite s_k takes to perform task k. ψ_k is the priority of task k that denotes the reward when k is performed. t^k_{b} and t^k_{e} are the start time and end time of the corresponding VTW. $\text{tar}_k \in \text{TARGET}$ denotes the task k corresponds to the ground target tar_k. $\text{circle}_k \in \mathbb{N}$ indicates the orbital circle of the satellite in which the current task is located.

5. Given a set of data transmission resources GRD. $\forall gs \in GRD$, $gs \equiv \langle lon_{gs}, lat_{gs}, alti_{gs}, range_{gs} \rangle$. Where, lon_{gs} is the longitude of the ground station gs, and lat_{gs} is the latitude of the ground station gs. $alti_{gs}$ is the altitude of the ground station gs. $range_{gs}$ is the antenna reception range of the ground station gs.
6. After the calculation of the visible time window from the satellites to the ground stations, the set of data transmission activities is obtained, and denoted as DT. $\forall djob \in DT$, $djob \equiv \langle s_{djob}, g_{djob}, t_{db}^{djob}, t_{de}^{djob}, livedt_{djob} \rangle$. In which, $s_{djob} \in SAT$ is the satellite involved in this data transmission. $g_{djob} \in GRD$ is the ground station involved in this data transmission. $\left[t_{db}^{djob}, t_{de}^{djob}\right]$ is the time window of the data transmission. $livedt_{djob}$ is the transmission mode of the data transmission activity djob. If $livedt_{djob} = 1$ indicates the real-time transmission mode, otherwise indicates the playback transmission mode. The amount of data transmitted in a data transmission activity is usually related to the length of the visible time window $(t_{de}^{djob} - t_{db}^{djob})$. Typically, a satellite cannot transmit all the data in the onboard memory in a single data transmission process. Thus, a selection of the data block from onboard memory to be transmitted is required. It is assumed that the EOS can automatically select the observation data of high-priority target and transmit it to ground first.
7. Given the decision variables x_k^s, $xg_{djob}^{s,g}$, and $xgl_{djob}^{s,g}$, where, $x_{task}^s \in \{0, 1\}$. If $x_k^s = 1$ denotes the observation task k will be performed by the satellite s, otherwise, it will not be performed in this VTM. Similarly, $xg_{djob}^{s,g} \in \{0, 1\}$. If $xg_{djob}^{s,g} = 1$ indicates that the data transmission activity djob of the satellite s will be received by the ground station g, otherwise, it will not be performed. $xgl_{djob}^{s,g}$ is an auxiliary decision variable. It makes sense, when $xg_{djob}^{s,g} = 1$. $xgl_{djob}^{s,g} \in \{0, 1\}$, the $xgl_{djob}^{s,g} = 1$ denotes real-time transmission mode for the data transmission activity djob, otherwise, playback transmission mode. $k \in TASK$, $s \in SAT$, $djob \in DT$, and $g \in GRD$.

The purpose of ground-based centralized satellite task scheduling is to find reasonable values of decision variables that enable the satellite earth observation programme to maximize the overall benefits while satisfying all satellite constraints.

3.1.2 Scheduling Strategy Analysis

In Sect. 3.1.1, the decision variables are defined, where x_{task}^s determines whether the corresponding meta-task is performed, and $xg_{djob}^{s,g}$, the $xgl_{djob}^{s,g}$ determines whether the corresponding data transmission activities are performed and in which transmission mode. The ground-based centralized satellite task scheduling problem is composed of two interdependent subproblems: the satellite earth observation task scheduling problem and the satellite data transmission scheduling problem. During the scheduling process, the satellite's available memory capacity decreases as it completes earth observation tasks and stores observation data in onboard memory.

Once the memory is full, the satellite can no longer continue earth observation. Similarly, the satellite's available memory capacity is restored as it performs data transmission and downlinks data to ground stations. When the satellite operates in real-time transmission mode, existing data in onboard memory cannot be transmitted in real-time mode. Thus, solving these two subproblems together is necessary to achieve the optimal solution from a global perspective, although it requires a larger computational effort.

Both the satellite earth observation task scheduling problem and the satellite data transmission scheduling problem have been shown to have NP-hard computational complexity, and integrating the two subproblems results in an exponential increase in computational effort, slower algorithm convergence, and complex constraint nesting relations that need to be addressed. In applications, the algorithm's computation time is limited, necessitating a scheduling algorithm that can provide a relatively satisfactory solution in a short period.

To balance solution optimization and computational burden, two mainstream approaches have emerged. The first approach, known as the progressive optimization strategy, optimizes the satellite observation task and data transmission resources separately and iteratively for relatively large problem sizes. The second approach, known as the global optimization strategy, considers the two subproblems together and searches for the overall optimal solution of the problem from a global perspective for relatively small problem sizes. The subsequent sections will introduce these two approaches separately.

3.2 Centralized EOS Task Scheduling Method Under a Progressive Optimization Strategy

The centralized satellite task scheduling method based on a progressive optimization strategy employs a "divide-and-conquer" strategy. Initially, the two subproblems, satellite observation task scheduling and satellite data transmission scheduling, are solved sequentially, and subsequently, the interdependency between them is studied, leading to a scheduling scheme for the entire problem.

As per the above analysis, the centralized satellite task scheduling method based on a progressive optimization strategy can be divided into three scheduling phases (e.g., Fig. 3.1). The first phase is the earth observation task scheduling, which considers only the constraints and optimization objectives related to the earth observation task (ignoring factors like observation data transmission and assuming infinite onboard memory capacity). The output of this stage is the satellite observation programme.

The second phase is the satellite data transmission scheduling, which considers the data transmission resources and the satellite observation programme generated in the first phase. This phase considers the constraints associated with data transmission and outputs the satellite data transmission programme.

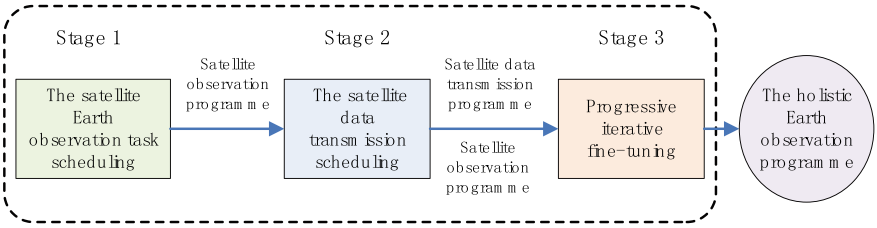


Fig. 3.1 Scheduling process of progressive optimization strategy

The third phase considers the interdependency between earth observation task scheduling and data transmission resource scheduling. It focuses on the constraints connecting the previous two stages, such as memory capacity constraints, and employs a progressive iterative fine-tuning method comprising the stochastic hill-climbing algorithm and constraint propagation mechanism to enhance the scheduling results of the first two stages. The output of the third phase is the holistic earth observation programme.

3.2.1 Centralized Scheduling for EOS Observation Tasks

1. CSP-based scheduling models and solution methods

Considering the payload characteristics and constraints of earth observation satellites, we have developed a constraint satisfaction problem (CSP) model to represent the scheduling problem. The CSP model is widely used as a centralized scheduling model for earth observation tasks.

In this model, we have selected the task importance (as described in Sect. 2.2.4) as the objective function. The task importance is expressed as follows:

$$V_{\text{imp}}^{\text{ob}} = \max \sum_{s \in \text{SAT}} \sum_{k \in \text{TASK}} x_k^s \cdot \psi_k \tag{3.1}$$

Equation (3.1) indicates that the scheduling model aims to achieve the maximum number of more important earth observation tasks; i.e., the optimization objective is to maximize the task importance, while satisfying various constraints, as described in detail below.

(C1) Satellite single power-on time constraint. The duration of each task's power-on time must be within the range of the minimum and maximum single observing time.

$$\begin{aligned} &x_{\text{task}}^s \cdot \Delta T_{\text{m}}^s \le x_{\text{task}}^s \cdot \left(t_{\text{e}}^{\text{task}} - t_{\text{b}}^{\text{task}}\right) \le x_{\text{task}}^s \cdot \Delta T_{\text{l}}^s, \\ &\forall s \in \text{SAT}, \ \forall \text{task} \in \text{TASK} \end{aligned} \tag{3.2}$$

In which, the $\Delta T_{\mathrm{m}}^{s}$ and $\Delta T_{\mathrm{l}}^{s}$ are minimum and maximum single observing time of satellite s, respectively.

(C2) Satellite working mode switching time constraint. Each earth observation task requires the onboard sensor to operate in a specific working mode. To ensure the sensor operates efficiently, the time interval between any two consecutive observations (with different working mode) of the same satellite must be longer than or at least equal to the needed transition time from one working mode another.

$$\begin{aligned} &x_{k1}^{s} \cdot x_{k2}^{s} \cdot \left(t_{\mathrm{b}}^{k2} - t_{\mathrm{e}}^{k1} + \mathrm{pre}^{s} + \mathrm{post}^{s}\right) \leq x_{k1}^{s} \cdot x_{k2}^{s} \cdot \mathrm{trans}^{s}{}_{\mathrm{mod}\ _{k1},\ \mathrm{mod}\ _{k2}}, \\ &\forall s \in \mathrm{SAT},\ \forall k1, k2 \in \mathrm{TASK},\ t_{\mathrm{b}}^{k2} > t_{\mathrm{b}}^{k1} \end{aligned} \tag{3.3}$$

In which, the pre^{s} is the power-on preparation time of the satellite s and the post^{s} is the power-off stabilization time of the satellite s. $\mathrm{trans}_{i,j}^{s}$ is the transition time from working mode i to working mode j, where $i, j \in \mathrm{MODE}^{s}$.

(C3) Sensor maximum cumulative working time constraint in a single circle. Within one orbital circle (one orbital circle of the satellite around the earth), the accumulated satellite sensor working time cannot exceed the longest cumulative working time in a single circle.

$$\sum_{\substack{k \in \mathrm{TASK} \\ \mathrm{Circle}_k = \mathrm{cirnum}}} x_{k}^{s}\left(t_{\mathrm{e}}^{k} - t_{\mathrm{b}}^{k} + \mathrm{pre}^{s} + \mathrm{post}^{s}\right) \leq \Delta T_{\mathrm{lc}}^{s}, \quad \forall s \in \mathrm{SAT},\ \forall \mathrm{cirnum} \in \mathbb{N} \tag{3.4}$$

In which, the $\Delta T_{\mathrm{lc}}^{s}$ is the longest cumulative working time of the satellite s in a single circle.

(C4) Sensor maximum cumulative working time constraint in a single day. The cumulative sensor working time of the satellite within a single orbit cannot exceed the longest working time allowed for a single orbit. Furthermore, within a 24-h period, the cumulative sensor working time cannot exceed the maximum working time allowed for a single day.

$$\begin{aligned} &\sum_{k=\mathrm{task\_}p}^{\mathrm{task\_}q} x_{k}^{s} \cdot \left(t_{\mathrm{e}}^{k} - t_{\mathrm{b}}^{k} + \mathrm{pre}^{s} + \mathrm{post}^{s}\right) \leq \Delta T_{\mathrm{ld}}^{s}, \\ &\forall s \in \mathrm{SAT},\ t_{\mathrm{e}}^{k_p} - t_{\mathrm{b}}^{k_q} \leq 24(h),\ t_{\mathrm{b}}^{k_q} > t_{\mathrm{b}}^{k_p} \end{aligned} \tag{3.5}$$

where $\Delta T_{\mathrm{ld}}^{s}$ is the longest cumulative working time of the satellite s in a single day. From Eq. (3.5), it can be seen that a sliding time window can be used to verify if C4 is satisfied within a 24-h period.

(C5) The constraint of meta-task performing number. Each ground target has a maximum limit on the number of observations, after which further observations would not yield any significant benefits. Therefore, no target needs to be observed more than this limit. The number of observations varies for different types of targets, with point targets typically having an upper limit of one observation to complete the mission. For targets with stereo imaging requirements, the upper limit is usually set to 2–4 observations.

$$\sum_{\substack{\text{task}\in\text{TASK}\\ \text{Tar}_{\text{task}}=\text{tar}}} x_{\text{task}}^{s} \le \text{rot}_{\text{tar}}, \quad \forall s \in \text{SAT}, \ \forall \text{tar} \in \text{TARGET} \tag{3.6}$$

(C6) Satellite capability constraint. Satellite sensors can perform only one task with a specific working mode at any given moment.

$$x_{k1}^{s} \cdot x_{k2}^{s} \cdot \left(t_{\text{b}}^{k2} - t_{\text{e}}^{k1}\right) \cdot \left(t_{\text{e}}^{k2} - t_{\text{b}}^{k1}\right) \ge 0, \ \forall s \in \text{SAT}, \ \forall k1, k2 \in \text{TASK} \tag{3.7}$$

Based on the formulation of the CSP model presented above, it is evident that the centralized satellite earth observation task scheduling problem can be reduced to the multi-dimensional 0–1 knapsack problem by considering only a subset of the constraints. The multi-dimensional 0–1 knapsack problem is a classical NP-hard problem according to algorithmic complexity theory [127], and no polynomial time solution method for this type of problem is known. Genetic algorithms are heuristic algorithms that simulate the process of biological reproduction and evolution in nature, and they have been widely employed to solve complex function optimization problems, combinatorial optimization problems, planning and scheduling problems, among others [128]. Genetic algorithms have several features, including implicit parallelism, less requirement for prior knowledge, and neighborhood-based search, which make them suitable for solving the centralized earth observation task scheduling problem.

Thus, we propose the Satellites Observation Task Centralized Scheduling Algorithm (SOCSA) based on the elite archive genetic algorithm. The main steps of the SOCSA are as follows:

Algorithm name: SOCSA
Input: meta-task set TASK, satellite set SAT
Output: satellite observation programme $\left\{x_{\text{task}}^{s}\right\}$, task $\in$ TASK, $s \in$ SAT

(continued)

(continued)

begin
1 Code the problem, initialize the population, and set the second population to an empty set.
2 Select the parent individuals father1 and father2 from the population, perform the crossover operation to generate offspring individuals offspring1 and offspring2. Adding them to the second population.
3 If the ratio of individuals in the second population to those in the population is smaller than the crossover probability, goto 2.
4 Select individuals from the second population to perform the mutation operation.
5 If the ratio of muted individual to un-muted one is smaller than the mutation probability, goto 4.
6 Select some individuals from the population and the second population to form the next generation population using the selection operator, and save elitisms.
7 Set the second population to the empty set.
8 Obtain constraints of the satellite set SAT and do constraints handling for the population.
9 If the exit condition is not satisfied, goto 2.
10 decode, output satellite observation programme $\left\{x_{\text{task}}^{s}\right\}$, task $\in$ TASK, $s \in$ SAT.
end

SOCSA encodes the problem to generate multiple individuals (chromosomes), each representing a feasible satellite observation solution programme. These individuals form the population, and SOCSA simulates biological evolution by applying crossover, mutation, and selection operators to the parent population to generate individuals for a new offspring population. For infeasible solutions generated by the evolutionary process, SOCSA employs constraint handling (statement 8). It assigns a fitness value to each individual, with the task importance objective $V_{\text{imp}}^{\text{ob}}$ serving as the fitness criterion. The higher the task importance objective, the greater the individual fitness. The iterative process continues to retain individuals with high fitness values (i.e., the more optimal satellite observation programme) and eliminates those with low fitness values to obtain a satisfactory scheduling result. The following sections describe the SOCSA genetic operator design and constraint handling process in detail.

(1) Problem coding of SOCSA

The problem coding for SOCSA is illustrated in Fig. 3.2. We utilize equal-length 0–1 coding to construct the chromosome for representing the satellite observation programme, with each gene locus denoting the decision variable of an observation task. If $x_k^s = 1$ denotes the observation task k will be performed by the satellite s, otherwise, it will not be performed in this VTM, where $t \in$ TASK, $s \in$ SAT. The subsequent genetic operator operations and fitness value calculations are based on this problem encoding.

(2) Crossover operator design of SOCSA

For the coding characteristics of the centralized satellite earth observation task scheduling problem, we have designed a multi-point crossover operator. This operator selects a crossover point (CP) gene in the meta-task sequence of each satellite and performs the crossover operation, as shown in Fig. 3.3. The parent individual,

father_1, is obtained by roulette selection, where the probability of each chromosome being selected is proportional to its fitness value. The second parent individual, father_2, is obtained by random selection. Assuming the current chromosome contains five chromosome segments (a sequence of meta-tasks of five satellites), we generate a crossover point for each chromosome segment to achieve a multi-point crossover operation.

(3) Mutation operator design of SOCSA

The mutation operator employs a stochastic reverse method at a single point. It randomly selects a gene related to an observation task and alters its execution state. Figure 3.4 depicts the mutation process. In Fig. 3.4a, the mutation operation causes the satellite to perform the task represented by the mutation point (MP) gene, whereas the task would not have been performed before the mutation. Conversely, Fig. 3.4b illustrates the opposite scenario where the mutation operation results in the satellite not performing the MP gene task.

In population-based optimization algorithms, such as genetic algorithms, the crossover and mutation operators generate new individuals that may contain infeasible solutions. Therefore, constraint handling mechanisms are necessary to ensure the feasible solutions are generated. In particular, the constraint handling process in

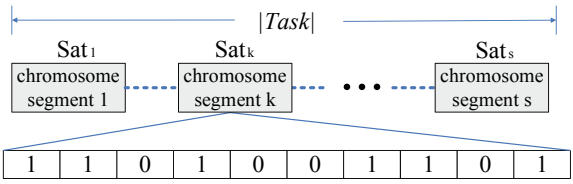


Fig. 3.2 Schematic diagram of SOCSA coding

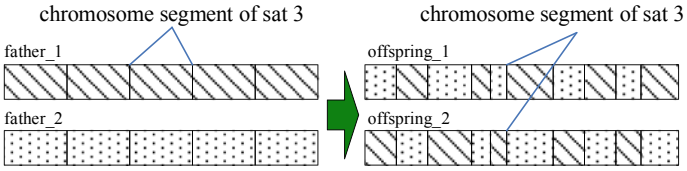


Fig. 3.3 Schematic diagram of the operation of the SOCSA crossover operator

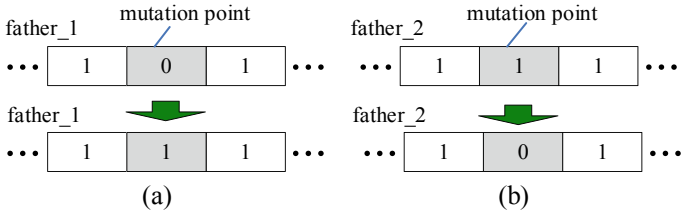


Fig. 3.4 Schematic diagram of the operation of the SOCSA mutation operator

the crossover and mutation operators focuses on constraints (C1), (C2), and (C6). If a gene in the current chromosome conflicts with the gene at the crossover or mutation point, the meta-task represented by the conflicting gene is canceled to prevent infeasible solutions. After the generation of the population in each iteration, constraints (C3)–(C5) are then checked after population generated in each generation. The benefits of this treatment are:

- Improve the efficiency of crossover and mutation operators that need to be called many times, and speed up the operation of the algorithm.
- Let the crossover and mutation operators search solutions in the feasible region and the infeasible region under constraints (C3)–(C5) to increase the search range and increase the probability of optimal solution acquisition.

A detailed discussion of the constraints handling approach for (C3)–(C5) will be provided in the section entitled "Constraint Handling in SOCSA."

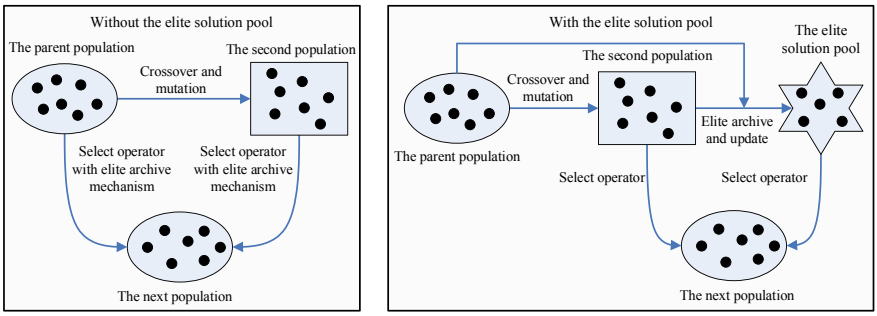
(4) Select operator design of SOCSA

The elite archive mechanism has become a popular selection operator in many genetic algorithm applications [128]. This mechanism plays a crucial role in enhancing the proximity of the approximate solution set to the optimal solution of the problem, while simultaneously preserving the diversity of the population. There are two primary methods to achieve this: the first involves considering the parent population together with the newly generated individuals after the evolutionary operation. In this approach, the selection process is based on fitness, and the next generation population is obtained by selecting individuals from both the parent population and newly generated population, rather than directly replacing the parent individuals with the new ones (as shown in Fig. 3.5a). The second approach involves constructing an external elite solution pool to maintain the top individuals obtained during the evolutionary process. For instance, the top-N individuals generated in each generation can be added to the elite solution pool, which is continuously updated, ensuring that it always contains the best Top-N individuals generated so far (as depicted in Fig. 3.5b). The elite archive mechanism is beneficial in addressing the issue of losing elite solutions during the evolutionary process due to the randomness of the selection operator. These two approaches are illustrated in Fig. 3.5.

In order to reduce the space and time cost of maintaining the elite solution during the SOCSA operation and increase the speed of the algorithm operation, an elite archive mechanism without elite solution pool is used in the SOCSA algorithm (as Fig. 3.5a), i.e., a roulette wheel selection operator with elite archive mechanism is designed. The probability of individual chromosomes in the parent population being selected into the offspring population is proportional to their fitness values, and a number of the best chromosomes are selected directly into the offspring population as elite solutions.

(5) Constraint handling of SOCSA

This section aims to discuss the handling of constraints (C3)–(C5) in the SOCSA algorithm, whereas constraints (C1), (C2), and (C6) have been dealt with during the



(a) Without the elite solution pool (b) With the elite solution pool

Fig. 3.5 Schematic diagram of the elite archive mechanism

crossover and mutation operations. Genetic algorithms commonly use two strategies for constraint handling: the penalty function (PF) method and the solution repair (SR) method. Among the two, the penalty function method is the most widely used [129]. Initially, this method was applied to constraint handling in mathematical optimization techniques. The idea is to create a penalty term for the chromosome that evaluates the extent of the current individual's constraint violation and apply it to the fitness value evaluation function. If the current individual violates the constraint, its fitness value is reduced. Essentially, the penalty function method converts a constrained optimization problem into an unconstrained optimization problem by penalizing infeasible solutions.

The problem solution space comprises of two regions: the feasible and infeasible regions (as depicted in Fig. 3.6). Several studies have demonstrated that the global optimal solution generally appears on the boundary between the feasible and infeasible regions. The advantage of the penalty function method is that it enables the search process to approach the global optimal solution from both the feasible and infeasible region directions [128]. During the evolution process of a genetic algorithm, a solution in the infeasible region (e.g., point A in Fig. 3.6) may contain more information and be closer to the optimal solution than that from the feasible region (e.g., point B in Fig. 3.6).

The penalty function method's unique bidirectional approximation process allows the genetic algorithm to converge faster. Compared to the solution repair method, the penalty function method does not require repeatedly calling complex solution repair algorithms, which reduces the computation time. In SOCSA, we adopt the penalty function method for constraint handling.

In SOCSA, the penalty function is designed to calculate the penalty term by multiplying the total degree of chromosome constraint violations (C3 ~ C5) by the corresponding penalty coefficient, as depicted in Eq. (3.8).

$$V_{\text{penalty}} = c_{\text{p}} \cdot \sum_{s \in \text{SAT}} \left(\omega_1 \cdot \Delta T^{s}_{\text{lc\_ex}} + \omega_2 \cdot \Delta T^{s}_{\text{ld\_ex}} + \omega_3 \cdot \Delta x^{s}_{\text{rot\_ex}} \right) \tag{3.8}$$

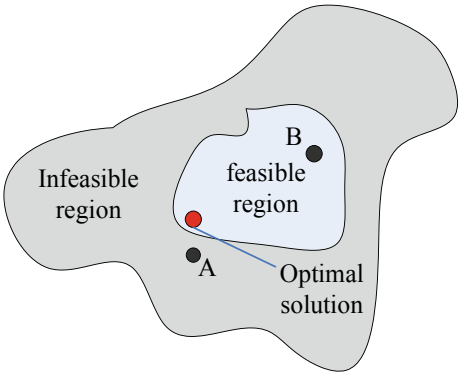


Fig. 3.6 Schematic diagram of feasible and infeasible regions

In which, the V_{penalty} is the penalty term. The $\Delta T^s_{\text{lc\_ex}}$ is the time that the satellite sensor cumulative working time at each orbital circle exceeds the ΔT^s_{lc} which is the longest cumulative working time of the satellite s in a single circle (for constraint C3). Similarly, $\Delta T^s_{\text{ld\_ex}}$ is the time that the satellite sensor cumulative working time within 24 h exceeds ΔT^s_{ld} which is the longest cumulative working time of the satellite s in a single day (for constraint C4). $\Delta x^s_{\text{rot\_ex}}$ is the total number of observations of the target that exceed the upper limit of the numbers of the target should be observed (for constraint C5). c_{p} ($c_{\text{p}} > 0$) is the penalty factor. ω_1, ω_2, and ω_3 are all constants $\in [0, 1]$, which are the weight coefficients of the constraint violation quantities, and $\omega_1 + \omega_2 + \omega_3 = 1$.

It is evident that if the chromosome (satellite observation programme) does not violate any constraint. The individual fitness value evaluation function of the SOCSA algorithm with the penalty term is shown in Eq. (3.9)

$$\text{Fitness} = \begin{cases} 0, & V_{\text{imp}} - V_{\text{penalty}} < 0 \\ V_{\text{imp}} - V_{\text{penalty}}, & \text{others} \end{cases} \tag{3.9}$$

One of the significant challenges in using penalty functions in any application is determining an appropriate penalty value [128]. Large penalty values discourage the algorithm from exploring infeasible regions and rapidly shift the search to the feasible region, increasing the likelihood of the algorithm being trapped in a local optimal solution. Conversely, low penalty values do not restrict the algorithm from searching the infeasible region most of the time, causing the algorithm's convergence process to slow down and the number of infeasible solutions in the population to increase.

Several applications have demonstrated that the value of penalty coefficient is related to the task distribution's character in scheduling problems [128]. However, in the centralized satellite earth observation task scheduling problem, the task distribution's nature varies significantly in different scheduling horizons, making it difficult to find a single penalty coefficient that suits all situations [125]. Furthermore, even different penalty coefficients are required in different stages of the genetic algorithm's evolution process. Initially, a low penalty coefficient is desirable to enable

the algorithm to search for optimal solutions in a larger region. When the algorithm is trapped in local optimal solution, the penalty coefficient should be reduced, and the algorithm is more likely to jump out of local optimal solution. In contrast, the penalty coefficient should be increased at the end of the evolution process to generate more feasible solutions in the population. Ideally, the algorithm should search for solutions in the boundary of the feasible region and the infeasible region to find the global optimum. If the search process remains in the feasible region for an extended period, changing the current penalty coefficient can redirect the search process to the infeasible region, and vice versa.

Based on the preceding analysis, a mechanism for a dynamic penalty function with adaptive adjustment of penalty coefficients can be devised, as illustrated in Fig. 3.7. The dynamic penalty function is an adaptive mechanism for adjusting penalty coefficients, analogous to the negative feedback approach. During the evolutionary process, the algorithm regulates the penalty coefficients in response to the current state of the population to manage the penalty strength in constraint handling and guide the population toward the anticipated direction of evolution.

The adaptive penalty factor $c_{\mathrm{p}}(\lambda)$ dynamically changes in the following way (where λ denotes the current evolutionary generation).

$$c_{\mathrm{p}}(\lambda+1)=\begin{cases} c_{\mathrm{p}}(\lambda)\cdot\theta_1, & \text{case \#1} \\ c_{\mathrm{p}}(\lambda)/\theta_2, & \text{case \#2} \\ c_{\mathrm{gen}}(\lambda+1), & \text{Other} \end{cases} \tag{3.10}$$

where the term "case #1" describes scenarios where either the best individual is infeasible or the best individual has remained unchanged for the last h generations, possibly due to being trapped in a local optimal solution. Conversely, "case #2" refers to situations in which the best individual has been feasible for the past h generations. The parameter $0<\theta_1,\theta_2<1$ and $\theta_1\neq\theta_2$ (to avoid cycling). The evolution penalty coefficient, denoted by $c_{\mathrm{gen}}(\lambda)$, is proportional to the pace of the evolutionary process in SOCSA and can be recursively defined by the following equation.

$$c_{\mathrm{gen}}(\lambda+1)=c_{\mathrm{p}}(\lambda)+c_{\mathrm{step}} \tag{3.11}$$

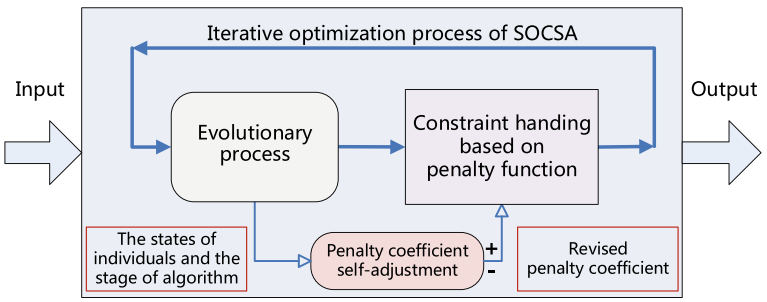


Fig. 3.7 Schematic diagram of the dynamic penalty function mechanism

where, c_{step} is the increment of penalty coefficient in evolutionary iteration, which can be expressed as:

$$c_{\text{step}} = \frac{c_{\max} - c_{\text{p}}(0)}{\lambda_{\max}} \tag{3.12}$$

In which, $c_{\text{p}}(0)$ is the initial penalty factor, and $\lambda_{\max}$ is the maximum number of iterations of the SOCSA. $c_{\max}$ is the upper bound of the preset penalty coefficient. And $c_{\max}$ can be empirically estimated.

2. Graph-based scheduling models and methods

In earlier studies, some researchers have proposed a graph-based model for formulating the satellite earth observation process. In this model, an earth observation satellite with roll capability orbits in space along a fixed trajectory, while the imaging roll angle and the satellite visible window for a given scheduling meta-task are predetermined. Consequently, the observation tasks have a temporal relationship with each other [11, 12]. Since satellites have limited capabilities for attitude adjustment, their imaging actions between transitions need to satisfy satellite constraints. To represent the observation tasks as vertices in a graph, a straightforward approach is to order the vertices according to the start time of observation. If two vertices (representing two tasks) are connected by an edge, the satellite can sequentially perform the two observation tasks in the order of their start times.

For an EOS sat (sat $\in$ SAT), each meta-task within its scheduling horizon can be represented as a vertex in a graph. The properties of the tasks, such as priority, working mode, start and end times of observation, are incorporated as attributes of the corresponding vertices.

Two virtual vertexes O_{S} and O_{T} are added to the graph G_{sat} that correspond to the starting vertex and the terminating vertex for a certain satellite task scheduling process, respectively. All the vertexes form the vertex set V_{sat}, and the vertexes are sorted by the order of the start time of the observation tasks.

The edge set of the graph G_{sat} is noted as E_{sat}. Assume that the starting vertex O_{S} and the terminating vertex O_{T} connect with all other real vertexes. For two actual vertices, if performing the two corresponding observation tasks satisfies the satellite working mode switching time constraint (C2), an edge is established between the two vertices. In other words, if the task represented by vertex A is executed, and the satellite can subsequently perform the task represented by vertex B. Therefore, a directed edge exists from vertex A to vertex B.

For $\forall A \in V_{\text{sat}}$, we have $\langle O_{\text{S}}, A\rangle \in E_{\text{sat}}$, the $\langle A, O_{\text{T}}\rangle \in E_{\text{sat}}$, $\langle O_{\text{S}}, O_{\text{T}}\rangle \notin E_{\text{sat}}$. And for $\forall A, B \in V_{\text{sat}}$, if the observation tasks represented by A and B satisfy the satellite working mode switching time constraint (C2), the graph has an edge $\langle A, B\rangle \in E_{\text{sat}}$. Thus, by examining whether an edge exists between each vertex and its subsequent vertices, we can determine the set of edges E_{sat} in the graph.

As the meta-tasks represented by the vertices have a time-ordered relationship, the graph $G_{\text{s}} = (V_{\text{sat}}, E_{\text{sat}})$ is an acyclic digraph. Therefore, every path from the virtual starting vertex O_{S} to the virtual terminating vertex O_{T} represents a candidate

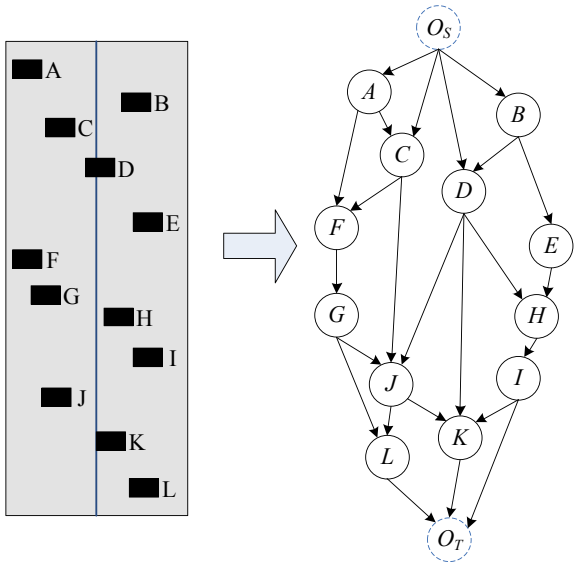


Fig. 3.8 A schematic diagram of the directed acyclic graph model for single satellite observation task scheduling

satellite observation programme for the EOS. A schematic representation of the directed acyclic graph model for observation task scheduling is presented in Fig. 3.8.

To handle multiple earth observation satellites, multiple directed graphs can be constructed, and these directed graphs collectively form the set of earth observation directed acyclic graphs [24], $G = \cup G_{\text{sat}}$.

Although the directed acyclic graph model can represent a single satellite earth observation task scheduling problem intuitively, whereas for multiple satellites observing task scheduling problem, the directed graph model makes the scheduling computation more difficult. Because the set of earth observation tasks from different satellites may contain the same targets in the scheduling horizon, the directed subgraphs for different EOS may have common vertexes in the graph G, as shown in Fig. 3.9.

Apparently, such common vertexes among subgraphs increase the complexity of the whole model and make it more difficult to figure out the satellite task scheduling results using the directed graph model. Thus, the directed acyclic graph model is more appropriate for representing single satellite earth observation task scheduling problems.

In the directed acyclic graph model for EOS task scheduling, the task importance criterion $V_{\text{imp}}^{\text{ob}}$ is still used as the objective function. In this case, we need to find the longest path (not the shortest path) in the graph, as it contains the most important meta-tasks. However, the longest path search problem is a typical NP-hard problem. The commonly used exact search algorithm is the graph path label updating algorithm, which updates the multiple labels information of the candidate paths continuously and removes paths that violate constraints (C1, C3–C6) as it searches for the path. If a meta-heuristic optimization algorithm, such as a genetic algorithm, is used

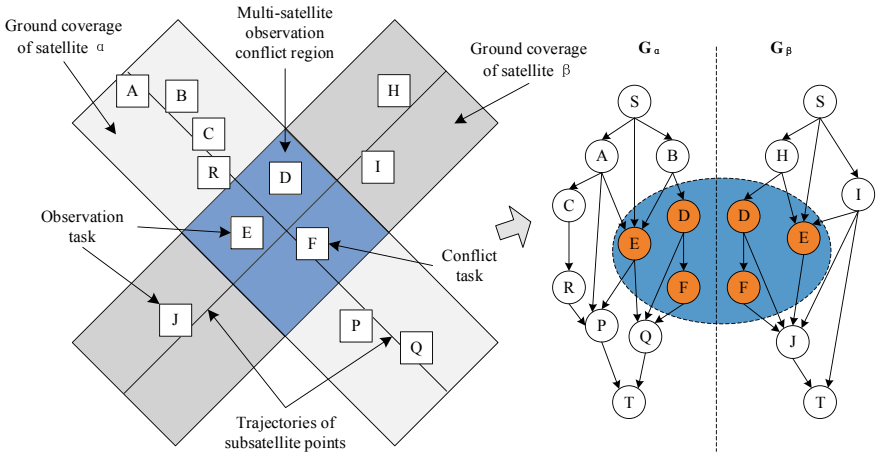


Fig. 3.9 A schematic diagram of the directed acyclic graph model for multiple satellite observation task scheduling

to solve the directed acyclic graph model for satellite observation task scheduling, the solution method is similar to the SOCSA algorithm described in the previous section and will not be repeated here.

3.2.2 Observation Task-Oriented Satellite Data Transmission Resources Scheduling

Task-oriented data transmission resource scheduling is the second stage of the centralized satellite task scheduling process that is based on a progressive optimization strategy. In the first stage, the satellite observation programme is generated, and in the second stage, the goal of satellite data transmission resource scheduling is to allocate data transmission resources efficiently for the satellite. The aim is to allow the satellite to transmit important earth observation data to the ground as much and as quickly as possible, while preventing the satellite's onboard memory from overflowing.

In this section, we present the satellite data transmission resource scheduling model and the corresponding solving algorithm for the established model.

1. Satellite data transmission resource scheduling model

Earth observation satellites orbit the earth, perform earth observation tasks, and store data in onboard memory. When the satellite obtains the service of data transmission resources, it transmits a portion of the data stored in the onboard memory to the ground station to complete a data transmission activity. The amount of data transmitted in one data transmission activity depends on the transmission rate of the

satellite data transmission equipment and the data transmission time (i.e., the data reception time of a ground station). Upon completion of the data transmission activity, the onboard memory releases space (as explained in Sect. 2.3.6). The definitions of the relevant concepts are given as follows.

(C7) Capacity constraints of data transmission resources. At any given moment, a single data transmission resource can only provide data transmission services to one satellite. If a ground station has multiple independent sets of data reception equipment, it can be considered as multiple data transmission resources located at the same position.

$$\begin{aligned} &\mathrm{xg}_{\mathrm{djob1}}^{s1,g} \cdot \mathrm{xg}_{\mathrm{djob2}}^{s2,g} \cdot \left(t_{\mathrm{db}}^{\mathrm{djob2}} - t_{\mathrm{de}}^{\mathrm{djob1}}\right) \geq 0, \\ &\forall s1, s2 \in \mathrm{SAT},\ \forall g \in \mathrm{GRD},\ \mathrm{djob1}, \mathrm{djob2} \in \mathrm{DT},\ t_{\mathrm{db}}^{\mathrm{djob2}} \geq t_{\mathrm{db}}^{\mathrm{djob1}} \end{aligned} \tag{3.13}$$

(C8) Satellite data transmission resource selection constraint. If a satellite can transmit the same batch of observation data through more than one data transmission resource simultaneously, it only needs to select one of them for data transmission at any given moment. Therefore, the satellite only needs to establish a communication link with one data transmission resource and transmit the data to the ground at that moment.

$$\begin{aligned} &\left[t_{\mathrm{db}}^{\mathrm{djob1}}, t_{\mathrm{de}}^{\mathrm{djob1}}\right] \cap \left[t_{\mathrm{db}}^{\mathrm{djob2}}, t_{\mathrm{de}}^{\mathrm{djob2}}\right] = \emptyset, \\ &\forall s \in \mathrm{SAT},\ \forall g1, g2 \in \mathrm{GRD},\ \mathrm{djob1}, \mathrm{djob2} \in \mathrm{DT},\ \mathrm{iff}\, \mathrm{xg}_{\mathrm{djob1}}^{s,g1} = \mathrm{xg}_{\mathrm{djob2}}^{s,g2} = 1 \end{aligned} \tag{3.14}$$

(C9) Satellite real-time transmission scenario constraint. When a satellite uses the real-time transmission mode, the time window for data transmission must completely cover the time window for the earth observation task. This allows the observation data of the task to be transmitted to the ground station immediately during the observation is being performed.

$$\begin{aligned} &\left[t_{\mathrm{b}}^{\mathrm{task}}, t_{\mathrm{e}}^{\mathrm{task}}\right] \subseteq \left[t_{\mathrm{db}}^{\mathrm{djob}}, t_{\mathrm{de}}^{\mathrm{djob}}\right],\ \forall s \in \mathrm{SAT},\ \forall g \in \mathrm{GRD}, \\ &\mathrm{djob} \in \mathrm{DT},\ \exists \mathrm{task} \in \mathrm{TASK},\ \mathrm{iff}\, \mathrm{xg}_{\mathrm{djob}}^{s,g} = 1,\ \mathrm{xgl}_{\mathrm{djob}}^{s,g} = 1 \end{aligned} \tag{3.15}$$

Definition 3.1: Time window conflict of data transmission activity If two data transmission activities that use the same data transmission resource but different satellites have overlapping visible time windows, they cannot be carried out simultaneously. This results in a conflict between the two data transmission activities and violates constraint (violating the constraint (C7)).

Definition 3.2: Conflict of selection of resources for data transmission If a satellite has the capability to transmit observation data to two data transmission resources simultaneously, there can be conflicts in the selection of resources for data

transmission between the two data transmission activities (violating the constraint (C8)).

Definition 3.3: Data transmission conflict window (DTCW) The longest window that contains continuous data transmission conflicts or a single data transmission activity. It is also named conflict window for short.

Definition 3.4: Compatible data transmission chain One or more data transmission activities within the same data transmission conflict window do not conflict with each other (and do not violate Constraint (C7)). These activities form a sequence ordered by the start time of data transmission.

Definition 3.5: Inclusion relation of compatible transmission chains If ξ_1 and ξ_2 are two compatible transmission chains, if data transmission activities in ξ_1 are all included in ξ_2 and the corresponding data transmission activities adopt the same transmission mode, then ξ_2 includes ξ_1, or denoted by $\xi_1 \subseteq \xi_2$.

As is shown in Fig. 3.10, A, B, C, D, and E are data transmission activities of the same data transmission resource, ordered by their start time of data transmission. The horizontal axis is timeline, and the length of rectangle represents the size of visible time window. We can identify that conflicts occur between A and B, B and C, C and D, E doesn't conflict with any transmission window. Therefore, A, B, C, and D constitute a data transmission conflict window, and E forms a conflict window by itself. All the data transmission activities of a resource can be divided into several conflict windows. $A-C$, $A-D$, and $B-D$ are compatible data transmission chains. Besides, A, B, C, D, and E form a compatible data transmission chain by themselves. The inclusion relation of the compatible transmission chains in Fig. 3.10 is $A, C \subseteq A-C$, $A, D \subseteq A-D$, and $B, D \subseteq B-D$.

(1) Data transmission resource scheduling objective function

The priority of a data transmission activity is determined by the priority of the targets included in the observation data being transmitted. If the data includes high-priority targets, the data transmission activity will have a higher priority. With an existing satellite observation programme, changes in the satellite data transmission programme will result in changes to the observation data transmitted to the ground

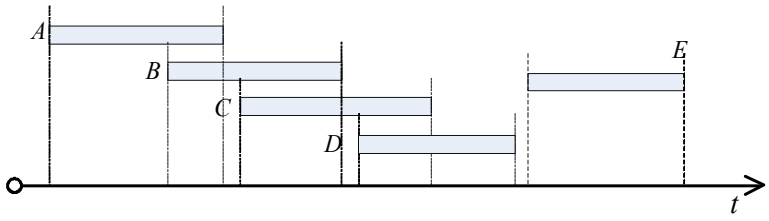


Fig. 3.10 Schematic diagram of data transmission conflict windows

stations, leading to a change in the priority of the data transmission activities. This is referred to as the dynamic priority of a data transmission activity djob $\in$ DT. For a single data transmission activity, its priority is a model variable that must be calculated in the scheduling process. The value of the priority is related to the overall priority of the observation data in the onboard memory of the satellite at the start time of the current data transmission activity.

Considering the characteristics of the satellite data transmission scheduling problem, the optimization objectives are formulated as follows:

(a) **Optimization objective for data transmission importance**

$$V_{\text{imp}}^{\text{djob}} = \max \sum_{\text{task} \in \text{Dd(djob)}} \psi_{\text{task}} \tag{3.16}$$

where Dd(i) is a mapping function that returns the set of meta-tasks whose observation data contained in the data transmission activity i, $i \in$ DT. The optimization objective of data transmission importance indicates that the priority of a data transmission activity is higher if the data to be transmitted is more important.

(b) **Optimization objective for data transmission timeliness**

$$V_{\text{urg}}^{\text{djob}} = \min \sum_{\text{task} \in \text{Dd(djob)}} \left(t_{\text{db}}^{\text{djob}} - t_{\text{e}}^{\text{task}} \right) \cdot \psi_{\text{task}} \tag{3.17}$$

The optimization objective of data transmission timeliness suggests that observation data containing higher priority targets should be transmitted as early as possible.

The priority (gain) of a single data transmission activity djob can be formulated as

$$\text{prty}_{\text{djob}} = \alpha_{\text{imp}}^{\text{dd}} \cdot V_{\text{imp}}^{\text{djob}} - \alpha_{\text{urg}}^{\text{dd}} \cdot V_{\text{urg}}^{\text{djob}} \tag{3.18}$$

In which, $\alpha_{\text{imp}}^{\text{dd}}$, $\alpha_{\text{urg}}^{\text{dd}}$ are the weights, which can be set by the user, or given by experts. It is important to note that the observation data transmitted through the real-time transmission mode and the playback transmission mode to a ground station may differ, resulting in different returns of the function Dd(djob). In the real-time transmission mode, both the observation sensor and the data transmission payload of a satellite are switched on simultaneously to perform observations and transmit the obtained data to the ground. During this process, the data in the onboard memory cannot be transmitted to the ground via the real-time data transmission activity. On the other hand, in the playback transmission mode, the satellite observation sensor remains switched off while the data transmission payload is switched on to transmit the data in the onboard memory to the ground. Therefore, the choice of data transmission mode for a data transmission activity djob depends on how to maximize

the benefits of data transmission resource scheduling, as the $\text{prty}_{\text{djob}}$ would also differ accordingly.

Based on the principle of maximizing the benefits of data transmission activities, the optimal objective function for data transmission resource scheduling can be expressed as follows:

$$V_{\text{imp}}^{\text{dd}} = \max \sum_{\text{djob} \in DT} \text{prty}_{\text{djob}} \cdot \text{xg}_{\text{djob}}^{s,g} \tag{3.19}$$

(2) The data transmission conflict section constraint graph

Taking into account the time window conflicts of data transmission activities and the constraints on the selection of satellite data transmission resources, we can construct the data transmission conflict section constraint graph (CSCG). The CSCG can be formally represented as $\text{CSCG} = \left(V^g, E_1^g, E_2^g\right)$, where V^g is the set of vertexes, representing the data transmission activity. E_1^g, E_2^g is the set of edges, representing the constraint relations. The vertexes connected by edges from E_1^g, form compatible data transmission chain. If two vertexes connected by edges from E_2^g, the corresponding two data transmission activities violate constraint C8 (satellite data transmission resource selection constraint).

The schematic diagram of CSCG is shown in Fig. 3.11.

Figure 3.11 is the CSCG diagram for two data transmission resources (Resource A, Resource B ∈ GRD). Where, A1 ∼ A8, B1 ∼ B9 ∈ DT are the data transmission activities associated with the data transmission resources A and B, respectively. A1 ∼ A8 and B1 ∼ B9 have been sorted according to the start time of the data

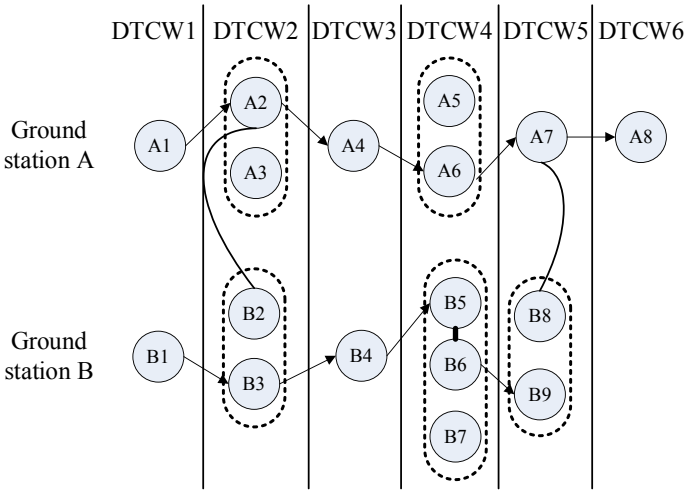


Fig. 3.11 Schematic diagram of the data transmission conflict section constraint graph

transmission and divided into data transmission conflict window (DTCW). There are six DTCWs in total in Fig. 3.11.

The data transmission activities wrapped in dashed boxes represent those in the same DTCW (e.g., A2 and A3). The data transmission activities with edges connected in the dashed boxes form a compatible data transmission chain (e.g., B5 and B6 $\in E_1^g$). The edges connecting the data transmission activities between different data transmission resources represent that the corresponding two data transmission activities violate Constraint C8, which is the satellite data transmission resource selection constraint (e.g., A2 vs. B2; A7 vs. B8 $\in E_2^g$). Based on the CSCG, the purpose of satellite data transmission resource scheduling is to select a chain of data transmission activities from the different DTCWs to form a path from A1 to A8 and B1 to B9 (as shown by the arrows in the middle of Fig. 3.11) that maximizes the value of the data transmission resource scheduling optimization objective (see Eq. 3.19).

Based on the optimization objective of the data transmission resource scheduling and the CSCG model, we can obtain Theorem 3.1.

Theorem 3.1 If there is an inclusion relationship between two compatible data transmission chain, the gain from executing the longer compatible data transmission chain is not less than the gain from executing the shorter one.

Proof Suppose there are two compatible data transmission chains ξ_1, ξ_2, and $\xi_1 \subseteq \xi_2$. If $\xi_1 = \xi_2$, then the above proposition clearly holds. Therefore, it is only necessary to prove the case $\xi_1 \subset \xi_2$. From Definition 3.5 we can see if $\xi_1 \subset \xi_2$, then $\exists$djob $\in$ DT, such that djob $\notin \xi_1$, and djob $\in \xi_2$. The gain of executing the compatible data transmission chain ξ_2 is no less than the gain of executing ξ_1 plus executing the data transmission activity djob, according to the evaluation criteria of the optimization objective for data transmission importance and the optimization objective for data transmission timeliness proposed above.

Since the execution of the data transmission activity djob is greater than or equal to 0, the gain obtained from executing a compatible data transmission chain ξ_2 at least as much as the gain obtained from executing ξ_1 a single data transmission activity. Thus, the original proposition is proven.

Based on Theorem 3.1, the compatible data transmission chains in a DTCW can be refined to reduce the search space of the algorithm, i.e., for all compatible data transmission chains in a DTCW, if there is an inclusion relationship between two compatible data transmission chains, the shorter one can be deleted. The above operation is performed for all DTCWs in the CSCG. The CSCG in the following refers to the refined CSCG.

2. Satellite data transmission resource scheduling algorithm

In the context of the communication scheduling problem within the US Air Force Satellite Control Network (AFSCN) ground station network, Barbulescu et al. have shown that the multiple ground station data transmission scheduling problem, with the objective of maximizing the total serving time while considering constraint (C7), is an NP-hard computational problem [110]. Our problem is further complicated

by the need to consider the dynamic priority characteristics of data transmission activities. To address this, we propose the task-oriented satellite data transmission resources scheduling algorithm (TDRSA) based on the elite archive genetic algorithm for the CSCG model. The framework of TDRSA is similar to that of the previously proposed SOCSA and will not be reiterated here. Instead, we adopt the optimal objective function of data transmission resource scheduling ($V_{\text{imp}}^{\text{dd}}$ shown in Eq. 3.19) as the evolutionary fitness value for TDRSA. This section will focus on the encoding of the problem and the design of genetic operators.

(1) Problem coding of TDRSA

For the satellite data transmission scheduling problem, we use an equal-length integer encoding approach to represent each DTCW of the CSCG model. For each DTCW j in the CSCG corresponds to a decision variable d_j ($d_j \in \mathbb{N}$). If $d_j = \alpha$ represents the compatible data transmission chain α from DTCW j will be performed, i.e., for $\forall \text{djob} \in \alpha$, set $\text{xg}_{\text{djob}}^{s,g} = 1$. In addition, the $\text{xgl}_{\text{djob}}^{s,g} = 1$ indicates that the corresponding data transmission activity will be performed in the real-time transmission mode; otherwise, the playback transmission mode is used.

(2) Crossover operator design of TDRSA

The crossover operator in TDRSA is designed as a single-point crossover approach. This operator randomly selects two parent individuals in a roulette wheel manner based on their fitness value and determines a crossover point to implement the crossover process randomly (as shown in Fig. 3.12). After the completion of the crossover operation, the constraint handling process is performed. The division of DTCWs in the CSCG model and the formation of compatible data transmission chains ensure the satisfaction of constraint (C7). Therefore, we only need to handle constraint (C8) after the crossover operation. To address this constraint, we select edges from the set E_2^g of CSCG models and randomly choose one data transmission activity to execute while canceling any conflicting data transmission activities.

(3) Mutation operator design of TDRSA

The mutation operator in TDRSA adopts a single-point random variation approach. Specifically, the execution state of a compatible data transmission chain within a DTCW is changed using a random neighborhood search (as shown in Fig. 3.13). The individual to be mutated is generated through random selection. In Fig. 3.13, the gray area represents the mutated DTCW, which has a total of 5 alternative compatible

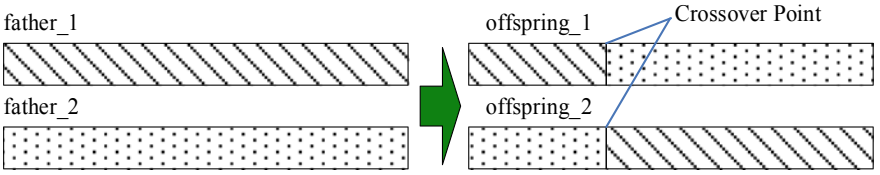


Fig. 3.12 Schematic diagram of the operation of the TDRSA crossover operator

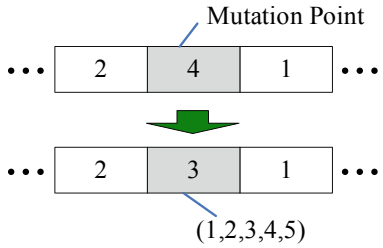


Fig. 3.13 Schematic diagram of the operation of the TDRSA mutation operator

data transmission chains. Before mutation, compatible data transmission chain 4 of the DTCW is executed. However, after mutation, compatible data transmission chain 3 will be executed instead. Additionally, through mutation, the transmission mode (real-time or playback) of the mutated compatible data transmission chain 3 may also be randomly changed.

After mutation, the constraint handling method used in the crossover operation is employed to check for any violations of constraint (C7) and constraint (C8). Additionally, constraint (C9) must also be verified and handled. If the mutation operator changes the transmission mode of a data transmission activity to real-time transmission mode but it cannot be performed in real-time mode, constraint (C9) is violated, and the transmission mode must be adjusted back to the original playback mode.

(4) Select operator design of TDRSA

The TDRSA selection operator is a variant of the roulette wheel selection operator that incorporates an elite archive mechanism similar to that of the SOCSA algorithm. This operator assigns probabilities to each chromosome in the parent population, based on their respective fitness values. Chromosomes with higher fitness values have a greater probability of being selected for the offspring population. Moreover, a few top-performing chromosomes are directly added to the offspring population as elite solutions.

(5) TDRSA algorithm constraint handling

Thanks to the data transmission conflict section constraint graph model, infeasible solutions generated during the iterations of the TDRSA algorithm can be corrected to feasible solutions in the crossover and mutation operators. Thus, the TDRSA algorithm does not require a special constraint handling algorithm.

3.2.3 Progressive Iterative Repair Mechanism

The initial two stages of the multi-stage scheduling problem involve resolving the satellite earth observation task scheduling problem and the satellite data transmission scheduling problem, respectively. These stages are essentially the process of finding the local optimum through multi-stage scheduling. However, it should be noted that

the combination of these optimal solutions does not necessarily result in the global optimal solution for the overall multi-stage scheduling problem.

For the proposed scheduling approach based on a progressive optimization strategy, neither of the first two scheduling stages considers the satellite memory capacity constraint. In the event that the onboard memory exceeds its capacity, the satellite cannot continue with its observation task until it executes a data transmission activity through the playback mode. The satellite memory capacity constraint is considered in this stage and described in detail below.

(C10) Satellite memory capacity constraint. The maximum memory capacity of the onboard memory must not be exceeded at any given time.

$$\sum_{\substack{\text{task}\in\text{TASK},s_{\text{task}}=s\\ t_{\text{e}}^{\text{task}}\le t}}\left(t_{\text{e}}^{\text{task}}-t_{\text{b}}^{\text{task}}\right)-\sum_{\substack{\text{djob}\in\text{DT},s_{\text{djob}}=s\\ t_{\text{de}}^{\text{djob}}\le t}}\left(t_{\text{de}}^{\text{djob}}-t_{\text{db}}^{\text{djob}}\right)\le \text{memy}^{s}\quad \forall t\in[t_{\text{B}},t_{\text{E}}],\ \forall s\in\text{SAT} \tag{3.20}$$

The third stage of the centralized EOS task scheduling method, based on a progressive optimization strategy, involves a scheduling result progressive iterative repair mechanism. This mechanism considers all constraints and optimization objectives related to observation tasks, EOSs and data transmission resources, and addresses the constraint violations that occur in the scheduling results of the first two stages. The overall earth observation programme is generated after this process.

The fundamental reason why it is difficult to obtain the global optimum of the overall earth observation programme under the progressive optimization strategy is that the divide-and-conquer strategy in the three-stage scheduling process. However, it is inevitable to sacrifice a part of optimality in order to give a satisfactory solution to the problem in a limited time. The existence of the satellite memory capacity constraint (C10) is the most direct reason for the degradation of the optimality of the scheduling results. Simply removing those meta-tasks that violate constraint (C10) will definitely lead to the degradation of the optimality of the overall earth observation programme.

The idea of the progressive iterative repair mechanism is to repair the solutions violated the constraint (C10) through a heuristic strategy. It starts by removing lower-priority tasks with longer observation times until the satellite observation programme no longer violates any constraints. Next, it adds tasks with higher priority and shorter observation times until a constraint is violated. This process is repeated several times to produce the final earth observation programme.

3.3 Centralized EOS Task Scheduling Method Based on a Global Optimization

The divide-and-conquer strategy is based on the key idea of progressive optimization, which divides the centralized satellite task scheduling problem into three subproblems. On the one hand, this strategy effectively reduces the complexity of problem-solving and computation burden; thus, it is possible to obtain a user-satisfactory solution within an acceptable computation time. On the other hand, due to the progressive optimization strategy, it is difficult to ensure that the algorithm searches for the optimal solution from a global perspective. For example, during the satellite earth observation task scheduling stage, only observation tasks with high priority are considered for scheduling, without taking into account whether the onboard memory is overflowing or when the data in onboard memory will be transmitted to the ground. Similarly, during the data transmission resource scheduling stage, the existing satellite observation programme cannot be modified even if there is no available data transmission resource to transmit the observation data in onboard memory to the ground. In some extreme cases, it may happen that a target is observed very early but downlinked very late, and the observation data occupies satellite onboard memory for a long time, increasing the risk of memory overflow without contributing to the data transmission timeliness objective.

In recent years, with the significant increase in computational power, the centralized EOS task scheduling method based on a global optimization strategy has received more and more attention. This method considers the observation process and the data transmission process from a global optimization perspective to find a more reasonable problem solution. Although it has rarely been applied to practical engineering, it plays an indispensable role in analyzing the optimization upper bound of satellite task scheduling problems from a theoretical perspective. In the foreseeable future, with the further enhancement of hardware computing power, this method will have good application prospects. In this section, a centralized satellite task scheduling method based on a global optimization strategy is presented.

3.3.1 EOS Scheduling Model Based on Global Optimization Strategy

The CSP model is still utilized to formulate the satellite task scheduling problem under the global optimization strategy. In contrast to the progressive optimization strategy, both the task importance optimization objective in the satellite earth observation task scheduling stage and the task timeliness objective in the data transmission resource scheduling stage are considered simultaneously. This approach aims to find a solution that optimizes both objectives under a global optimization perspective.

Task importance optimization objective is formulated as follows:

$$V_{\text{imp}}^{\text{wh}} = \max \sum_{s \in \text{SAT}} \sum_{\text{task} \in \text{TASK}} x_{\text{task}}^{s} \cdot \psi_{\text{task}} \cdot \text{down}(\text{task}) \tag{3.21}$$

In which, the down(task) function is used to obtain the result whether the observation data of task can be downlinked to the ground within the scheduling time horizon. If it can be, then down(task) = 1; otherwise down(task) = 0.

Compared to Eq. (3.1), Eq. (3.21) requires not only that the satellite performs as many and more important earth observation tasks as possible but also that the observation data of these tasks be transmitted to the ground within the current scheduling time horizon.

Task timeliness optimization objective is formulated as follows:

$$V_{\text{urg}}^{\text{wh}} = \max \sum_{s \in \text{SAT}} \sum_{\text{task} \in \text{TASK}} x_{\text{task}}^{s} \cdot \psi_{\text{task}} \cdot \left(t_{\text{E}} - t_{\text{task}}^{\text{down}}\right) \tag{3.22}$$

In which, $t_{\text{task}}^{\text{down}}$ is the observation data downlink time of task, which can be figured out from the satellite data transmission programme. The timeliness optimization objective requires that the observation data of tasks with higher priority should be transmitted to ground as early as possible.

The optimization objective for satellite task scheduling based on the global optimization strategy can be formulated as the form of a weighted sum of the two indicators mentioned above.

$$V_{\text{wh}} = \alpha_{\text{imp}}^{\text{wh}} \cdot V_{\text{imp}}^{\text{wh}} + \alpha_{\text{urg}}^{\text{wh}} \cdot V_{\text{urg}}^{\text{wh}} \tag{3.23}$$

In which, $\alpha_{\text{imp}}^{\text{wh}}$, $\alpha_{\text{urg}}^{\text{wh}}$ are the weighting coefficients, and the weights can be set by the users, or given by the experts.

In the CSP model for satellite task scheduling based on the global optimization strategy, the scheduling process needs with respect to the constraints C1–C10 (see Sect. 3.2 for details), which are not repeated here.

3.3.2 *Earth Observation Satellites Scheduling Algorithm with Data Downlink*

In this section, we propose a centralized earth Observation Satellites Scheduling Algorithm with Data Downlink (SSADD), based on an elite archive genetic algorithm. The algorithm framework is similar to that of SOCSA, as introduced in Sect. 3.2.1, and will not be reiterated here. However, we will provide a detailed description of the genetic operators and constraint handling process used in SSADD.

(1) Encoding of SSADD

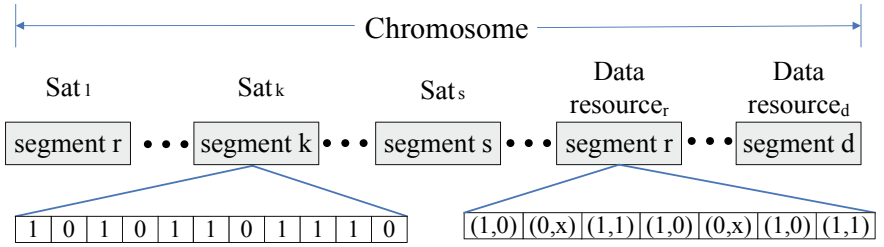


Fig. 3.14 Schematic diagram of SSADD coding

The problem encoding of SSADD is shown in Fig. 3.14. We construct the chromosome using equal-length 0–1 coding to represent the satellite observation programme (first half of the chromosome) and satellite data transmission programme (he second half of the chromosome). Each gene locus of the first half of the chromosome represents the decision variable of an observation task. If $x_k^s = 1$ denotes the observation task k will be performed by the satellite s, otherwise, it will not be performed in this VTM, where $t \in$ TASK, $s \in$ SAT. Similarly, each gene locus of the second half of the chromosome represents the data transmission decision variables $\{\mathrm{xg}_{\mathrm{djob}}^{s,g}, \mathrm{xgl}_{\mathrm{djob}}^{s,g}\}$.

(2) Crossover operator design of SSADD

We have developed a location-constrained multi-point crossover operator, as shown in Fig. 3.15, to perform crossover operations on individuals. This operator selects several crossover points from both the first and second halves of the chromosome to ensure that both observation tasks and data transmission activities are processed by the crossover operator. In Fig. 3.15, the current chromosome consists of three observation task chromosome segments and two data transmission activity chromosome segments. We obtain individual father_1 using roulette selection, where the probability of each chromosome being selected is proportional to its fitness value. The individual father_2 is obtained by random selection.

(3) Mutation operator design of SSADD

We have developed a location-constrained two-points mutation operator for SSADD. Specifically, one mutation point is selected from the first half of the chromosome, while another point is selected from the second half of the chromosome. When it comes to observation task chromosome segments, this operator selects an observation task gene randomly and changes the executing state for that gene. With regard to data transmission activity chromosome segments, the operator not only inverts the executing state for the data transmission activity, but also randomly assigns the data transmission mode (i.e., real-time transmission or playback) for that gene. It is noted

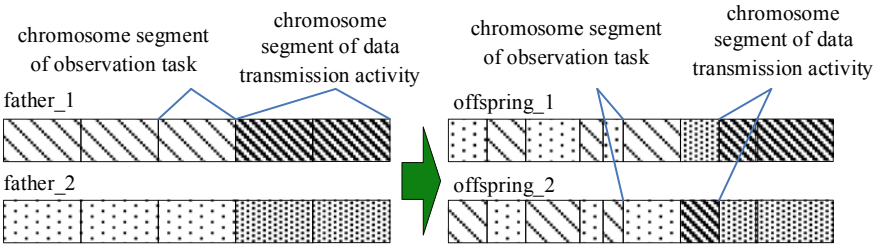


Fig. 3.15 Schematic diagram of the operation of the SSADD crossover operator

that, the mutation operator needs with respect to the satellite real-time transmission scenario constraint (C9).

(4) Select operator design of SSADD

The SSADD selection operator adopts an elite archive mechanism, which is similar to that of the selection operator of SOCSA, which can be found in detail in Sect. 3.2.1.

(5) Constraint handling of SSADD

Similarly to the constraint handling process of SOCSA, constraints C1, C2, C6–C9 can be handled (directly canceling the observation tasks or data transmission activities that violate any constraints) during the crossover and mutation operation of SSADD. In this section, we focus on how to handle constraints (C3), (C4), and (C10).

The SOCSA algorithm utilizes a penalty function-based constraint handling method. However, the solution space of SSADD is considerably larger than that of SOCSA, and the penalty function approach is relatively inefficient and more prone to falling into local optima. As a result, we have designed a constraint handling method based on the solution repair method for SSADD. The key idea of the solution repair method is to design an algorithm to transform the infeasible solutions (violating some constraints) into feasible ones (violating no constraints).

If the solution repair method is well designed and the computing cost is low, the solution repair method usually achieves better results compared to the penalty function method [129]. For constraints (C3), (C4), and (C10), a solution repair method based on the stochastic greedy strategy is designed as follows:

Algorithm name: SolutionRepair4SSADD
Input: chromosome before solution repair chrom
Output: chromosome after solution repair chrom

(continued)

(continued)

begin
1 while CheckCircleConstraint(chrom_pre) == false
2 $\text{Task\_Set}^s_{\text{circle}} = \text{FindCircleTask(chrom\_pre)}$
3 Random select a task k from $\text{Task\_Set}^s_{\text{circle}}$ // *tasks with low priority and late downlink time have a higher probability of being selected*
4 set $x^s_k = 0$ // *Cancel the execution of this task*
5 end while // *Handle the constraint* (C3)
6 while CheckDayConstraint(chrom_pre) == false
7 $\text{Task\_Set}^s_{\text{day\_span}} = \text{FindDaySpanTask(chrom\_pre)}$
8 Random select a task k from $\text{Task\_Set}^s_{\text{day\_span}}$ // *tasks with low priority and late downlink time have a higher probability of being selected*
9 set $x^s_k = 0$ // *Cancel the execution of this task*
10 end while // *Handle the constraint* (C4)
11 while CheckMOverflowConstraint(chrom) == false
12 $\text{Task\_Set}^s_{\text{overflow}} = \text{FindOverflowTask(chrom\_pre)}$
13 Random select a task k from $\text{Task\_Set}^s_{\text{overflow}}$ // *tasks with low priority and late downlink time have a higher probability of being selected*
14 set $x^s_k = 0$ // *Cancel the execution of this task*
15 end while // *Handle the constraint* (C10)
end

In the solution repair algorithm, the chrom_pre is the first half of the chromosome chrom, which represents the execution state of observation tasks.

Statement 1–5 handles constraint C3 (sensor maximum cumulative working time constraint in a single circle), the function CheckCircleConstraint() detects whether the chromosome violates constraint C3. If constraint C3 is violated, the function returns false. FindCircleTask() returns an observation task set involving all tasks violates constraint C3 within the earliest circle. The algorithm randomly selects one task from that circle (the strategy for selecting a task is that tasks with low priority and late downlink times have a higher probability of being selected) and cancels it until constraint C3 is no longer violated.

Statement 6–10 handles constraint C4 (sensor maximum cumulative working time constraint in a single day), the process of dealing with constraint C4 is similar to constraint C3. The function CheckDayConstraint() detects whether the chromosome violates constraint C4. If constraint C4 is violated, the function returns false. FindDaySpanTask() returns an observation task set involving all tasks violates constraint C4 within the first 24 h time span.

Statements 11–15 are responsible for handling constraint C10 (Satellite memory capacity constraint). The function CheckMOverflowConstraint() checks whether the onboard memory will overflow (returning false if constraint C10 is violated). If so, the FindOverflowTask() function returns the set of tasks that caused the overflow of onboard memory, and the subsequent processing is similar to the handling of constraints C3 and C4.

Apparently, the time complexity of the algorithm SolutionRepair4SSADD is $O\left(n^2\right)$. Therefore, it is a polynomial time algorithm.

3.4 EOS Scheduling for Complicated Observation Task

The scheduling approaches introduced above are for targets that can be regarded as static points on the ground. In practice, there are two types of complicated observation targets, namely area targets and moving targets. In this section, we will focus on the satellite observation task scheduling methods for the two special types of targets.

3.4.1 EOS Task Scheduling for Area Target

Observation targets can be divided into two categories based on the relative size relationship between the field of view of the onboard sensor and the ground target area: point targets and area targets. Point targets are small relative to the field of view of the onboard sensor and can be completed through a single observation by one satellite. Examples of point targets include airports, ports, and other facilities. Scheduling methods for point targets have been introduced in Sects. 3.2–3.3. In contrast to the point target satellite observation scheduling problem, the area target satellite observation scheduling problem has the following characteristics.

(1) Need to consider the geometric properties of an area target

Since the area target is generally larger than the field of view of onboard sensor, it cannot be regarded as a point on ground. Thus, the geometrical characteristics of the area target and the field of view of onboard sensor must be considered (the geometrical characteristics of the coverage region of different types of sensors are generally different).

(2) Need to consider the decomposability of observation tasks for an area target

Since an area target cannot be covered by a single observation, it needs to be divided into several subtargets. Each subtarget is an observation strips, which can be performed by one observation via onboard sensors. Considering the characteristics of satellite observation process, the direction of observation strips division is parallel to the satellite orbit direction.

(3) Need to consider the completion degree of an area target

For point targets, there are only two possible states: satisfied (i.e., observed) or unsatisfied (i.e., unobserved). However, for area targets, partial satisfaction can be achieved if only a portion of the corresponding subtargets is observed. If an area target is completely covered, there should be an additional benefit beyond the sum of the benefits of its corresponding subtargets. Moreover, the relationship between the completion degree of an area target and the benefit is nonlinear. The benefit of completing remaining subtargets on this basis increases with the degree of completion. Thus, the evaluation method for the completion degree of an area target (coverage rate of the area target) needs to be designed.

The typical processing approach for area targets generally involves a two-stage strategy of "decomposition first, scheduling second." In other words, the area target observation scheduling problem is decomposed into two subproblems: area target decomposition and satellite observation task scheduling for subarea targets. During the first stage, the area target is divided into multiple subtargets or strips that can be fully covered by onboard sensors via a single observation. The parameters, such as switch-on and switch-off time and satellite roll angle, of the onboard sensor are determined for each time window of satellite access to the subarea targets. The second stage involves scheduling the observation tasks for subarea targets (strips) which is very similar to conventional observation task scheduling algorithms for point targets. And for area targets, we need to consider the evaluation method for completion degree and benefit of observations to area targets.

1. Area target decomposition algorithm

Area target decomposition involves decomposing the area target observation tasks based on satellite visible windows and orbits and constructing a set of candidate observation strips for each time window. Given that area target decomposition is closely related to satellite parameters such as orbit and field of view, this section proposes a time-stamped observation strip model to capture the coverage characteristics of satellite onboard sensors. Additionally, the section introduces the constraint satisfaction polygon cutting algorithm (CSPCA) for area target decomposition.

(1) **Time-stamped observation strip model**

The observation strip is a subtarget divided from an area target, and it can be covered by an onboard sensor within a single observation.

Definition 3.6: Time-stamped coverage boundary The time-stamped coverage boundary describes the boundary of coverage area when an onboard sensor takes a photo to ground. It contains the location, coverage area, and access time of a satellite. Therefore, a time-stamped coverage boundary usually corresponds to a sequence of time-stamped latitude and longitude coordinate points, denoted as $\mathrm{TBoundary} = (\mathrm{TPoint}_1, \ldots, \mathrm{TPoint}_M)$. In which, TPoint denotes a time-stamped coverage boundary point, defined as (Latitude, Longitude, TimeLabel), denoting the position and satellite access time of the boundary point, respectively.

Definition 3.7: Time-stamped observation strip (TStrip) Time-stamped observation strip is a rectangular area covered by onboard sensor via a single observation. A time-stamped observation strip TStrip can be represented by a seven-tuple, $\mathrm{TStrip} \equiv \langle \mathrm{strip\_ID}, \mathrm{AreaTar}, \mathrm{sat}, t_{\mathrm{b}}^{\mathrm{task}}, t_{\mathrm{e}}^{\mathrm{task}}, \mathrm{angle}, \mathrm{TBoundary} \rangle$, where strip_ID denotes the identifier of the observation strip, and AreaTar is the identifier of the area target corresponding to the observation strip. sat is the satellite performing the observation. $t_{\mathrm{b}}^{\mathrm{task}}$ and $t_{\mathrm{e}}^{\mathrm{task}}$ are the switch-on and switch-off time of e corresponding onboard sensors respectively. angle is the observation angle used by the satellite to perform the corresponding observation activity. TBoundary is the time-stamped coverage boundary of the observation strip.

For a TStrip, the following basic operations can be performed on it.

(1) Set a value to $t_{\mathrm{b}}^{\mathrm{task}}$ and $t_{\mathrm{e}}^{\mathrm{task}}$.
(2) Update the value of the TBoundary (the attributes of the corresponding TPoint).
(3) Covering relationship analysis: both the observation strip and the area target can be regarded as Polygon. We can do spatial intersection operation between them to determine whether the two polygons overlap. If the result is empty (not overlap at all), return 0; else, return 1.

The length of the observation strip depends on the difference between $t_{\mathrm{b}}^{\mathrm{task}}$ and $t_{\mathrm{e}}^{\mathrm{task}}$, and the above operations (1) and (2) enable the correction of the observation strip length to make it reasonable (not too large or small) to alleviate the rick of violating satellite constraints.

For example, when the length of observation strip is very long/short (the area target range is very large/small), a satellite may violate constraint C1 (satellite single power-on time constraint).

Operation (3) is mainly used to calculate the coverage of the observation strip with respect to the area to be observed, which can be used as an important factor for deciding whether the observation strip is performed or not during task scheduling.

After giving the definition of TStrip, the constraint-satisfiable area target decomposition algorithm will be described as follows.

(2) **Constraint-satisfiable area target decomposition algorithm**

To decompose an area target into candidate observation strips (which can be viewed as an earth observation meta-task), this section utilizes the observation angle discretization strategy based on the access time window of a single satellite to the area target. It constructs observation strips with different observation angles within the maximum observation coverage of the satellite.

Firstly, the notations are defined as follows:

(1) Area target sets $\mathrm{ATARGET} \equiv \left\{\mathrm{AreaTar}_1, \mathrm{AreaTar}_2, \ldots, \mathrm{AreaTar}_{\mathrm{Np}}\right\}$, Np is the total number of area targets.
(2) The satellite imaging roll angle is denoted as β, and range of the imaging roll angle is $(\beta_{\min}, \beta_{\max})$. If the satellite does not have rolling capability, then $\beta = 0$.
(3) The minimal stride of imaging roll angle of Satellite sat_i. is denoted as $\Delta\beta_i$.
(4) The visible window set of satellite sat_i to $\mathrm{AreaTar}_j$ can be denoted as $T_{ij} = \left\{t_1, t_2, \ldots, t_{N_{ij}}\right\}$, where $t_1, t_2, \ldots, t_{N_{ij}}$ is the satellite visible windows and N_{ij} is the total number of satellite visible windows.
(5) For kth satellite visible window between satellite sat_i and area target $\mathrm{AreaTar}_j$, can be denoted as $t_{N_{ijk}}$, where $k \in \left[1, N_{ij}\right]$. There are several candidate observation strips in $t_{N_{ijk}}$ with different satellite imaging roll angle. The set of candidate observation strips in $t_{N_{ijk}}$ can be denoted as $I_{ijk} = \left\{\mathrm{strip}_{ijk1}, \mathrm{strip}_{ijk2}, \ldots, \mathrm{strip}_{ijkN_{ijk}}\right\}$, which is the decomposition result of $\mathrm{AreaTar}_j$ in $t_{N_{ijk}}$. strip_{ijkv} means the vth candidate observation strips in $t_{N_{ijk}}$. The set of candidate observation strips of the area target $\mathrm{AreaTar}_j$ can be denoted as

$I_j = \{I_{1j}, I_{2j}, \ldots, I_{N_S j}\}$, where N_S is the number of satellites involved in the scheduling.

The constraint-satisfiable area target decomposition algorithm (CSPCA) is designed to obtain the candidate observation strips. CSPCA will generate the set of candidate observation strips for the pairs between each satellite and each area target. Taking the satellite sat_i for the area target AreaTar_j as example. Firstly, CSPCA will figure out all the satellite visible windows between satellite sat_i and area target AreaTar_j. Then, for each satellite visible window, the candidate satellite observation strips are generated with different imaging roll angle (increasing from $\beta_{\min}$ to $\beta_{\max}$ in a certain stride $\Delta\beta_i$) and the spatial coverage relationship between the strips and the area target. Lastly, the candidate strips of all satellite visible windows are combined to form the set of observation strips between satellite sat_i and area target AreaTar_j. Similarly, the observation strips of all satellites for all area targets can be calculated.

The pseudo-code of CSPCA is as follows:

Algorithm name: CSPCA
Input: Satellite set SAT, Area target set ATARGET, Scheduling time horizon $[t_B, t_E]$
Output: The set of candidate observation strips IS

begin
1 **for** $j \leftarrow 1$ to Np
2 **for** $i \leftarrow 1$ to Ns
3 $T_{ij} \leftarrow \text{ComputeTimeWindow}(\text{sat}_i, \text{AreaTar}_j, t_B, t_E)$ *// figure out satellite visible windows*
4 **for** $k \leftarrow 1$ to $N_{T_{ij}}$
5 $\beta \leftarrow \beta_{\min}$;
6 **while** $\beta < \beta_{\max}$
7 $\beta \leftarrow \beta + \Delta\beta$
8 $\text{strip}_{ijkv} \cdot \text{TimeLabelCoords} \leftarrow \text{ComputeStripCoordinate}(\text{sat}_i, \text{AreaTar}_j, \beta)$ *// figure out the boundary points of the strip.*
9 $\text{strip}_{ijkv} \cdot t_b^{\text{task}}, \text{strip}_{ijkv} \cdot t_e^{\text{task}} \leftarrow \text{ComputeAccess}(\text{sat}_i, \text{strip}_{ijkv} \cdot \text{Coords})$ *// figure out the observation start time and end time of the strip.*
10 $I_{ijk} \leftarrow I_{ijk} \cup \{\text{strip}_{ijkv}\}$
11 **end while**
12 $I_{ij} \leftarrow I_{ij} \cup \{I_{ijk}\}$
13 **end for**
14 $I_j \leftarrow I_j \cup \{I_{ij}\}$
15 **end for**
16 IS $\leftarrow \bigcup_{j=1}^{\text{Np}} I_j$
end

2. Satellite observation task scheduling for subarea targets (strips)

After the constraint-satisfiable area target decomposition, the set of candidate observation strip is generated. For each candidate observation strip, it can be completely

covered via a single observation by satellite. Thus, the observation strips can be regarded as point targets.

The area target observation strip scheduling problem is similar to the point target observation scheduling problem and is formulated using a constraint satisfaction problem (CSP) model. However, the area target observation strip scheduling problem has unique characteristics that require a proper objective function. Unlike point targets, area targets are considered completed only when they are fully covered (100%). In practice, observation tasks for area targets are often only partially satisfied, meaning that only a subset of the observation strips are performed. Additionally, observation strips that belong to the same area target have a relationship with each other, and the relationship between the completion degree of an area target and its associated benefit is nonlinear. Therefore, a proper objective function must be designed for the CSP model to accurately capture the characteristics of the area target observation strip scheduling problem. The higher the degree of completion, the higher the benefit of completing remain subtargets (strips) on this basis.

This section proposes an optimization objective function for the area target observation strip scheduling problem. The objective function is based on partial reward of area targets, considering the criteria of task importance. Then the scheduling method for the area target observation strips is introduced.

(1) Partial reward of an area target

For every satellite visible window, satellite can only select one candidate observation strip of the visible window to perform, which is only a part of the whole area target. Since the EOS resource is limited, the area target observation tasks may only be partially satisfied during the scheduling time horizon. For example, for 20 area targets observation scheduling scenario, the following are two possible scheduling results.

Result 1: all 20 area target observation tasks achieved 50% imaging coverage within the limited scheduling time horizon.

Result 2: among the 20 area target observation tasks, 10 area targets had an 80% completion rate and the other 10 area targets had only a 20% completion rate for the limited scheduling time horizon.

Typically, result 2 is considered as a better result, as a half of these area targets obtain a higher imaging coverage (close to 100%). The benefit is quite low, if the imaging coverage for an area target is lower than 50%, since we cannot get the useful general information of the area target.

To better describe the observed benefit of an area target, this section assumes that the benefit within the area target is positive correlation to the area of the region covered by the observed strip and defines the partial reward of an area target $\mathrm{AreaTar}_j$ as follows:

$$\mathrm{PartialReward}\left(\mathrm{AreaTar}_j\right) = P\left(\mathrm{Cov}_{\mathrm{AreaTar}_j}, \varepsilon, \delta\right) \tag{3.24}$$

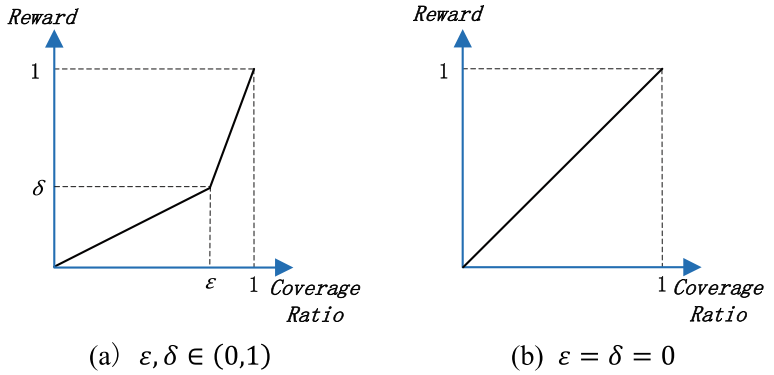


Fig. 3.16 Relationship between imaging coverage rate and observation benefit of an area target

In which, $\mathrm{Cov}_{\mathrm{AreaTar}_j}$ denotes the observed coverage ratio of the area target $\mathrm{AreaTar}_j$, and it is numerically equal to the area of the union of all performed observation strips divided by the area of $\mathrm{AreaTar}_j$. $P\left(\mathrm{Cov}_{\mathrm{AreaTar}_j}, \varepsilon, \delta\right)$ is the reward factor, and it can be formulated as follows:

$$P\left(\mathrm{Cov}_{\mathrm{AreaTar}_j}, \varepsilon, \delta\right) = \begin{cases} \frac{\delta}{\varepsilon}\mathrm{Cov}_{\mathrm{AreaTar}_j}, & \varepsilon, \delta \in (0,1), \mathrm{Cov}_{\mathrm{AreaTar}_j} \le \varepsilon \\ \frac{1-\delta}{1-\varepsilon}\left(\mathrm{Cov}_{\mathrm{AreaTar}_j} - \varepsilon\right) + \delta & \varepsilon, \delta \in (0,1), \mathrm{Cov}_{\mathrm{AreaTar}_j} > \varepsilon \\ \mathrm{Cov}_{\mathrm{AreaTar}_j}, & \varepsilon = \delta = 0 \end{cases} \tag{3.25}$$

In which, ε and δ are parameters of reward factor, both of them $\in [0, 1)$. From Eq. (3.25), we can see that when the observed coverage ratio of the area target is above the threshold ε, the observation benefit of the area target is the $\frac{\delta}{\varepsilon} - 1$ times to the original one. If we set $\varepsilon = \delta = 0$, the observation benefit of the area target is proportional to the coverage rate $\mathrm{Cov}_{\mathrm{AreaTar}_j}$. Figure 3.16 illustrates the observation benefit of the area target function with different ε and δ.

(2) Objective function for area target observation strip scheduling

For the area target observation strip scheduling problem, we still use the task importance criteria as the objective function for optimization. For area targets, the degree of target imaging coverage also needs to be taken into account. The relationship between imaging coverage rate and observation benefit of an area target has been formulated by PartialReward(). Thus, the task importance optimization objective is defined as follows:

$$V_{\mathrm{imp}} = \max \sum_{j \in \mathrm{TARGET}} \psi_{\mathrm{AreaTar}_j} \cdot \mathrm{PartialReward}\left(\mathrm{Cov}_{\mathrm{AreaTar}_j}\right) \tag{3.26}$$

In which, the ψ_{AreaTar_j} is the priority of the area target AreaTar_j, that denotes the total reward when AreaTar_j is covered by 100%.

(3) Methodology for area target observation strip scheduling

Using the CSPCA area target decomposition algorithm, area targets can be decomposed into multiple observation strips, each of which can be treated as an earth observation meta-task. Consequently, the area target observation task scheduling problem is transformed into an observation task scheduling problem with multiple associated point targets, i.e., the observation strips. We can adopt an algorithmic framework and operators similar to the SOCSA introduced in Sect. 3.2.1, and the optimization objective is employed as Eq. (3.26).

3.4.2 EOS Task Scheduling for Ocean Moving Target

The previous chapter focused on the scheduling of satellite observation tasks for fixed targets. However, scheduling observations for moving targets are also of great interest. The observation scheduling problem for moving targets is more complex than that for stationary targets due to several factors. The challenges of moving targets observation scheduling include the complexity of the geographical environment of the moving target, the uncertainty of the target's trajectory, and the high level of timeliness required for the observation task. In this section, satellite observation task scheduling for ocean moving target is introduced.

The moving targets in this section refer to the moving targets on the ocean surface, i.e., large ships or fleets. The exact destination and route of the moving target are unknown. Therefore, the difficulty of the EOS task scheduling problem for ocean moving target lies in how to make accurate prediction of the current position of a moving target.

Satellite observation task scheduling for moving targets typically involves two stages: the search for discovery and the relay observation. During the search for discovery stage, the exact position and velocity of the moving target are unknown. Instead, we have to rely on the EOSs to observe the most likely area to find the target. This stage requires guessing the target's position within a certain range of the ocean area. Once the moving target is discovered by a satellite, the relay observation stage begins. During this stage, the satellite that first observed the target relays the target's exact position and velocity to other satellites, and we need to infer the position of the moving target when the next satellite passes the concerned ocean area, enabling them to track and observe the target. Therefore, the search for discovery and relay observation stages require careful planning and scheduling to ensure efficient use of satellite resources while satisfying observation task requirements.

Theoretically, both in search for discovery stage and relay observation stage, we have to infer the exact position of the target, which can be modeled as a state estimation problem. In this section, a typical algorithm based on target distribution

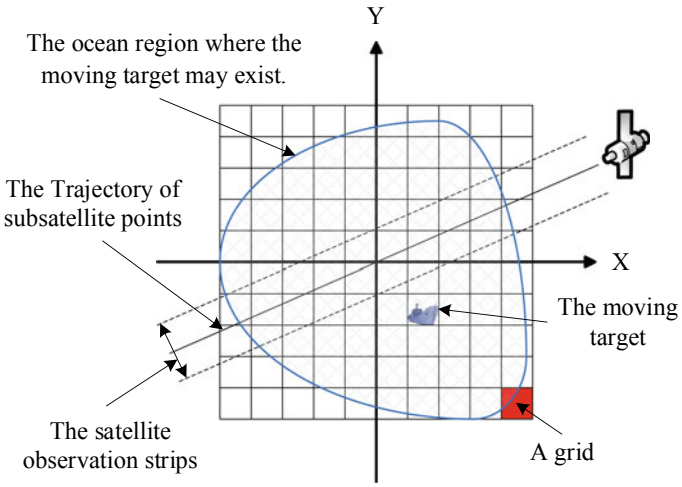


Fig. 3.17 Schematic diagram of regular grid division for the concerned ocean area

updating in grid for the EOS task scheduling for ocean moving target problem is introduced.

1. Regular grid division for the concerned ocean area

In order to infer the position of a moving target using the state estimate theory, it is necessary to discretize the ocean region where the moving target may exist. Thus, we divide the ocean region into regular grids, as shown in Fig. 3.17. After the grid division, we need to calculate the probability that the moving target exists in each grid and make satellite observation strips cover the grids with the highest probability.

The granularity of the grid division has a significant impact on the estimation computation. When the division granularity is too large, the localization accuracy of the moving target is compromised. On the other hand, if the granularity is too small, it can significantly increase the computational burden, leading to excessively long computation times. To address this challenge, the grid length and width are typically set to one-half of the minimum width of the satellite observation strips, taking into account the specific characteristics of the satellite observation process.

2. Calculation of target transfer probabilities based on stochastic maneuver models

Previous studies generally assume that the target moves in a straight line, and thus the target distribution probability follows Gaussian distribution. In fact, when the target moves in a certain ocean area, it may move in a straight line, or it may carry out circular motion, polyline motion, etc. Using one of the above motion models can not accurately describe the motion characteristics of a moving target.

For these reasons, we assume that the moving target follows a stochastic maneuver model, i.e., that the target has the same probability of moving to any position in any direction within its range of motion. Assuming the target appears within the grid i

at the timestamp t_{n-1}, and the average velocity of the moving target is v, then for any timestamp t_n the probability of the moving target lines in a circle (the center is the center point of grid i and the radius is $d = v \cdot \Delta t$) is 1. Within the circle, the distribution probability of the moving target appearing at any grid is equal.

According to the assumptions of the stochastic maneuver model described above, the transfer probability from grid i to the grid j (the probability of target moving from the grid i to the grid j) at nth observation can be modeled based on maximum likelihood probability estimation, i.e.,

$$q(i, j|n) = \max\left(0, \frac{v \cdot \Delta t - d(i, j)}{|v \cdot \Delta t - d(i, j)|}\right) \cdot \frac{p(i|n-1)}{|N_i|} \tag{3.27}$$

where $d(i, j)$ denotes the Euclidean distance from the center of the grid i to that of the grid j. And $\max\left(0, \frac{v \cdot \Delta t - d(i,j)}{|v \cdot \Delta t - d(i,j)|}\right)$ denotes if $d(i, j)$ is more than the maximum distance the target can moved within Δt ($v \cdot \Delta t$), the probability of the moving target exists in grid j is 0. N_i indicates the set of the grid that covered by a circle with grid i as the center and $v \cdot \Delta t$ as the radius. Thus, $|N_i|$ is the number of grids contained in set N_i. $p(i|n-1)$ is the probability of the moving target existing in grid i, at the $(n-1)$th observation.

3. Observation strips generation algorithm for moving target based on transfer probability updating

On the basis of grid dividing and moving target transfer probability, the observation strips generation algorithm for moving target is introduced in this section. The flowchart of the algorithm is shown in Fig. 3.18.

From the main steps of the algorithm, as shown in Fig. 3.18, the probability of the moving target in each grid is estimated based on the transfer probability calculation and the historical observation information during each satellite's visible window over the concerned ocean area. The probability distribution of the moving target in the grid is dynamically updated and guides the subsequent observation actions. If the algorithm runs for the first time, it enters the leftmost branch, and the probability of each grid is initialized. Otherwise, it goes into either the search for discovery stage (middle branch) or the relay observation stage (right branch) based on whether the moving target was discovered in the last observation. In both stages, the posterior probability of the moving target's existence in each grid (i.e., the target distribution probability) is updated based on the results of the previous satellite observation action.

(1) Grid probability initialization

When the algorithm runs for the first time, there is no historical observation information available for the moving target. In such a situation, we can assume that the moving target may be located at any grid in the concerned ocean area, and therefore, the prior probability of the moving target existing in any grid is equal to each other. The sum of the prior probability of all grids in the concerned ocean area is equal to 1. The target distribution probability under each grid can be calculated as follows:

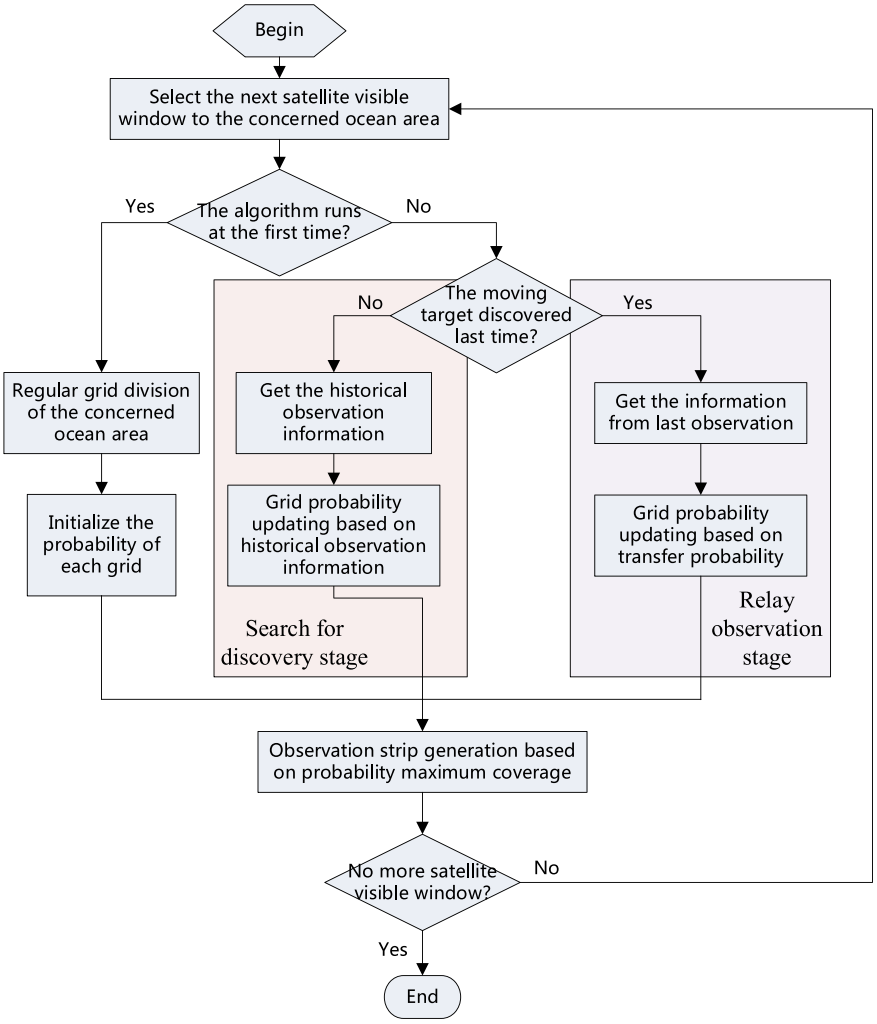


Fig. 3.18 Flowchart of the observation strips generation algorithm for moving target

$$p(i|0) = \frac{1}{|N_{\mathrm{G}}|} \tag{3.28}$$

where N_{G} denotes the sum of all grids in the concerned ocean area.

(2) Posterior probability updating for each grid in search for discovery stage

If a target is not found in the last observation, there are two different cases. Case 1: the target has not been discovered from historical observations, case 2: the target has been discovered from historical observations, and was not found in the last observation.

In case 1, the posterior probability of the grids covered by the last observation strips is zero, since the moving target is not discovered by last observation. The posterior probability of the grids can be calculated by the following equation.

$$p(j|n) = \sum_{i \in N_G} q(i, j|n) \tag{3.29}$$

where the $p(j|n)$ is the posterior probability of the grid j at nth observation, and the $q(i, j|n)$ is the transfer probability from grid i to the grid j at nth observation which can be calculated by Eq. (3.27).

In case 2, we assume that last time, the target was discovered in the historical observation (the observation time is t_0), the corresponding location is grid of i_0, and the velocity of the moving target is v. In the following observations (the observation time is $t_1, t_2, \ldots t_{n-1}$, respectively), the target has never been discovered. Now we need to estimate the location of the moving target at the next observation t_n, which can be expressed as the probability of the moving target existing in each grid. At t_0, since the target was discovered, the probability of the target existing in grid i_0 is one, and the probability of the other grids is zero at timestamp t_0.

Therefore, from t_0 to t_1, the probability of the target existing in grid j can be calculated as

$$p(j|1) = \max\left[0, \frac{v \cdot (t_1 - t_0) - d(i_0, j)}{|v \cdot (t_1 - t_0) - d(i_0, j)|}\right] \cdot \frac{1}{|N_{i_0}|} \tag{3.30}$$

where, $d(i_0, j)$ denotes the Euclidean distance from the center of the grid i_0 to that of the grid j. N_{i_0} indicates the set of the grid that covered by a circle with grid i_0 as the center and $v \cdot (t_1 - t_0)$ as the radius.

For the following observations $t_2, \ldots t_{n-1}$, the posterior probability of the grids can be calculated by Eq. (3.29).

(3) Posterior probability updating for each grid in relay observation stage

The target was found in the grid i_{n-1} last observation at timestamp t_{n-1}, and thus the posterior probability of the moving target existing in grid i_{n-1} is one, and the probability of the other grids is zero at timestamp t_{n-1}. In the nth observation (at timestamp t_n), the target may move to another grid during the span $(t_n - t_{n-1})$.

We assume that the probability of the target moving in all directions is equal, and the target moves at a uniform speed. Therefore, the probability of the target existing in grid j at timestamp t_n can be calculated as follows:

$$p(j|n) = \max\left[0, \frac{v \cdot (t_n - t_{n-1}) - d(i_{n-1}, j)}{|v \cdot (t_n - t_{n-1}) - d(i_{n-1}, j)|}\right] \cdot \frac{1}{|N_{i_{n-1}}|} \tag{3.31}$$

where, $d(i_{n-1}, j)$ denotes the Euclidean distance from the center of the grid i_{n-1} to that of the grid j. v is the velocity of the moving target. $N_{i_{n-1}}$ indicates the set of the grid that covered by a circle with grid i_0 as the center and $v \cdot (t_1 - t_0)$ as the radius.

(4) Satellite observation strip generation based on maximum coverage probability

Based on the probability distribution of the grids in which the moving target is likely to exist, the scheduling algorithm generates observation strips for each satellite, ensuring that the strips cover the grids with the highest probability of containing the target. In the case of optical satellites, generating the observation strip requires determining the switch-on time and roll angle of the satellite. In contrast, for SAR satellites, the scheduling algorithm must simultaneously determine the switch-on time and working mode. These parameters, including switch-on time, roll angle, and working mode, are crucial observation parameters in the satellite observation scheduling algorithm. If the observation parameters of a satellite are fixed, the ground observation strip is determined. And we can figure out the score of the ground observation strip at nth observation as follows:

$$\text{Score} = \sum_{j \in N_{\text{oc}}} p(j|n) \tag{3.32}$$

where N_{oc} is the grid set covered by the ground observation strip, and $p(j|n)$ is the probability of the moving target existing in grid j at nth observation. We only need to traverse all the satellite parameters and find the strip with the highest score.

(5) Moving target observation task scheduling calculation methodology

The generation of observation strips for moving targets can be considered as earth observation meta-tasks, which can be scheduled using meta-task scheduling algorithms. However, the generation of subsequent observation strips depends on the outcome of the previous observation, which may result in conflicts with the existing meta-tasks in the satellite earth observation programme. Therefore, subsequent meta-tasks for moving targets need to be generated dynamically and added to the existing programme based on the results of previous observations. The satellite earth observation programme needs to be rescheduled to solve the conflicts, which will be introduced in Chap. 4.

3.5 Learnable EOS Task Scheduling

With the rapid development of machine learning technology in recent years, there is increasing interest in combining machine learning methods with satellite task scheduling algorithms to further improve performance. In this section, we will introduce satellite task scheduling based on historical earth observation programme case-based learning as an example of learnable EOS task scheduling. EOSs in fixed orbits fly over the same ground region in a certain period with the same trajectory, and the

revisiting interval is called the satellite orbit period. Since the position of most ground targets does not change over a long period of time, the earth observation programme of a satellite exhibits periodically similar characteristics. The latent heuristic information contained in similar historical satellite observation programme can be used to guide scheduling for current observation tasks.

The centralized satellite observation task scheduling algorithm, SOCSA, proposed in Sect. 3.2.1, does not take into account the historical satellite observation programme. If the heuristic information contained in similar historical satellite observation programme could be effectively used to guide the scheduling algorithm, it could potentially improve the optimization of the solution and reduce computational time. Case-based learning (CBL) is an approach that utilizes knowledge from historical cases to address new problems. CBL is suitable for fields where regular knowledge is difficult to find and causality is hard to express using an exact model, and it is capable of adapting to knowledge inconsistency problems [130]. In recent years, CBL has been widely applied in various fields, including medical diagnosis, machining design, circuit design, software engineering, and so on. Methods based on CBL have demonstrated better performance than traditional methods, and the advantages of CBL include accessibility and ease of application compared to complex structured descriptions of knowledge, such as precise mathematical models or rules. However, the process of CBL is usually designed for specific application filed so that it cannot be used directly in different areas.

Introducing the CBL idea to the satellite observation task scheduling problem, heuristic information can be extracted from the results of previous scheduling results, and applied to the solution of current satellite observation task scheduling problem, thus further improving the performance of the scheduling algorithm, and reducing the computational time. The EOSs task scheduling with CBL can be divided into four stages: case feature representation and extraction, case retrieval and matching, case revision, and case application [107] as Fig. 3.19 shown.

Each of the components of the CBL satellite observation task scheduling framework is described respectively as follows.

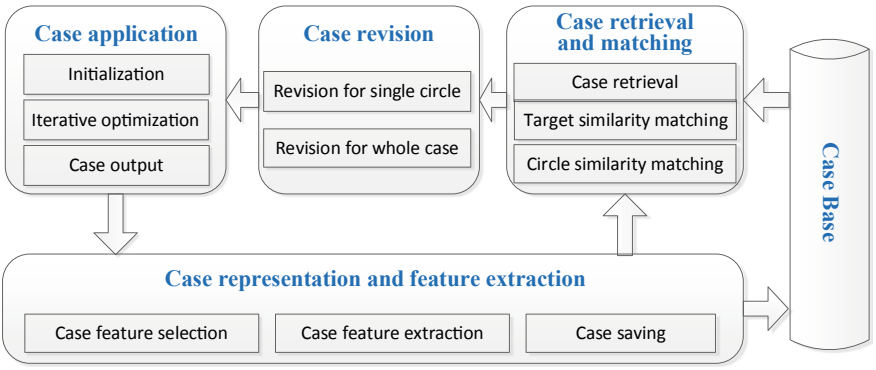


Fig. 3.19 Case-based learning framework for satellite observation task scheduling

3.5.1 Case Representation and Feature Extraction

Case representation is a critical step in case-based learning, which involves representing new problems and cases in the case base using a structured approach. The choice of case representation method is crucial since it must be able to capture all the relevant information in the cases and facilitate efficient case indexing. In our approach, we represent satellite observation tasks using various features such as the observation target set, meta-task set, target location, target access time window, target observation demand, task priority, and sensor requirement. We also include the observation task scheduling results as features of a case. Based on these selected features, we construct the basis structure of a case that serves as the representation of the case. And then historical scheduling results can be stored into case base.

3.5.2 Case Retrieval and Matching

Based on case representation, we need to retrieve the historical cases which can match the current scheduling scenario to be solved. We retrieve historical cases during the periods near-by previous intervals of the satellite orbit period. Furthermore, match the scheduling scenario with the retrieved cases according to the geographical information of targets. The process is as follows:

1. Satellite task scheduling case retrieval

The track of subsatellite point will return to the original track after a satellite orbit period. Therefore, a satellite can observe the same targets with the same observation parameters after several satellite orbit periods.

According to this characteristic, we take the satellite orbit period as the cycle period and find the historical cases whose scheduling horizon is before that of current scheduling scenario as candidates, as shown in Fig. 3.20. In the figure, the $T_$Sat denotes the satellite orbit period. It is worth noting that the alternative cases retrieved by the satellite orbit period may not necessarily match the current scheduling scenario exactly, and further case matching is required because the meta-tasks may be different in each scenario.

2. Satellite task scheduling case matching

Case matching refers to selecting historical EOS task scheduling cases from the case base that are similar to the current scheduling scenario. The assumption is that if the set of EOS and ground targets is similar in scheduling scenario A and scenario B, then their scheduling results should be similar as well. Thus, potential heuristic information can be obtained from historical cases that are similar to the

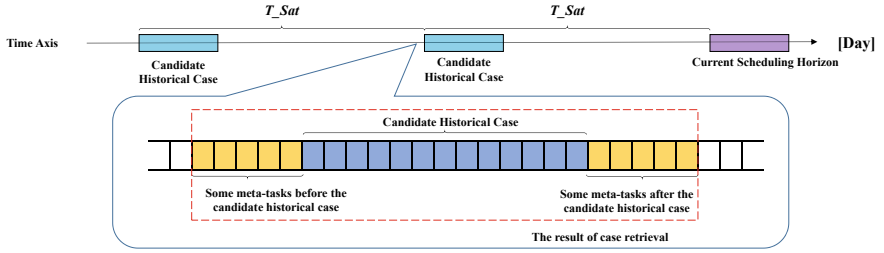


Fig. 3.20 Schematic diagram of satellite task scheduling case retrieval

current scheduling scenario, which can be used to solve the current EOSs scheduling problem. This section introduces two phases of similarity matching methods.

(1) Similarity matching based on target location

Target matching is the first phase of similarity matching used to estimate the similarity between two targets with the same priority. The similarity between two targets can be determined by checking if they have the same priority and if their spatial locations are close to each other. It is argued that targets with different priorities cannot be considered similar. For targets with the same priority, their spatial locations are analyzed to determine their similarity. Thus, similarity between point targets, between a point target and an area target, and between area targets is addressed and will be introduced, respectively.

An intuitive idea to determine whether two targets are similar in spatial location is to use a distance metric. For two point targets with the same priority ptar1, ptar2 $\in$ TARGET, we use the Manhattan distance to define the distance between two point targets as follows:

$$\text{distance}(\text{ptar1}, \text{ptar2}) = \left|\text{lon}_{\text{ptar1}} - \text{lon}_{\text{ptar2}}\right| + \left|\text{lat}_{\text{ptar1}} - \text{lat}_{\text{ptar2}}\right| \tag{3.33}$$

The two point targets are considered to be similar if the Manhattan distance between them is less than the distance threshold δ_{ptsi} which is a hyperparameter of our method.

For a point target ptar1 and an area target atar2 with the same priority, the similarity between them can be formulated as follows:

$$\text{distance}(\text{ptar1}, p) \leq \delta_{\text{ptsi}}, \ \exists p \in \text{atar2} \tag{3.34}$$

For two area targets atar1 and atar2, the similarity between them can be modeled as follows:

$$\text{distance}(p1, p2) \leq \delta_{\text{ptsi}}, \ \exists p1 \in \text{atar1}, \exists p2 \in \text{atar2} \tag{3.35}$$

(2) Single-circle similarity matching

The second phase of similarity matching is single-circle similarity matching. Single-circle matching is used to match a certain circle of scheduling tasks with a certain circle of historical tasks on the same satellite. They are considered similar if the amount of the similar targets between current scenario and historical case in the corresponding orbit circle up to the single-circle matching threshold δ_{circle}. For tasks belonging to one circle of a satellite, the similarity between a historical case and current scheduling scenario is illustrated as Fig. 3.21.

Usually, an EOSs scheduling scenario consists of several orbit circles. Therefore, the overall similarity between the current scheduling scenario and a historical case can be calculated based on the single-circle similarity. The overall satellite task similarity calculation is shown in Fig. 3.22.

Finally, the similarity between the current scheduling scenario and the historical case is judged based on the amount of similar circles, and the overall similarity measure of the two scenarios can be formulated as:

$$M_{\text{simi\_total}} = \frac{\text{match\_cir\_number}}{\text{total\_number}} \tag{3.36}$$

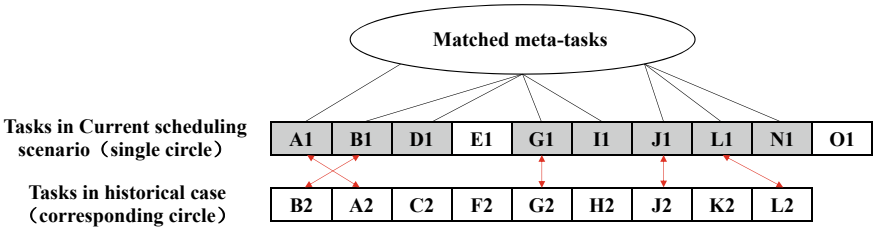


Fig. 3.21 Schematic diagram of single-circle similarity matching

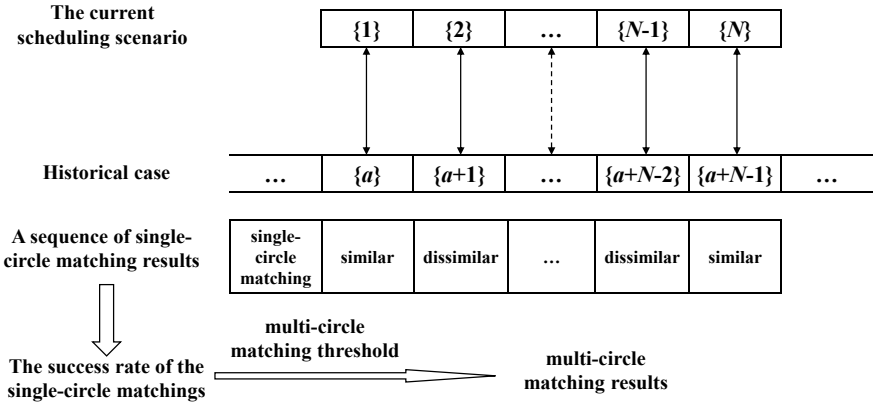


Fig. 3.22 Schematic diagram of the overall satellite task similarity determination

where match_cir_number is the number of orbit circles that can be matched between the current scheduling scenario and the historical case. total_number is the total number of the circles the scheduling scenario contains. If $M_{\text{simi\_total}} \geq \delta_{\text{total}}$, then the current scheduling scenario is considered to be similar to the historical case. Where δ_{total} is the threshold value to determine whether the current scheduling scenario is similar to the historical case.

3.5.3 Satellite Observation Programme Case Revision

For the selected cases with the most similar characteristics to the current scheduling scenario, the mismatched earth observation tasks in the historical scheduling scenario are revised so that the historical scheduling scenario can be better applied to the current scheduling computation. The revision method consists of two parts: removing tasks from the historical cases that do not match the tasks in the current scheduling scenario, and updating the satellite visible window and observation requirements of the tasks in the historical scheduling scenario.

3.5.4 EOS Task Scheduling Based on Case-Based Learning

The most crucial aspect of case-based learning is utilizing the heuristic information suggested by the historical cases to direct the scheduling calculations for the current scenario. To address the EOSs observation task scheduling problem, we must employ similar historical scheduling cases to guide the current search process toward the most promising areas of the solution space to obtain an optimal scheduling result. In this regard, we continue to use the genetic algorithm architecture to solve the EOSs observation task problem. It should be noted that the case-based learning approach presented in this section can also be applied to other meta-heuristic algorithms such as tabu search, ant colony algorithm, particle swarm optimization, and so on.

In the context of the regular genetic algorithm, case heuristic information can influence operators such as population initialization, selection operator, crossover operator, and mutation operator. Population initialization can be improved by randomly fine-tuning similar cases, which often result in high-quality individuals. The selection operator, crossover operator, and mutation operator can be designed to refer to the results of the state of the corresponding scheduling tasks in historical cases with a certain probability, thereby making use of the heuristic information provided by the cases.

Moreover, the historical cases are also obtained by the heuristic search algorithm which is the same as the current one, and there is still room for improving the optimization degree of the scheduling result. Overreferencing the historical cases may also lead the algorithm to continue to fall into local optimal solutions. In view of this, the satellite observation task scheduling based on case-based learning and

genetic algorithm (STSCGA) is designed. The problem coding is the same as that of the SOCSA algorithm introduced in Sect. 3.2.1. The main steps of STSCGA are as follows:

Algorithm name: STSCGA
Input: Earth observation meta-task TASK, the set of satellites SAT, a collection of similar cases that have been retrieved SCASE
Output: Earth observation programme $\{x_{\text{task}}^{s}\}$, task $\in$ TASK, $s \in$ SAT

```
begin
1   Randomly generating the population RdmGroup      // Population generation randomly
2   Set i = 1                                        // Iteration times of population evolution
3   while (i ≤ G1)
4       SOCSA_Selection ()                           // Selection operator used in SOCSA
5       SOCSA_Crossover ()                           // Crossover operator used in SOCSA
6       SOCSA_Mutation ()                            // Mutation operator used in SOCSA
7       Constraint handling for population
8   end while
9   Generating the second population CaseGroup based on historical cases SCASE
10  Combine CaseGroup with RdmGroup and form the whole population
11  while(G1 < i ≤ G2)
12      CBL_Selection ()                             // Case-based learning selection operator
13      CBL_Crossover ()                             // Case-based learning crossover operator
14      CBL_Mutation ()                              // Case-based learning mutation operator
15      if (The best individual of population has not changed after G3 consecutive iterations)
16          SOCSA_Selection ()                       // Selection operator used in SOCSA
17          SOCSA_Crossover ()                       // Crossover operator used in SOCSA
18          SOCSA_Mutation ()                        // Mutation operator used in SOCSA
19      end if
20      Constraint handling for population
21  end while
22  while(G2 < i ≤ G4)
23      SOCSA_Selection ()                           // Selection operator used in SOCSA
24      SOCSA_Crossover ()                           // Crossover operator used in SOCSA
25      SOCSA_Mutation ()                            // Mutation operator used in SOCSA
26      Constraint handling for population
27  end while
28  i = i + 1
29  Decoding and output Earth Observation Programme {x_task^s}, task ∈ TASK, s ∈ SAT
end
```

$G1$, $G2$, $G3$, and $G4$ are hyperparameter of STSCGA. The STSCGA algorithm comprises 29 statements that initialize the algorithm parameters (statements 1–2), adopt the selection, crossover, and mutation operators from SOCSA (statements 3–8), use fine-tuned historical earth observation programmes as high-quality individuals to form the second population (statement 9), merge the second population with the current population (statement 10), and apply case-based learning selection, crossover, and mutation operators to the evolutionary process (statements 11–21). The three case-based learning evolutionary operators are described in the subsequent

paragraphs. If the best individual of the population has not changed after $G3$ consecutive iterations, the algorithm switches to regular evolutionary operators (the selection, crossover, and mutation operators used in SOCSA) to tempt to jump out of the suspected local optima. The evolutionary process continues on the entire population using the same selection, crossover, and mutation operators from SOCSA (statements 22–27) to prevent local optima caused by case-based learning operators and exploit more optimal solutions. The evolution generations counter is updated (statement 28), and the results of SOCSA are outputted (statement 29) before the algorithm exits. Statements 3–8 can be seen as a "warm-up" process to the randomly generated population RdmGroup, preventing the rapid elimination of randomly generated individuals due to the high fitness value of the individuals generated from the historical cases, which would compromise population diversity. Statements 22–27 aim to perform unbiased genetic operations on the mixed population to reduce the probability of the population falling into a local optimum. Constraints are handled in the same way as in SOCSA (through statements 7, 20, and 26). The case-based learning population initialization, selection, crossover, and mutation operators are described as follows:

1. Case-based learning population initialization operator

The population initialization operator of case-based learning selects a similar historical case randomly and generates an individual by fine-tuning the historical case randomly. This process is executed as follows: for each meta-task in the current scheduling scenario, if a similar task is found in the historical case, the execution state of the meta-task is more likely to be consistent with the scheduling result of the historical case, with a higher probability (e.g., 80%). If no similar meta-task is found in the historical case, the execution state of the meta-task is determined randomly.

2. Case-based learning selection operator

The case-based learning selection operator selects individuals from RdmGroup and CaseGroup respectively by roulette method. The individuals selected form RdmGroup (which is generated by random initialization) constitute a half of the population, and the individuals selected form CaseGroup (which is generated by case-based learning initialization) form another half of the population. This can effectively maintain population diversity. The elite archive mechanism is also designed in the case-based learning selection operator, which is consistent with that of the SOCSA algorithm (introduced in Sect. 3.2.1).

3. Case-based learning crossover operator

The case-based learning crossover operator selects two parent individuals and does crossover to generate two offsprings. The crossover operation is the same as the crossover operation of SOCSA (introduced in Sect. 3.2.1). The difference between case-based learning crossover operator and the SOCSA crossover operator lies in how to select the two parent individuals. The parent individual selection strategy is the case-based learning crossover operator that has the equal probability of selecting two individuals from RdmGroup, or selecting two individuals from

CaseGroup, or selecting one individual from RdmGroup and another individual from RdmGroup. The probability of the three situations is 1/3, respectively. If the case-based learning crossover operator selects one individual from RdmGroup and another individual from RdmGroup, the two offsprings are randomly assigned to CaseGroup or RdmGroup.

4. Case-based learning mutation operator

The case-based learning mutation operator uses a random flipping variation approach, where an individual is randomly selected and a gene locus of that individual is randomly selected with a certain probability to determine whether the gene locus is reversed or not.

The probability of determining whether the meta-task execution represented by that gene locus is reversed can be obtained by counting the corresponding task execution status in the similar historical cases. For example, if the corresponding task is performed in a lot of historical cases, the task should be performed with a higher probability in the current scheduling scenario. The individuals to be mutated are selected from CaseGroup and RdmGroup with an equal probability.

5. Updating of the case base

The case base should be updated continuously, the current scheduling result should also be stored in case base. In this way, the satellite observation task scheduling for the current scenario and the historical case base updating become a closed loop. With the operation of case-based learning mechanism, the optimization degree of the newly generated scheduling results will continue to be improved. The historical scheduling cases with higher benefits can provide better support for the future scheduling scenario. This will form a virtuous circle.

Chapter 4
EOS Task Rescheduling for Dynamic Factors

Chapter 3 introduced a ground-based centralized satellite task scheduling model and methods that can be applied to the task scheduling problem of earth observation satellites (EOSs) in static scenarios. The static scenarios refer to situations in which the satellite resources and observation tasks engaged in scheduling scenarios will not change once the scheduling process begins. However, EOSs operate in a complex environment full of changes. They may experience temporary failures, be repaired, or have new observation tasks submitted at any time. Failing to account for these changes during the task scheduling process will inevitably result in suboptimal scheduling outcomes.

To address this problem, the study of EOS task rescheduling for dynamic factors has emerged as a hot topic in EOS task scheduling research [131]. As a necessary complement to the EOSs task scheduling under static conditions, this chapter will focus on the study of the EOSs task rescheduling method for dynamic factors. These factors mainly include the generation of new observation tasks and satellite failures/restarts.

4.1 Problem Description and Analysis

4.1.1 Classification and Analysis for Dynamic Factors

Satellite earth observation task rescheduling is generally caused by internal and external factors. Internal factors originate from the satellite system itself. The dynamic changes in power, storage space, and transmission resources of the satellite require adjustments to the original earth observation programme. External factors include changes in observation tasks (also referred to as observation requirements in some documents) and changes in the external environment during satellite operation (such as cloud occlusion caused by changes in meteorological conditions during optical satellite imaging). Two dynamic factors—the uncertainty of observation task

H. Chen et al., *Earth Observation Satellites*,
https://doi.org/10.1007/978-981-99-3565-9_4

emergence and the dynamic change of satellite resources—are present in all types of satellite task scheduling and are described in detail below.

1. The uncertainty of observation task emerging

In practice, new observation tasks may emerge at any time. Customers may submit new observation requests, and some of these requests may be urgent or essential (such as flood or volcanic eruption observation). Additionally, feedback information from ongoing observation tasks may require changes to the existing satellite observation programme. For example, observation tasks for moving targets in the ocean may require frequent revision as the predicted position of the moving target changes based on the last observation results. The ability to accommodate observation feedback significantly enhances the effective utilization of a set of satellite resources.

2. Changes in the dynamics of satellite resources

Satellites operate in a complex space environment, and their internal failures (such as telemetry equipment, sensor, power supply, or memory failure), environmental impacts (such as solar flares), or external factors (such as malicious attacks) can cause satellite resources to fail temporarily. As a result, observation tasks in the original satellite observation programme may not be carried out as intended, which can result in significant losses to end users of earth observation system. The aforementioned dynamic changes in satellite resources are almost unpredictable, making the implementation of satellite earth observation task rescheduling subject to a high degree of uncertainty.

4.1.2 Problem Modeling

To address the aforementioned uncertainties, satellite earth observation task rescheduling mainly adopts two approaches.

The first approach involves performing a complete rescheduling, which entails recalculating the task scheduling based on all current candidate observation tasks and satellite resources when dynamic factors emerge. The resulting satellite observation programme replaces the old one, and the two may be significantly different from each other. This may cause confusion among customers, as the response status to their numerous observation requests may change constantly. Additionally, the computation time required for a complete rescheduling is usually long. Therefore, complete rescheduling is not commonly used in practice.

The second approach, called satellite task partial rescheduling, involves making partial adjustments to the existing satellite observation programme. In other words, the original observation programme is revised partially to adapt to the new scheduling scenario, such as new tasks or satellite resources. This approach responds to dynamic factors while maximizing the benefits of the rescheduling process, making it widely used in practice.

This chapter focuses on satellite task partial rescheduling models and methods. Firstly, we provide a problem description and formulation.

1. Given a scheduling time horizon $w_{\text{schedule}} = [t_{\text{B}}, t_{\text{E}}]$. t_{B} and t_{E} are the start time and end time of the scheduling time horizon.
2. Given a EOSs set SAT. For $\forall s \in \text{SAT}$, $s \equiv \langle \text{MODE}^s, \text{memy}^s, \text{trans}^s_{i,j}, \text{pre}^s, \text{post}^s, \Delta T^s_{\text{m}}, \Delta T^s_{\text{l}}, \Delta T^s_{\text{lc}}, \Delta T^s_{\text{ld}}, \omega^s \rangle$. Where MODE^s is the set of observing modes of satellite s. For an optical satellite, MODE^s is the imaging roll angle of the satellite s. For a synthetic aperture radar (SAR) satellite, MODE^s is the set of parameters of SAR sensors; for electromagnetic detection satellites, the working mode is the parameters of the onboard electromagnetic signal receiver. memy^s is the total capacity of the onboard memory of satellite s. If the onboard memory capacity is full (also known as onboard memory overflow), the satellite cannot continue its earth observation task using the playback transmission mode. $\text{trans}^s_{i,j}$ is the transition time from working mode i to working mode j, in which $i, j \in \text{MODE}^s$. pre^s is the power-on preparation time of the satellite s. post^s is the power-off stabilization time of the satellite s. ΔT^s_{m} and ΔT^s_{l} denote the minimum and maximum single observing time of satellite s. ΔT^s_{lc} is the longest cumulative working time of the satellite s in a single circle. ΔT^s_{ld} is the longest cumulative working time of the satellite s in a single day. ω^s is the data acquiring/transmission ratio of the satellite s, i.e., the ratio of data generated per unit time from onboard earth observation sensor to the data transmitted per unit time of the data transmission payload. ω^s characterizes the data transmission capability of the satellite.
3. Given an observation target set TARGET. For $\forall \text{tar} \in \text{TARGET}$, $\text{tar} \equiv \langle \text{lon}_{\text{tar}}, \text{lat}_{\text{tar}}, \text{rot}_{\text{tar}} \rangle$ in which, lon_{tar} and lat_{tar} are the target's longitude and latitude. rot_{tar} is the maximum number of effective observations to the target tar. There is no furthermore reward if the observation times of tar are more than rot_{tar}.
4. Given a meta-task set TASK, which is described in Sect. 3.1.1. For $\forall k \in \text{TASK}$, $k \equiv \langle s_k, \text{mod}^s_k, \psi_k, t^k_{\text{b}}, t^k_{\text{e}}, \text{tar}_k, \text{circle}_k \rangle$. Where $s_k \in \text{SAT}$ denotes the satellite that perform task k in the VTW. $\text{mod}^s_k \in \text{MODE}^s$ denotes the working mode that satellite s_k takes to perform task k. ψ_k is the priority of task k that denotes the reward when k is performed. t^k_{b} and t^k_{e} is the start time and end time of the corresponding VTW. $\text{tar}_k \in \text{TARGET}$, denotes the task k corresponds to the ground target tar_k. $\text{circle}_k \in \mathbb{N}$ indicates the orbital circle of the satellite in which the current task is located.
5. For a satellite $s \in \text{SAT}$ after static scheduling (as presented in Chap. 3), the satellite observation programme of s is denoted as $\text{JOB}^s_{\text{init}}$.
6. In the case of new task emerged, given a set of new targets to be observed $\text{TARGET}_{\text{new}}$, and the corresponding set of new meta-tasks of satellite s can be denoted as $\text{task}^s_{\text{new}}$.
7. In the case of satellite resource failure, the set of changes can be denoted as DIST. For $\forall \delta \in \text{DIST}$, $\delta \equiv \langle s^\delta, t^\delta_{\text{off}}, t^\delta_{\text{on}} \rangle$. Where $s^\delta \in \text{SAT}$ is the satellite identifier, and t^δ_{off} is the time stamp when satellite k failed, and t^δ_{on} is the recovery

time stamp of satellite k. Thus, satellite is unavailable in the period $\left[t_{\text{off}}^{\delta}, t_{\text{on}}^{\delta}\right]$, $\left[t_{\text{off}}^{\delta}, t_{\text{on}}^{\delta}\right] \subseteq [t_{\text{B}}, t_{\text{E}}]$.

Based on the problem description above, for $s \in \text{SAT}$, the dynamic rescheduling for new tasks of satellite s is to insert new task set $\text{task}_{\text{new}}^{s}$ into the existed satellite observation programme $\text{JOB}_{\text{init}}^{s}$, resolve conflicts between tasks and form the rescheduling result $\text{JOB}_{\text{new}}^{s}$. Similarly, the dynamic rescheduling for satellite resource failure is to insert the tasks that cannot be performed by the failure satellite to the observation programme of the other remained satellites, and form the rescheduling result $\text{JOB}_{\text{new}}^{s}$.

In order to measure the change degree between original observation programme $\text{JOB}_{\text{init}}^{s}$ and the revised one $\text{JOB}_{\text{new}}^{s}$ of satellite s, the rescheduling change degree [125] is defined as follows.

Definition 4.1: Task change rate (TCR) ζ

$$\zeta = \frac{\sum_{s\in\text{SAT}}\left|\text{JOB}_{\text{init}}^{s} - \text{JOB}_{\text{new}}^{s}\right|}{\sum_{s\in\text{SAT}}\left|\text{JOB}_{\text{init}}^{s}\right|}. \tag{4.1}$$

ζ describes the degree of change between the initial scheduling result $\text{JOB}_{\text{init}}^{s}$ and the rescheduling result $\text{JOB}_{\text{new}}^{s}$. The difference between $\text{JOB}_{\text{init}}^{s}$ and $\text{JOB}_{\text{new}}^{s}$ is larger; the value of ζ is higher.

In practice, the upper limit $\zeta_{\max}$ of ζ is given as a constraint for rescheduling. Apparently, if all tasks of the initial scheduling result are required to be performed in the satellite observation programme after rescheduling, then $\zeta_{\max} = 0\%$.

4.1.3 Mapping Between Dynamic Factors

If the dynamic rescheduling for new tasks aims at finding more optimal scheduling results when new tasks emerge, the dynamic rescheduling for satellite resource failure is to reduce the loss in the case of satellite temporary unavailable. Once a satellite resource failure occurs, some satellites will not be available for a certain period of time, and the satellite observation programme for that period cannot be performed. The tasks that cannot be performed normally due to satellite resource failure is called failure tasks. The set of earth observation targets corresponding to the failed tasks is denoted as $\text{TARGET}_{\text{off}}$. In the rescheduling process, $\text{TARGET}_{\text{off}}$ needs to be assigned to the remaining available satellite resources, in order to reduce the loss caused by a part of satellite resources failure.

If we consider $\text{TARGET}_{\text{off}}$ as the set of new emerged earth observation targets $\text{TARGET}_{\text{new}}$, then the dynamic rescheduling for satellite resource failure can be

transformed into the dynamic rescheduling for new tasks. The mapping method is as follows.

1. Delete tasks that cannot be performed in JOB^{s}_{init}, due to the satellite resource failure. And the satellite observation programme after task deleting form JOB^{s}_{init} is denoted as JOB^{s}_{on}.
2. Filter unavailable satellite resources (the unavailable periods of satellite resources). The remaining available satellite resources can be denoted as SAT_ON.
3. Perform target access calculations between $TARGET_{off}$ and SAT_ON and form $\bigcup_{s\in SAT\_ON} Task^{s}_{off}$.

$\bigcup_{s\in SAT\_ON} Task^{s}_{off}$ can be regarded as the new tasks, and SAT_ON can be regarded as the set of available satellites in the current scenario. Therefore, dynamic rescheduling for satellite resource failure problem can be transfer to dynamic rescheduling for new tasks problem.

4.1.4 Driving Strategy of Heuristic Dynamic Rescheduling

The EOS task dynamic rescheduling algorithm is a reactive scheduling method that responds to dynamic factors. One important consideration is determining when to start the rescheduling process. There are three typical strategies: periodic-driven, event-driven, and hybrid-driven.

The periodic-driven strategy involves a fixed rescheduling period, where new tasks emerging in the current cycle must wait until the start of the next cycle before being handled. This strategy is also known as batch task scheduling.

In contrast, the event-driven strategy starts the rescheduling process immediately once a new task emerges. New tasks do not have to wait for the next cycle of the periodic-driven strategy, which allows them to be handled in a timely manner. However, the computation burden of this strategy can be high if new tasks emerge frequently.

The hybrid-driven strategy combines the periodic-driven and event-driven strategies. For normal new tasks, the rescheduling process starts under the periodic-driven strategy. For urgent or very important new tasks, the rescheduling process starts immediately under the event-driven strategy. This strategy is commonly used in practice.

Regardless of the driven strategy, when the rescheduling process starts, the rescheduling algorithm try to revised the existed satellite observation programme according to the new emerging tasks. Two typical EOS task rescheduling algorithms will be described in Sects. 4.2 and 4.3, respectively.

4.2 EOS Task Rescheduling Based on Heuristic Strategy

Rule-based heuristic rescheduling algorithms are one type of typical EOS task rescheduling methods. The basic idea of these algorithms is to insert the new task into a proper position of the existed satellite observation programme and resolve conflicts between the inserted new tasks and the original ones via heuristic rules. This section will introduce the details of rule-based heuristic EOS task rescheduling algorithm.

4.2.1 A General Method of Heuristic Dynamic Rescheduling

If we insert a new observation task into the existed satellite observation programme, we can see there are three relationship of the new task and the existed tasks: separation, intersection, and inclusion as shown in Fig. 4.1.

The rule-based heuristic EOS task rescheduling algorithm is designed to handle the three relationships between the new task and original tasks as shown in Fig. 4.1.

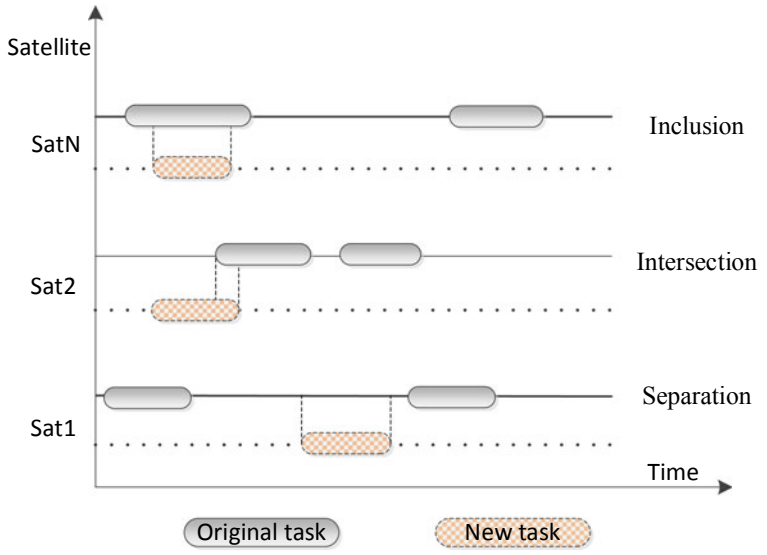


Fig. 4.1 Relationship between the new task and the original task in initial satellite observation programme

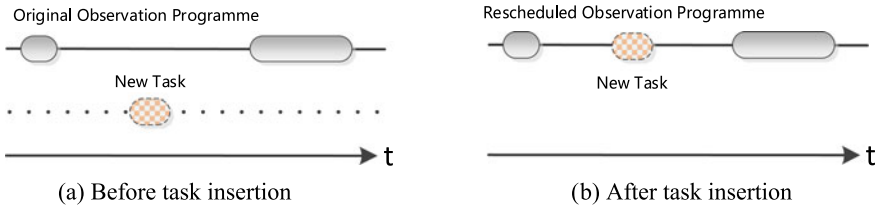


Fig. 4.2 Schematic diagram of the task insertion operator

The algorithm inserts new tasks and resolves conflicts between tasks using different operators, which are described as follows.

1. Task insertion operator

Task insertion operator can be used if there are enough available free slots in the original observation programme into which the new task can be inserted. Therefore, the task insertion operator is applicable when the relationship between the new task and the existed tasks is separation. The process of task insertion is shown in Fig. 4.2.

2. Task merging operator

Task merging is a technique used when there is insufficient free time in the original satellite observation programme to accommodate a new task. Instead, the new task can be merged with one of the existing tasks in the programme. This requires merging both the satellite visible window and the working mode of the two tasks. For instance, in the case of an optical imaging satellite, merging two tasks requires the observation time windows of both tasks to intersect, and the roll angle of the satellite to be recalculated to cover both tasks. Therefore, task merging is applicable when there is a relationship between the new task and the existing ones, either through intersection or inclusion. The process of task merging is illustrated in Fig. 4.3.

3. Task arbitration operator

Task arbitration operator is used in the situation that a new task has a time conflict with one of the tasks in the original observation programme and the two tasks cannot be merged. The task arbitration operator determines which task will be performed among the conflict tasks based on heuristic rules.

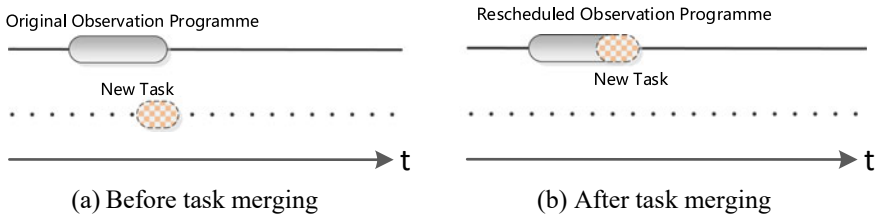
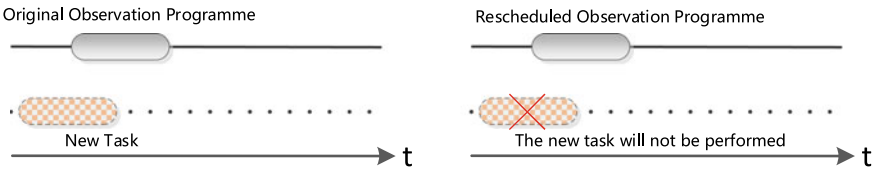


Fig. 4.3 Schematic diagram of the task merging operator



(a) Before task arbitration (b) After task arbitration

Fig. 4.4 Schematic diagram of the task arbitration operator

The heuristic rules used in EOS task scheduling are expert-designed and tailored to the specific requirements of satellite operation services. These rules aim to simplify and streamline the scheduling process by automating decision-making based on predetermined criteria. For example, a common heuristic rule is "higher priority tasks take precedence in arbitration." This means that a task with a higher priority will be prioritized over a lower-priority task, resulting in the removal of the latter from the observation programme. In Sect. 4.2.2, we will delve into the specifics of the heuristic rules employed in EOS task scheduling.

Therefore, the task arbitration operator is applicable when the relationship between the new task and the existed tasks is intersection or inclusion. The process of task arbitration is schematically shown in Fig. 4.4.

4.2.2 Rules of Heuristic Rescheduling for New Task Insertion

Dynamic rescheduling requires both optimality of the rescheduling observation programme and the computation timeliness; many scholars proposed heuristic rule-based strategies [51, 132] for new task insertion, which can be divided into two categories.

1. Task insertion based on priority

The task priority-based heuristic insertion rule is a simple and effective method for generating high-quality rescheduled observation programme [133]. The main idea of this rule is to prioritize tasks based on their importance, which is the objective function of EOS task scheduling. Firstly, the new tasks are sorted in descending order of priority. Then, they are inserted into the observation programme one by one. If the insertion satisfies the constraints, it is successful. Otherwise, a merging operation is attempted. If merging fails, an arbitration operation is used. Finally, the lowest priority tasks are dropped until all the constraints are satisfied, and the constraint handling process (as described in Sect. 3.2.1) is performed. This heuristic insertion rule is simple to implement and can respond quickly to dynamic rescheduling scenarios. However, it may not always produce the optimal solution.

The flow chart of the task insertion rules based on priority is shown in Fig. 4.5.

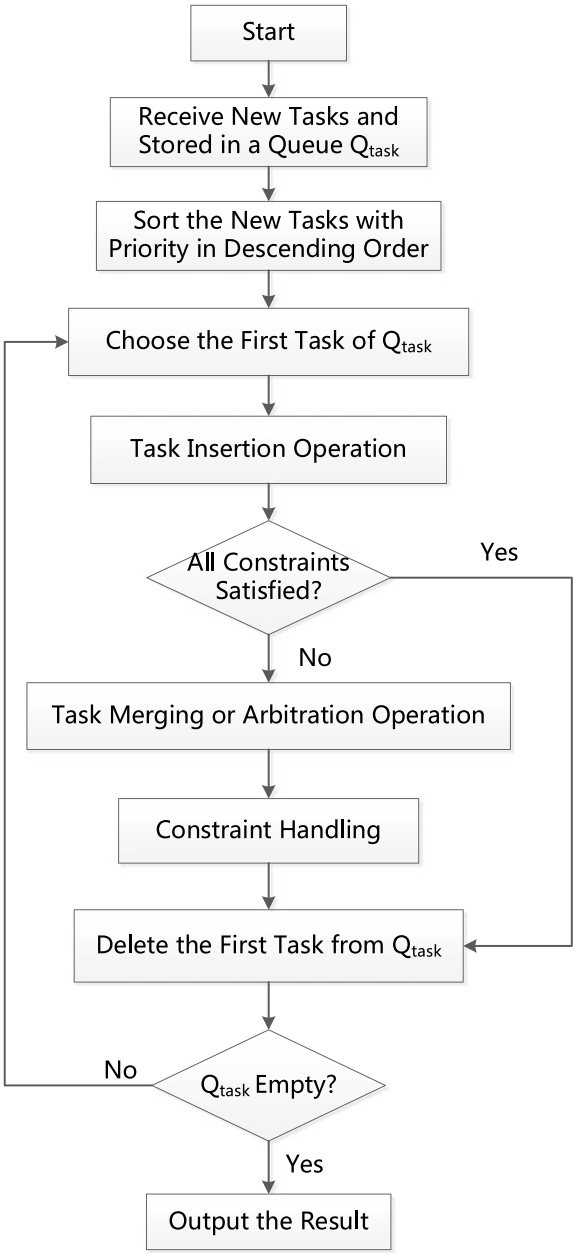


Fig. 4.5 Flow chart of the task insertion rules based on priority

It is noted that we also can replace the priority with other indicators to determine whether the new task can take place of the existed one when conflict occurs. The other indicator most adopted in EOS task rescheduling is task opportunity factor ρ_i [61, 131].

The task opportunity factor ρ_i is defined as:

$$\rho_i = \frac{\psi_{\text{task}}}{\text{Num\_win}_i}, \tag{4.2}$$

where ψ_{task} is the priority of the observation task, and Num_win_i refers to the total number of the satellite visible windows to the ground target.

From the definition of the task opportunity factor ρ_i, we can see the number of the satellite visible windows of the ground target refers to the observation opportunities of the ground target. If the number of the satellite visible windows is high, there are a lot of opportunities to complete the observation, and the urgency (ρ_i) of each observation task to the ground target is low.

2. Iterative repair-based task insertion

Unlike the direct task insertion based on task priority, the iterative repair-based task insertion considers not only the priority of the new task but also the impact of the new task insertion to the whole observation programme. It searches for the opportunity of the inserting the dropped tasks into the observation programme again.

The insertion of a new observation task with higher priority may result in the dropping of an existing task (tasks). In the absence of the dropped task (tasks), a new observation time window may become available in the observation programme, creating an opportunity to insert another candidate task (other candidate tasks). To manage the dropped tasks, a conflict task queue is utilized, which stores them in order of their priority (from high to low). During the rescheduling process, the tasks in the conflict task queue are revisited and reinserted into the observation programme to improve its optimization.

For example, a typical rescheduling process is shown in Fig. 4.6. In Fig. 4.6(1), new task A is inserted in to the observation programme successfully, and task M and N are dropped, since they conflict with higher priority task A. Both task M and N are stored in the conflict task queue. In Fig. 4.6(2), the priority of new task B is higher than that of task A; thus, task A is replaced by task B, and task A is also stored in the conflict task queue. In Fig. 4.6(3), the rescheduling algorithm tries to insert the tasks of the conflict task queue into the observation programme. Currently, the conflict task queue contains task A, M and N. Apparently, task A cannot be inserted successfully, while task M and N can. Figure 4.6(4) shows the final satellite observation programme. The status of the conflict task queue is shown in Table 4.1.

The flow chart of the iterative repair-based task insertion rule is shown in Fig. 4.7.

In summary, the distinction between the task insertion methods based on priority rules and iterative repair lies in their approach to handling dropped tasks. The iterative repair-based method stores dropped tasks in a conflict task queue and attempts to re-insert it into the observation programme, whereas the priority-based method takes

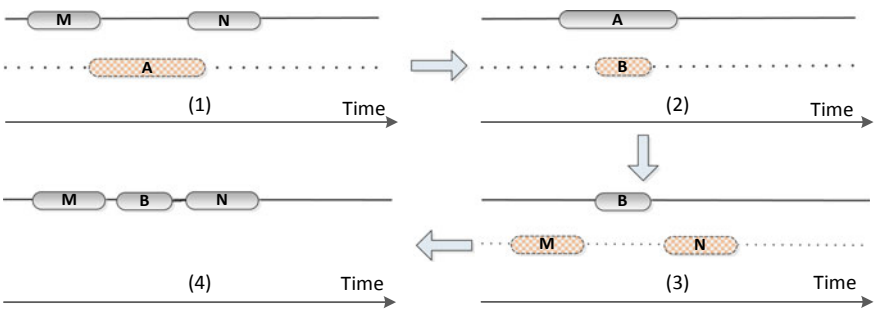


Fig. 4.6 Typical rescheduling process with the conflict task queue

Table 4.1 Elements stored in the conflict task queue at each step of Fig. 4.6

Step	Handling task	Elements in the conflict task queue
(1)	*A*	Null
(2)	*B*	*M* and *N*
(3)	*M* and *N*	*A*, *M* and *N*
(4)	Null	*A*

no action. While the iterative repair-based method may yield more optimal results, it also incurs additional computation time. The priority-based method is more suitable for applications with the highest timeliness requirements in an urgent rescheduling situation.

4.3 EOS Task Rescheduling Based on Intelligent Optimization Operator

4.3.1 EOS Task Rescheduling Based on SWO

The heuristic rule-based dynamic rescheduling method for EOS tasks presented in the previous section involves selecting appropriate insertion locations for new tasks using heuristic policies. This approach enables a quick response to task rescheduling scenarios. In this section, we will introduce the EOS task dynamic rescheduling method based on heuristic local search, which can typically produce more optimal results than heuristic rule-based rescheduling methods, albeit with a slightly longer computation time.

Squeaky-Wheel Optimization (SWO) is a typical heuristic local search method that has been successfully applied in several fields such as graph coloring, satellite range scheduling, space-based astronomical observation scheduling, etc. [125]. In

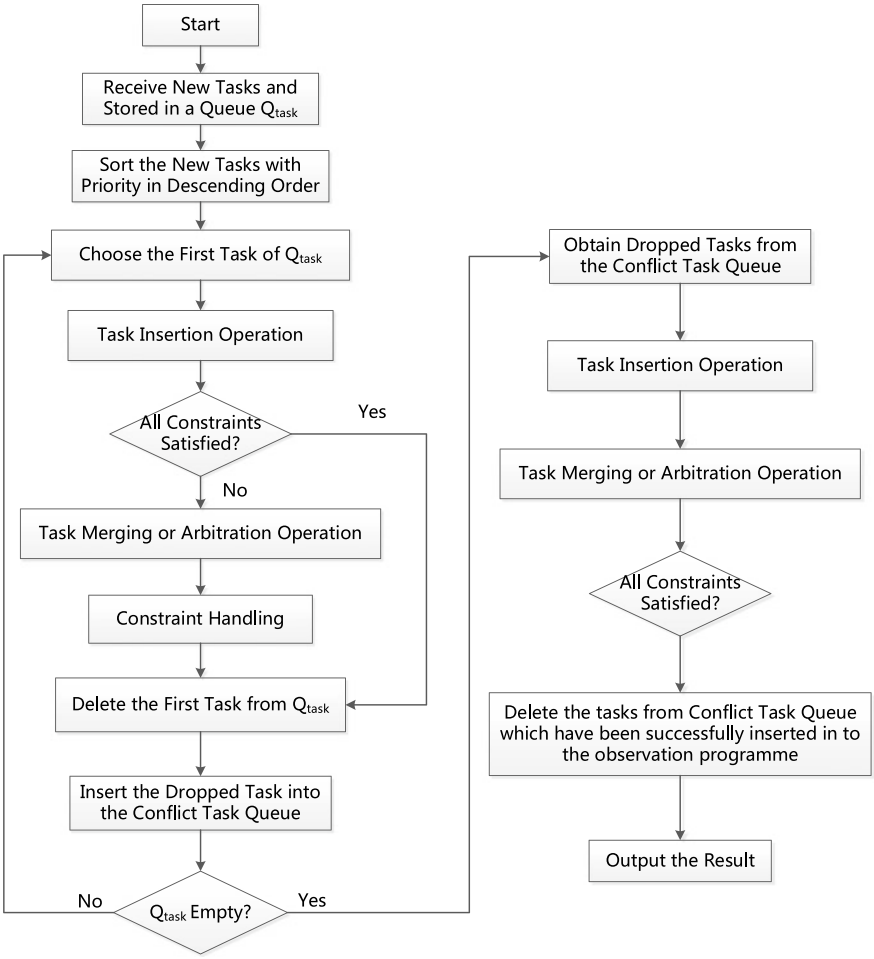


Fig. 4.7 Flow chart of the iterative repair-based task insertion rule

this section, we present the Satellites Dynamic reScheduling Algorithm based on memorized SWO (SDSA) [134].

1. Basic SWO optimization operator

The core idea of the SWO operator is an iterative cycle of "construct-analyze-priority adjustment" as shown in Fig. 4.8. The SWO operator consists of three modules, namely a constructor, an analyzer and a prioritizer.

In a scheduling problem, a solution is constructed by a greedy algorithm based on task priorities. The analyzer then evaluates the scheduling results generated by the constructor to identify any shortcomings and assigns blame values to each task. Next, the prioritizer reorders the tasks based on their blame values to guide the constructor

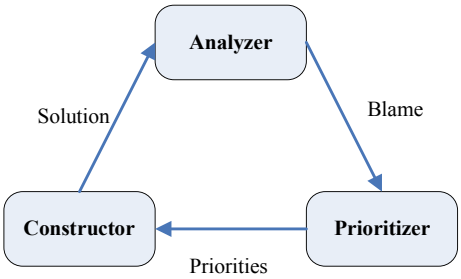


Fig. 4.8 Schematic diagram of the optimization process of the SWO operator

to prioritize the tasks with higher blame values and generate a new solution. This cycle continues until a termination condition is met.

2. Design of dynamic rescheduling algorithm based on SWO with memory

In SDSA, tasks can be divided into two categories: scheduled observation task sequence and unscheduled observation task sequences. In the initial state, the scheduled observation task sequence is $\bigcup_{s\in SAT} JOB^s_{init}$ and the unscheduled task sequence is $\bigcup_{s\in SAT} Task^s_{new}$. During the iteration of the SDSA algorithm, the SWO operator continuously attempts to assign unscheduled tasks with high blame value and performs constraint handling on the scheduled task sequence to drop the conflicting tasks with low blame value, and add these dropped tasks to the unscheduled task sequence. The scheduled task sequence of SDSA is the rescheduling result. The framework of SDSA algorithm is described below.

Algorithm name: SDSA
Input: original scheduling result $\bigcup_{s\in SAT} JOB^s_{init}$, new tasks set $\bigcup_{s\in SAT} Task^s_{new}$
Output: dynamic rescheduling result $\bigcup_{s\in SAT} JOB^s_{new}$

begin
1 set Assigned $= \bigcup_{s\in SAT} JOB^s_{init}$, UnAssigned $= \bigcup_{s\in SAT} Task^s_{new}$
2 Blame = Analyzer (UnAssigned)
3 Prioritizer (UnAssigned, Blame)
4 (Assigned, UnAssigned) = Constructor (Assigned, UnAssigned)
5 if the algorithm satisfies the end condition then
6 exit
7 else goto 2
8 end if
9 $\bigcup_{s\in SAT} JOB^s_{new}$ = Assigned
end

In the SDSA algorithm, statements 2–4 call the analyzer, prioritizer and constructor of SWO, respectively, for iterative optimization. Statements 5–8 perform the end condition judgment of the algorithm and exit if the algorithm satisfies the end condition, otherwise continue the cycle optimization process.

Unlike the basic SWO operator, in SDSA, the analyzer and prioritizer act only on the unscheduled task sequence (UnAssigned), while the constructor acts on both the

scheduled task sequence (Assigned) and unscheduled task sequence (UnAssigned). To prevent tasks from being repeatedly scheduled and dropped, a memory module is introduced in the iterative optimization process of SWO. Details about the analyzer, prioritizer, and constructor in SDSA are described below.

1. The analyzer

The analyzer is the key component of the SWO, and the capability of analyzer will directly affect the optimization performance of SWO. The analyzer in SDSA gives a blame value to every element of UnAssigned (the unscheduled task sequence).

Similar to Chap. 3, the task importance objective is mainly considered. Therefore, the task with higher priority is considered to have a larger blame value. For a task i, the blame value of i can be formulated as:

$$\text{Blame}^i = \frac{\psi_{T^i}}{\sum_{j \in \text{UnAssigned}} \psi_{T^j}}. \tag{4.3}$$

In which, T^i is the observation target of task i, and ψ_{T^i} is the priority of the observation target T^i. In the iterative optimization process, we also have to consider some other factors and revise the blame value of task i. The factors are as follows.

- Task change rate. If the task change rate ζ exceeds its upper-bound $\zeta_{\max}$, then the blame value of the tasks in original observation task sequence ($\bigcup_{s \in \text{SAT}} \text{JOB}^s_{\text{init}}$) increases. The effect of task change rate to the blame value of task i can be formulated as follow.

$$\begin{aligned} &\text{if } \zeta > \zeta_{\max},\ i \in \text{UnAssigned} \cap \left(\bigcup\nolimits_{s \in \text{SAT}} \text{JOB}^s_{\text{init}} \right) \\ &\text{Blame}^i = \text{Blame}^i + \max_{j \in \text{UnAssigned}} \left(\text{Blame}^j \right). \end{aligned} \tag{4.4}$$

- Memory effect. To prevent a task from being repeatedly scheduled, dropped, and rescheduled during the optimization cycle, we designed a list that keeps track of the number of times each task has been scheduled and dropped. If a task is scheduled and then dropped, its blame value is reduced since it can be regarded as repetitive and useless work. The more times a task is scheduled and dropped, the smaller its blame value becomes. The memory effect on the blame value of a task for repetitive work can be formulated as follows.

$$\forall i \in \text{UnAssigned},\ \text{Blame}^i = \text{Blame}^i \cdot (\gamma)^{\text{ADT}^i}. \tag{4.5}$$

In which, ADT^i is the number of times of the task i to be scheduled and then dropped. $\gamma \in (0, 1)$ is the decay factor. The introduction of memory effect to the SWO

operator is beneficial to reduce the computation time and accelerate the convergence speed of the SDSA.

2. The prioritizer

The prioritizer sorts the tasks in UnAssigned (the unscheduled task sequence) in descending order of the blame value. The "QuickSort" algorithm is employed in the prioritizer of the SDSA.

3. The constructor

The constructor gets the tasks from UnAssigned (the unscheduled task sequence) that has been sorted by prioritizer one by one and make attempt to insert them into Assigned (the scheduled task sequence) in turn.

If the scheduled tasks do not conflict with the newly inserted task, the insertion is considered successful. Otherwise, the rescheduling evaluation value (e.g., priority of a task) of the current task to be inserted and the scheduled task(s) that conflict with it are calculated. The task with the higher rescheduling evaluation value will remain in the observation programme, while the other conflicting task(s) will be dropped.

If the one conflicting with the current task to be inserted is also newly inserted task in this round, the current meta-task will be dropped directly. The main steps of the constructor of the SDSA are as follows.

Algorithm name: SWO_Constructor
Input: scheduled task sequence Assigned, unscheduled task sequence UnAssigned
Output: updated scheduled task sequence Assigned_new
updated unscheduled task sequence UnAssigned_new

```
begin
1  set Assigned_new = Assigned, UnAssigned_new = φ, UnAssigned1 = UnAssigned
2  while UnAssigned1! = φ
3        Fetch the first element k from UnAssigned
4        ConflictSet = GetConflictTask (Assigned_new, k)
5        if ConflictSet = φ then
6              Insert the k into Assigned_new
7        else if ConflictSet ∩ UnAssigned1 = NULL then
8              calculate the rescheduling evaluation value of k and elements of ConflictSet
9              if (The sum of the evaluation values of all tasks in ConflictSet ≤ that of k)
                 or (task ∈ ⋃_{s∈SAT} JOB^s_init, and ξ ≥ ξ_max) then
10                   Drop all task contained in ConflictSet from Assigned_new
11                   Insert k into Assigned_new
12                   Add all elements of ConflictSet to UnAssigned_new
13             end if
14       else insert k into UnAssigned_new
15       end if
16       Delete k from UnAssigned1
17 end while
end
```

In SWO_Constructor algorithm, Statement 5 obtains the set of tasks which conflict with the task to be inserted (k).

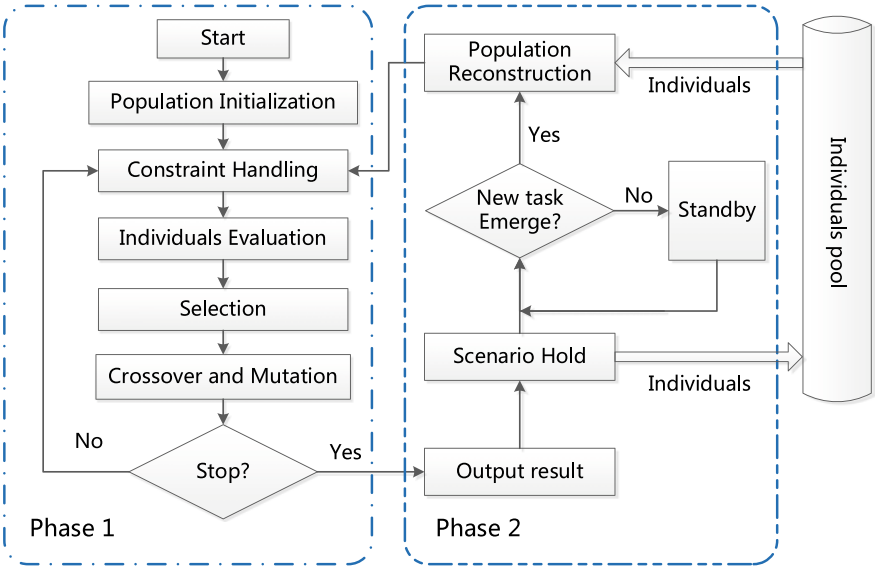


Fig. 4.9 Main steps of the SORSA algorithm

4.3.2 EOS Task Rescheduling Based on Evolutionary Computation

In Chap. 3, we introduced SOCSA, an EOS task scheduling algorithm under static situations. SOCSA is based on the evolutionary computation method, which naturally accommodates dynamic rescheduling scenarios. Therefore, this section aims to extend the capabilities of SOCSA and propose a new algorithm, Satellites Observation Task Re-Scheduling Algorithm based on evolutionary computation (SORSA). The framework of SORSA is shown in Fig. 4.9.

From Fig. 4.9, we can see that the running state of SORSA can be divided into two running phases.

After SORSA starts running, it enters phase 1. The population is randomly generated, and the typical evolutionary cycle (namely the fitness evaluation, selection, crossover, and mutation) is running. When the exit condition is satisfied, the algorithm outputs the results and enters the phase 2.

In the phase 2, all information of the current population is saved to the individuals pool, also known as "scenario hold" operation, and the algorithm enters standby mode. SORSA remains in standby mode unless new tasks emerge. When new tasks emerge, a population is generated that contains both the new tasks and the original tasks based on the basic structure of individual information in the individuals pool. The algorithm then enters phase 1 again. SORSA operates in this cycle. The encoding method, population initialization method, fitness evaluation, crossover, mutation, and

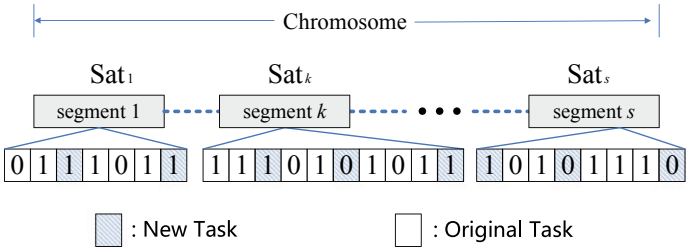


Fig. 4.10 Schematic of the population reconstruction when new tasks emerge

selection operator of SORSA are identical to those of SOCSA (described in Chap. 3). Therefore, we can conclude that phase 1 of SORSA is equivalent to SOCSA.

The following will focus on the two main steps in phase 2 of SORSA, i.e., scenario hold and population reconstruction.

1. The scenario hold operation of SORSA

The scenario hold operation is to save the individuals in the current population to the individuals pool. It is noted that a de-duplication operation needs to be performed when saving new individuals into the individuals pool, since there may be identical individuals in the current population.

2. The population reconstruction of SORSA

The population reconstruction is the process of retrieving the individuals from the individuals pool and combining them with the new observation tasks to form a new population. The process of the population reconstruction is shown in Fig. 4.10.

In Fig. 4.10, the twilled squares indicate new tasks and the white squares indicate original tasks. The encoding of SORSA is identical to that of SOCSA (details in Sect. 3.2.1), "1" means that the task will be performed and "0" otherwise. The tasks of the same satellite are sorted by the start time of observation.

The de-duplication operation is performed in the scenario hold operation, which results in a smaller number of individuals in the individuals pool compared to the population size. To form the new population, all individuals in the individuals pool are first retrieved. If the size of the new population is still insufficient, individuals are randomly retrieved from the individuals pool with equal probability and added to the new population until the population size requirement is met. The coding sequences of individuals are updated to contain the new tasks, and their execution states are randomly initialized. There is no need to be concerned about whether the newly generated individuals may violate any constraints, as the constraint handling will be addressed in phase 1.

Chapter 5
Distributed Satellite Task Scheduling Models and Methods

This chapter presents an introduction to satellite task scheduling methods in distributed scenarios. Both distributed and centralized satellite task scheduling have their own advantages and disadvantages, which are applicable to different application scenarios. Together, they form the mainstream technology system of multi-satellite joint task scheduling. The distributed satellite task scheduling is based on the multi-agent system (MAS) theory in distributed artificial intelligence. In this approach, each satellite participating in the scheduling process is considered an intelligent agent undertaking earth observation tasks based on its capabilities and benefits. These satellite agents interact with each other to rapidly form a multi-satellite observation programme.

The chapter first describes the multi-satellite distributed task scheduling problem and analyzes its application scenarios along with the multi-agent system architecture. Then, it establishes the distributed satellite task scheduling model. Finally, it proposes the distributed satellite task scheduling method based on the cooperation mechanism, such as contract network protocol and blackboard model.

5.1 Problem Description and Analysis

5.1.1 Formulation of the Distributed Satellite Task Scheduling Problem

In a centralized scheduling scenario, multiple satellites typically perform earth observation tasks under the unified management of the EOSOC. The EOSOC can conduct centralized scheduling based on obtaining all satellite information and data transmission resources, giving it good global optimization and solution capabilities. However,

H. Chen et al., *Earth Observation Satellites*,
https://doi.org/10.1007/978-981-99-3565-9_5

the centralized satellite task scheduling model has some limitations that should be considered.

(1) Longer solving time: The computational complexity is high because the global optimization algorithm solves the problem based on all satellite information. As the number of observation requests and the number of satellites increase, the computational operation becomes enormous, resulting in a significant increase in computation time [24].
(2) Low dynamic adaptability: In the centralized task scheduling scenario, if a new task arrives, it is necessary to replan all tasks to develop a new earth observation programme or use a heuristic correction method to locally fine-tune the current programme. However, rescheduling is time-consuming and typically undesirable. If it becomes the norm for new tasks to arrive at any time, the earth observation programme will be repeatedly modified, and its optimality will be drastically reduced [135]. Consequently, centralized scheduling for satellite tasks makes responding reasonably to a dynamic changing environment of tasks and resources challenging.
(3) Insufficient scalability: The centralized task scheduling method is closely coupled with specific satellite constraints. When a new satellite is deployed, the original scheduling algorithm must be adjusted to accommodate the change, making expanding the centralized scheduling method challenging. If a newly added satellite's constraints and usage rules differ significantly, it may even require redesigning the existing centralized task scheduling method [136].
(4) Weak encapsulation of building blocks: In the centralized task scheduling method, the capabilities and constraints of each satellite need to be mathematically modeled, requiring the satellite center to master their technical parameters. However, in practice, various satellite operation centers usually manage different series of satellites, and technical parameters are generally inconvenient to disclose to each other. Consequently, applying the centralized satellite task scheduling method in such scenarios is challenging [137].

The emergence of the distributed satellite task scheduling method has addressed the limitations of the centralized approach. In this model, multiple satellite task planners are represented as intelligent agents that exchange information on tasks, resources, equipment status, and other relevant data to plan and decide on their tasks, leading to the collaborative observation of ground targets. The distributed method offers several advantages over the centralized approach.

Firstly, the algorithms used in distributed satellite task scheduling are naturally parallel, and satellite agents can be deployed on different computing nodes. As a result, the allocation of earth observation tasks among satellite agents is accomplished efficiently through negotiation algorithms, reducing computation time.

Secondly, the distributed task scheduling approach operates in an online processing mode, enabling real-time processing of earth observation tasks. When a task is received, each satellite agent determines whether to execute it based on available resources while negotiating with other agents for task repetition. This differs from the centralized approach, which employs global optimization for initial scheduling and local heuristic correction for dynamic rescheduling.

Thirdly, the distributed task scheduling is based on the theory of multi-agent systems in distributed artificial intelligence, allowing new satellite deployment by modeling it as a new satellite agent and registering it into the multi-agent system. Similarly, satellite failure or withdrawal only requires the cancelation of the corresponding agent in the system.

Lastly, distributed satellite task scheduling allows for modeling EOSs or EOSOCs as agents, encapsulating task scheduling calculations within the agent. This enables the distributed collaborative task scheduling of satellites between multiple EOSOCs using negotiation algorithms without disclosing technical details of satellites under each EOSOC's jurisdiction. Joint negotiation algorithms can be used between multiple satellite centers to distribute earth observation tasks.

The use of distributed satellite task scheduling theory effectively addresses the limitations of centralized satellite task scheduling. To clarify our discussion, we shall refer to the multiple satellites involved in the scheduling process as distributed satellite clusters in the context of distributed satellite task scheduling. This scheduling method is grounded in the theory of agent and multi-agent systems in distributed artificial intelligence, which we will elaborate on in the subsequent sections.

5.1.2 Introduction to Agent and Multi-agent Systems

In the mid-1950s, McCathy [138] first proposed the idea of an agent. Since then, agents, particularly multi-agent systems, have become a popular topic in artificial intelligence and computer science [139, 140]. The original definition of an agent referred to someone who performed specific tasks on behalf of another individual or organization. However, in computational science, an agent is a hardware or software-based computer system with autonomy, reactivity, social ability, and proactiveness.

In the context of distributed satellite task scheduling, an agent has specific characteristics: (a) it exists in a particular environment and can perceive and influence it, (b) it has an autonomous purpose and can arrange activities to achieve its goal, and (c) it

can exchange information with other agents in the environment. An agent typically possesses three critical properties: perceptiveness, autonomy, and interactivity.

1. Perceptiveness

The external environment can influence an agent's problem-solving behavior and strategy. Therefore, it is necessary for the agent to continuously perceive any changes in the environment to ensure the goal's relevance and the programme's viability.

2. Autonomy

An agent possesses the capability of independent thinking and decision-making. It can make independent decisions without any human or agent intervention. It can adjust its behavior in response to changes within itself or in the external environment. Moreover, the agent can integrate information from other agents into its decision-making process.

3. Interactivity

Communication can occur through specific methods in a virtual environment with multiple agents. Each agent can refer to the information provided by other agents during independent decision-making, reason about the data received, and learn from the shared experiences.

Multi-agent systems have emerged as a prominent research topic in Distributed Artificial Intelligence (DAI). DAI is concerned with exploring how intelligent systems can achieve problem-solving in a logically or physically decentralized manner while collaborating. This approach is a computer simulation of the collaborative division of labor and cooperation mechanisms observed in humans [141]. DAI is comprised of two critical branches: distributed problem-solving (DPS) and multi-agent system (MAS). As research in this area progresses, there is a gradual integration of the two branches. Based on references [142, 143], MAS can be considered a further development of DPS.

A MAS is a distributed and autonomous system composed of multiple mutually independent agents, working together toward accomplishing specific tasks or achieving common goals. MAS can be viewed as a virtual society of agents based on the principles of bounded rationality [144] and society of mind [145]. Each agent cooperates through reasoning, planning, negotiation, and negotiation to jointly complete complex tasks that are difficult for a single agent to complete. The autonomy of each agent and its ability to collaborate with others is a fundamental characteristic of MAS. Each agent in MAS has access to only partial information, resulting in a localized perspective and problem-solving ability. This contrasts with single-agent systems, where a central controller typically has access to all the information. In addition, MAS has no central control, with data being decentralized or distributed. The computation process is typically asynchronous, concurrent, or parallel. In summary, the primary features of MAS include autonomy, collaboration, localized knowledge,

decentralized or distributed data, and asynchronous, concurrent, or parallel computation processes. These characteristics make MAS suitable for solving complex problems that would be difficult to solve with a single-agent system. Multi-agent systems have the following main characteristics.

1. Social nature

Within a multi-agent system, each agent operates as a member of a virtual society composed of several agents. As a part of this virtual society, each agent can communicate, cooperate, negotiate, compete, manage, and control other agents to achieve its objectives and contribute to the overall social value of the system.

2. Isomorphism

In a multi-agent system, each agent typically possesses unique capabilities that enable them to perform different collaborative divisions of labor. Consequently, a multi-agent system composed of agents with varying task-processing capabilities exhibits heterogeneous characteristics.

3. Collaborative nature

In a virtual society, each agent must undertake complex tasks that often require collaboration with other agents. Thus, collaboration is a fundamental element of a multi-agent system, enabling members with diverse objectives to negotiate and cooperate to enhance the problem-solving ability of the system as a whole.

Multi-agent system technology can express complex systems, as each member agent within a given multi-agent system possesses social, heterogeneous, and collaborative characteristics. As such, it provides a unified model and framework for various practical systems.

In the distributed satellite task scheduling process, the satellite cluster can be considered a distributed intelligent system, resembling a virtual society. Each satellite has autonomous capabilities and can collaborate to perform complex earth observation tasks. Accordingly, current research aims to model the distributed satellite task scheduling problem using multi-agent systems.

5.2 Distributed Satellite Task Scheduling Model Based on Multi-agent Systems

5.2.1 Social Role Analysis of Multi-agent Systems

The first step in developing a distributed satellite task scheduling model using a multi-agent system is to identify the elements and objects within the satellite cluster that must be mapped as agents. Given that individual earth observation satellites are independent of each other and that the satellites are distributed across different locations, each satellite and its internal components in the cluster are grouped together and mapped as a single agent. This approach results in all satellite agents forming a multi-agent system. As each agent in the multi-agent system has unique social

and heterogeneous characteristics, the composition structure of the satellite cluster multi-agent system varies based on the collaborative division of labor among the satellite agents.

To further classify the collaborative division of labor among satellites in a distributed satellite cluster, the characteristics of various multi-agent system composition structures are examined. For instance, Schetter et al. [146] conducted a study of the TechSat 21 project, which analyzed the collaborative division of labor among satellites with varying autonomous capabilities in a distributed satellite cluster. According to Schetter et al.'s findings, satellite agents can be classified into four levels. I_1, I_2, I_3 and I_4 as shown in Fig. 5.1. Among them, the I_1 represents the satellite agent with the strongest autonomy capability, while I_4 the satellite agent with the weakest autonomy capability.

I_4 indicates that a satellite agent is only responsible for receiving and executing tasks. It can only receive and execute commands and tasks from other satellite agents. The satellite is only a task executor.

I_3 denotes that a satellite agent has local planning capability, which means it can only generate planning results related to its tasks. Such a planning process involves only the satellite itself and not the entire satellite cluster.

I_2 represents the satellite agent can interact with other satellite agents. This kind of agent can resolve conflicts with other satellite agents by coordination and negotiation to improve the whole system's performance. It requires the agent to have partial knowledge of the multi-agent system, i.e., the relevant knowledge of other satellite agents. Still, it cannot obtain all information within the whole satellite cluster.

I_1 denotes the satellite agent with the most substantial autonomy in the satellite cluster, and this type of satellite agent can obtain all information of the satellite

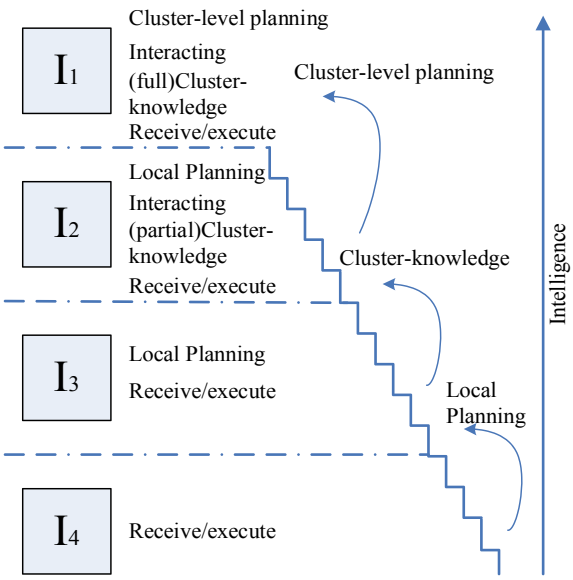


Fig. 5.1 Classification of satellite agents for different social roles [146]

cluster. It can generate the earth observation programme of the satellite cluster and assign the tasks undertaken to the appropriate satellites for execution.

By dividing their work and assuming responsibilities based on the four levels of autonomy described above, satellite agents can establish satellite clusters with distinct multi-agent system structures. Schetter et al. [146] have proposed several general architectures for such structures, including top-down, centralized, distributed, and fully distributed forms. Figure 5.2 displays the schematic diagram and interrelationships of these structures.

Figure 5.2 gives the structure of various multi-agent systems with different autonomy levels. As can be seen from the figure, the top-down coordination structure forms a master–slave structure. The I_1-level satellite agent makes the programme of the whole satellite cluster and assigns the relevant tasks to the bottom I_4-level satellite agent to execute. Located at the bottom of the centralized coordination structure are I_2-level and I_3-level satellite agents. In the centralized structure, the I_1-level satellite agent first plans the tasks of the whole satellite cluster and assigns specific tasks to I_2-level and I_3-level satellite agent, and then I_2-level and I_3-level satellite agent performs further scheduling and finally determines the execution plan. The multi-agent system with distributed coordination structure is an ideal structure with

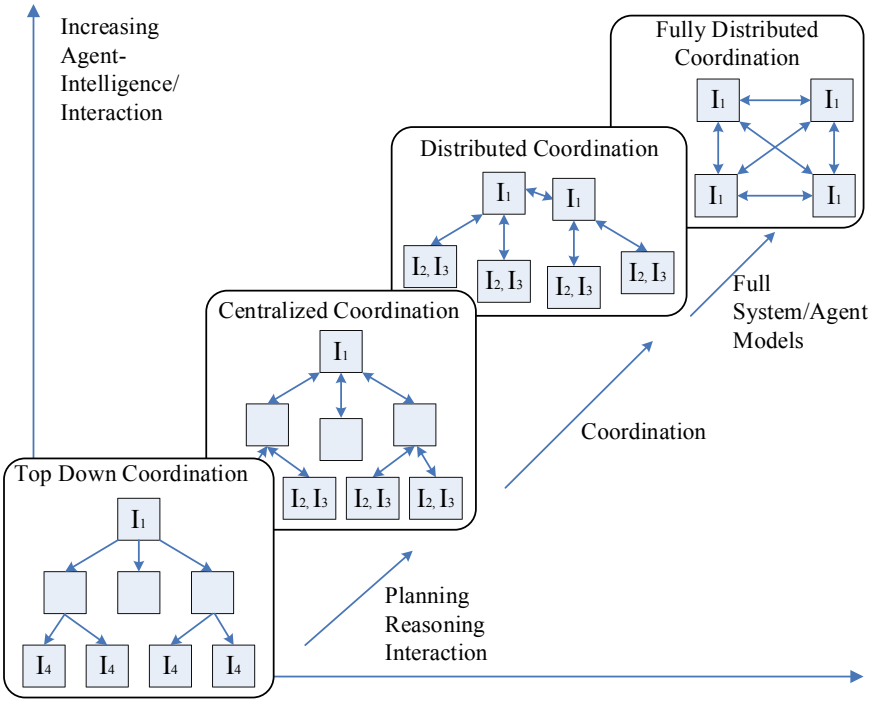


Fig. 5.2 Various multi-agent structures of distributed satellite clusters [146]

distributed hierarchy, which can be regarded as the union of several centralized coordination structures. This structure can fully exploit each satellite agent's adaptability, distribution, and autonomy capabilities. The fully distributed coordination structure assumes that all satellites in the satellite cluster have I_1-level autonomous capability and the planning and decision computation at the satellite cluster can be performed among all satellite agents. The fully distributed coordination structure has no hierarchy among the satellite agents, and its flat organizational structure guarantees the system a high degree of flexibility and reliability. Nevertheless, the communication overhead between agents in this structure is relatively high. As any agent can make global decisions, the interaction protocols can become intricate, and there can be significant redundancy of interaction information between each satellite agent.

5.2.2 Satellite Agent Model Construction

Based on the above analysis and carefully considering the intelligence and negotiation cost of the distributed satellite cluster, each satellite can be modeled as I_2-level or I_1-level intelligent agent. Then, the top-down and centralized coordination structures are not suitable for distributed task scheduling problem because they contain satellite agents with autonomous capabilities of I_3-level and I_4-level. Conversely, while the fully distributed coordination structure offers high flexibility and reliability, it is less efficient for negotiation and incurs higher costs for collaborative computation. Thus, we adopt the distributed coordination structure as the organizational structure for the multi-satellite multi-agent system. In this section, we introduce a distributed coordination structure multi-agent system with a variable division of labor as the organizational structure for an earth observation satellite cluster, where the division of labor among the satellites is flexible. At any moment, the collaborative division of labor of only one satellite in the satellite cluster is I_1-level satellite agent, and the collaborative division of labor for the remaining satellites is I_2-level satellite agent.

Drawing from the above analysis, we develop a model for a distributed satellite cluster multi-agent system in this section, illustrated in Fig. 5.3 (using a cluster of five satellites as an example). In Fig. 5.3a, each agent represents a satellite in the cluster, and intersatellite links facilitate information exchange between agents. Figure 5.3b depicts the multi-agent system structure of the satellite cluster at a specific moment, where agent 1 can collaborate with the entire cluster, while agents 2 through 5 possess the local scheduling capability of the satellite cluster.

The deliberative agent architecture is utilized to model each satellite in the satellite cluster, as presented in Fig. 5.4. The task planner, which is situated at the core of the satellite agent, is responsible for synthesizing the external information and current state and computing the earth observation meta-tasks that can be executed. Subsequently, it notifies the remaining satellite agents and adjusts its tasks in accordance with the negotiation results obtained from the external satellite agent. The external environment perceiver is responsible for sensing the dynamic changes in the external environment and processing the negotiation information provided by

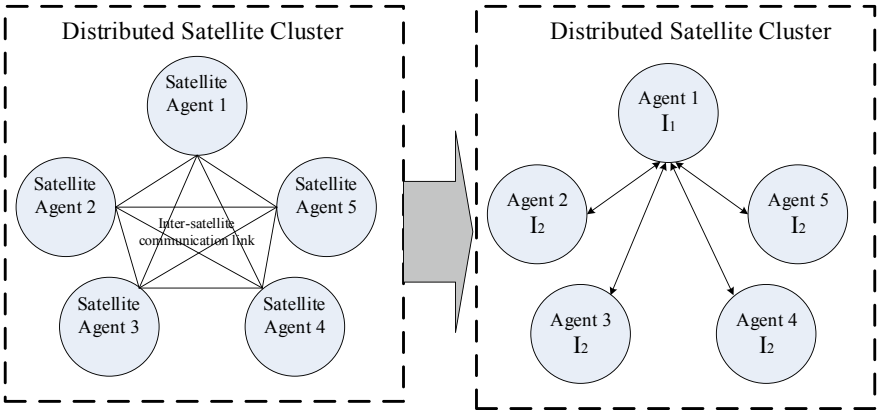


Fig. 5.3 Multi-agent system model for distributed satellite cluster. **a** Schematic diagram of multi-agent system structure. **b** Schematic diagram of satellite agent collaboration division of labor

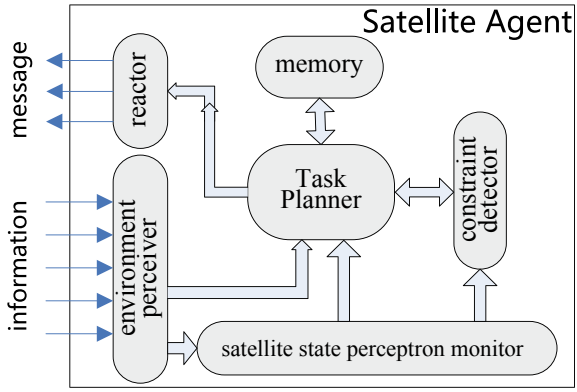


Fig. 5.4 Deliberative agent architecture for each satellite in the distributed satellite cluster

the external satellite agent. The satellite state perceptron monitors the satellite state in real-time and delivers the results to the task planner to facilitate decision-making. The memory component serves as the storage system for the satellite agent, storing the information related to the tasks executed by the satellite and all negotiation results, thus providing data support for the mission planner. Finally, the constraint detector is responsible for identifying the sequence of earth observation meta-tasks developed by the task planner and feeding the task planner with the earth observation meta-tasks that violate the satellite constraints.

5.3 Distributed Satellite Task Scheduling Methods

The interaction and collaboration mechanism among satellite agents is crucial for multi-agent systems to address the distributed task scheduling problem. The contract network protocol and the blackboard model are two widely adopted collaboration mechanisms in multi-agent systems. This section presents the distributed satellite task scheduling techniques based on these two collaboration mechanisms and recommends some enhancement strategies.

5.3.1 Distributed Task Scheduling Based on Contract Network Protocols

1. Contracts network protocol

The contract network protocol is a high-level protocol that facilitates effective agent collaboration by sharing tasks and simulating the contract bidding process [147]. It offers a solution to the task collaboration problem [148].

The protocol states that the contract network consists of some nodes, and each note represents an agent in the multi-agent system. Each node will dynamically assign one of the following three roles.

(1) Manager: The manager generates tasks and assigns them to other nodes. Typically, there is only one manager at a specific time.
(2) Worker: The worker completes the tasks. Usually, there are multiple workers in the system.
(3) Contractor: The contractor is the worker who has been awarded the tender and must carry out the corresponding tasks for which the tender was awarded.

The problem-solving process in the contract network is illustrated in Fig. 5.5.
The problem-solving approach comprises several steps, which are outlined below:

(1) The manager agent initiates a task notification and broadcasts it to the worker agents, as illustrated in Fig. 5.5a.
(2) The worker agent receives the task notification and evaluates whether to bid based on the task requirements, its capabilities, knowledge, and expected benefits. If the worker agent deems itself suitable for the task, it submits several bids to the manager agent as shown in Fig. 5.5b.
(3) The manager agent receives multiple bids and selects one or more agents to complete the task, based on the information provided in the bids. The selected agents become the contractors, as depicted in Fig. 5.5c.

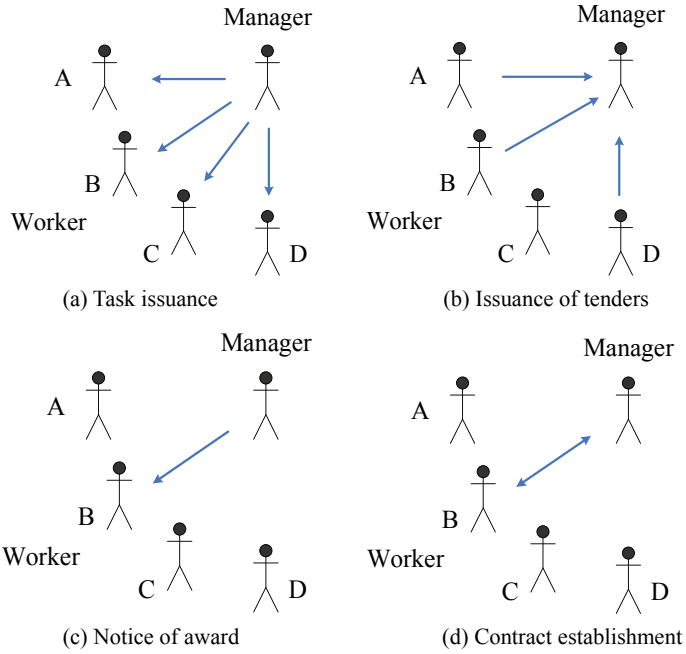


Fig. 5.5 Problem-solving process of the contract network protocol

(4) The contractor completes the task under the supervision of the manager agent. If the contractor fails to complete the task, it is considered a breach of contract and will be penalized accordingly as shown in Fig. 5.5d.

The distributed satellite task scheduling problem involves the selection of a manager agent from a satellite cluster to negotiate and allocate a series of earth observation targets among the remaining satellite agents. According to the traditional contract network protocol, the manager agent can only invite tenders for one observation target at a time, and the bidding order can significantly affect the scheduling result. This limitation poses a challenge to achieving optimized scheduling results and make full use of satellite resources.

To address this challenge, outsourcing and exemption mechanisms [149] have been introduced, inspired by the outsourcing substitution mechanism in business. The outsourcing mechanism enables an agent to outsource a task to other more suitable agents if they cannot complete it. The exemption mechanism is similar to the amnesty principle in tabu search algorithms. If the agent abandons the task already undertaken and accepts a new task, it will not be penalized as long as the overall system benefit increases. Based on these concepts, the Satellite Distributed Scheduling Algorithm Based on Outsourcing Contract Net (SDSOCN) has been developed.

Overall, the proposed algorithm seeks to enhance the performance of contract network-based distributed scheduling in satellite clusters by leveraging outsourcing and exclusion mechanisms. By adopting this approach, it is anticipated that the utilization of satellite resources can be significantly improved, and more optimal scheduling results can be achieved.

Algorithm name: SDSOCN
Input: set of earth observation targets to be assigned (TARGET)
Output: set of earth observation meta-tasks to be performed by the satellite cluster ($\bigcup_{s\in SAT} TASK_{do}^{s}$)

begin
1 randomly select a satellite agent s as the manager agent. $TARGET^{s} = TARGET$
2 while $TARGET^{s} \neq \emptyset$ //$TARGET^{s}$ is the bidding queue of the satellite agent s
3 manager agent removes the first target tar in $TARGET^{s}$, and broadcasts its tender information
4 for each $k \in SAT$
5 satellite agent k calculates the access time window for target tar and generate earth observation meta-tasks
6 satellite agent k initiate single-satellite scheduling calculations, send bids or non-participation messages
7 end for
8 The manager agent analyzes the benefit value and cost of the bidding, then determines the set of winning agents
9 assign the winning satellite agent set to SAT_{win} and send the winning satellite the winning message
10 the manager agent removes the task tar from TARGET set
11 for each $k \in SAT_{win}$ // for each winning satellite Agent
12 if satellite agent k has taken on a task that conflicts with the winning earth observation meta-task, then
13 satellite agent k removes the conflict earth observation meta-task from $TASK_{do}^{s}$
14 satellite agent k adds the target corresponding to the conflict earth observation meta-task to its tender queue $TARGET^{k}$
15 end if
16 satellite agent k inserts the winning earth observation meta-task into the set of earth observation meta-tasks to be executed ($TASK_{do}^{s}$).
17 end for
18 end while
19 The manager agent sends a broadcast to the remaining agents to find the next manager agent.
20 each agent wishing to become the next manager sends an application message to the current manager agent.
21 The manager agent selects the next manager agent based on the importance of the tasks in each applicant agent's queue of targets to be tendered.
22 mark the current manager satellite as agent s, go to 2.
end

In the SDSOCN algorithm, $TARGET^{s}$ in statement 2 is the set of observation targets to be tendered by the manager agent s. In statement 9, SAT_{win} is the set of satellite agents that winning the bid. Each satellite agent first determines the manager by competition. Then, the manager agent and worker agents carry out the normal bidding process based on the contract network protocol to complete the

task assignment work. The main differences between SDSOCN and the traditional contract network protocol are as follows.

- A manager agent is a special kind of worker agent. Therefore, the manager agent is also involved in bidding work.
- If the worker agent's winning earth observation meta-task conflicts with the undertaken earth observation meta-task, the conflicting earth observation meta-task is deleted directly (exemption mechanism). Then, its corresponding observation target is added to the worker agent's target queue to be tendered. When this worker agent becomes a manager agent, it will tender each task in the pending tender target queue (outsourcing mechanism).

2. Multi-agent collaboration processes based on contractual network protocols

The previous section gave the SDSOCN algorithm based on the outsourcing and exemption mechanisms. In this part, the collaboration process between agents is described in detail. For an agent in the multi-agent system model, the roles that may be assumed at a given moment are worker or manager. Based on the division method above, when the agent is a manager, then its collaboration is divided into I_1. When it is a worker, its collaborative division of labor is I_2. The following will describe the collaboration process with other agents when an agent acts as a worker or manager.

(1) Worker agent collaboration process

Figure 5.6 describes the collaboration process between the worker agent and the remaining agents. If the worker agent receives a manager turnover message, it will apply for conversion to the manager agent according to the current target queue to be tendered. If the request is successful, it becomes the next manager agent. If the worker agent receives a bidding message for an observation target, it first calculates the accessibility of the target to be observed. If the agent has no access time window for the tendered observation target (the observation task set is empty), a non-participation message is sent to the manager agent. Otherwise, the cost for executing the observation target is calculated, and a bid is submitted to the manager agent. The bid generation (BG) algorithm is as follows.

Algorithm name: BG
Input: earth observation target (tar)
Output: Bid information (Bid) to the earth observation target (tar)

(continued)

(continued)

begin
1 receiving a bid message from the manager agent for the target tar
2 perform accessibility calculations on the target (tar) to generate a collection of earth observation meta-tasks TASK_{tar}
3 if $\text{TASK}_{\text{tar}} = \phi$ then
4 sending a non-participation message to the manager agent
5 exit
6 end if
7 for each task $\in \text{TASK}_{\text{tar}}$
8 setting the evaluation value of the earth observation meta-task (task) as the bid gain
9 calculation the set of earth observation meta-tasks that conflicted with task ($\text{ConflictSet}_{\text{task}}$)
10 sum the evaluation values of each task in $\text{ConflictSet}_{\text{task}}$ as the bidding price
11 end for
12 select the least costly earth observation meta-task (task_{min}) from TASK_{tar}
13 using task_{min} to generate the bid information (Bid) and send it to the manager agent
end

Statement 9 in the BG algorithm obtains the assumed tasks in the work agent that conflict with the current tender earth observation meta-task by computing them and forming the set of conflicting tasks $\text{ConflictSet}_{\text{task}}$.

(2) Manager agent collaboration process

In task collaboration, the manager agent assumes a leading role in cooperation with the worker agents. The manager agent participates in the bidding process for earth observation targets in the target queue, following specific rules for bid evaluation and task assignment. Once the bidding process is completed, a manager replacement message is sent to the other worker agents to confirm the next manager agent under the rules. At this point, the current manager agent transitions into the work process of a worker agent. The collaboration process of the manager agent is illustrated in Fig. 5.7.

(3) Bid evaluation strategy for manager agent

After receiving the bid information, the manager agent will analyze the bid information and select one or more satellite agents that are most suitable to undertake the bidding task. The manager agent evaluates each bid using a rule-based reasoning expert system (e.g., CLIPS-based expert decision support system [126]). The following are the bid evaluation rules we mainly use.

(1) If the task has not been completed, the winning bid will be arranged if the benefit of task execution exceeds the cost.
(2) Preference shall be given to the bidder whose bid is the least costly.
(3) The bidder with the lowest load shall be selected in case of equal bidding cost.

Rule (1) guarantees that the satellite cluster will do its best to accomplish the earth observation task, reflecting the exemption principle described above. Rule (2)

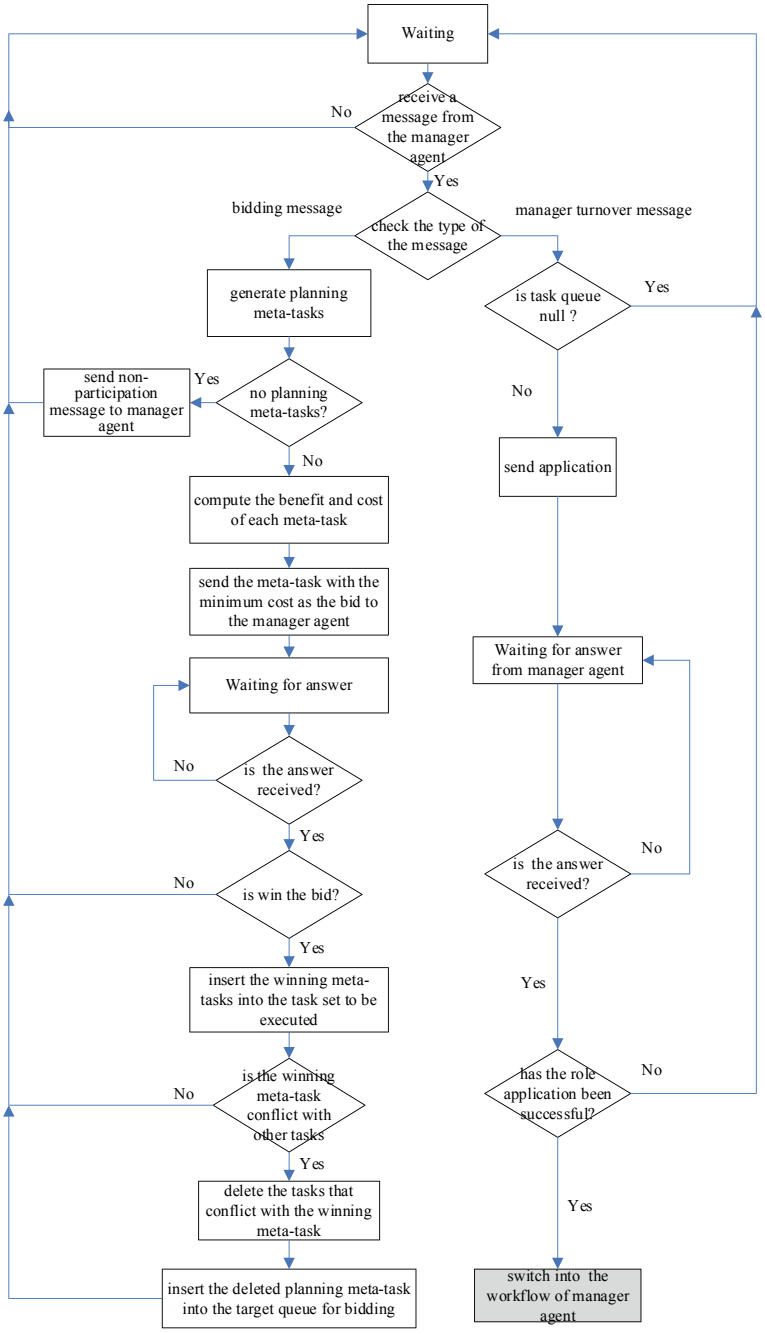


Fig. 5.6 Worker agent collaboration process

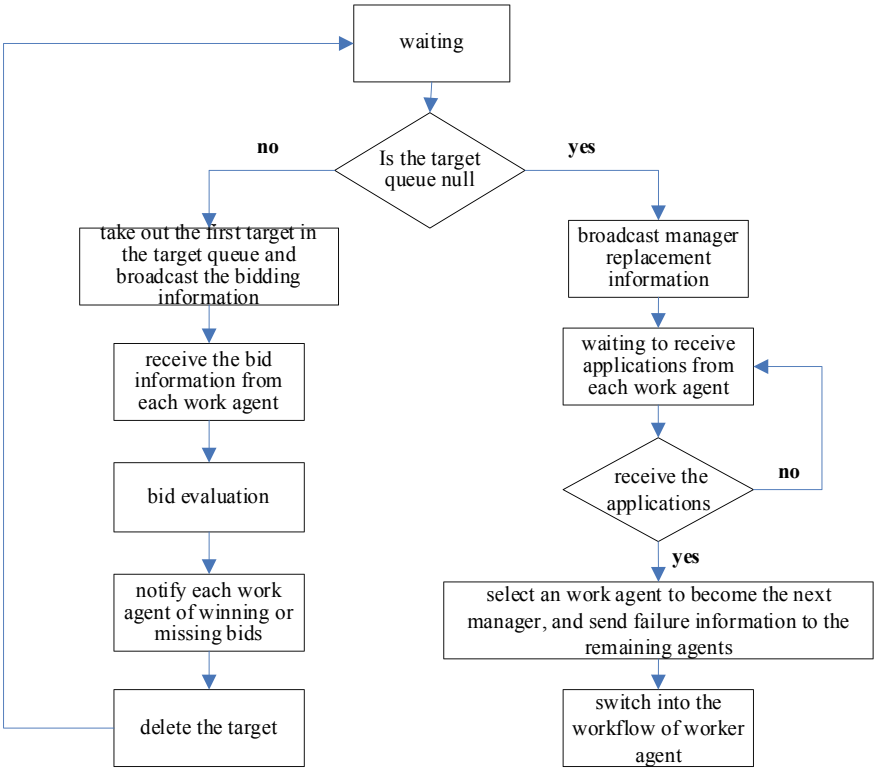


Fig. 5.7 Manager agent collaboration process

ensures the utilization of satellite resources. Rule (3) guarantees the balanced load of the satellites in the satellite cluster.

3. Further exploration

Here, we will analyze the shortcomings of the contract network protocol and propose some improvement strategies.

(1) Improved bidding strategies based on the horse-trading effect [135]

In the framework of contract network computing, the benefits of individual earth observation meta-tasks in an earth observation task are allocated in advance. In some specific application scenarios, the benefit obtained from partial completion of an earth observation task may not be proportional to the benefit obtained from full completion. For example, the benefit obtained from a task completed by 90% may be only 70% or even less. If there are multiple tasks, the user always expects to be able to meet some of them first, and the remaining resources are redistributed to other tasks to improve the completion of the tasks. Thus, a three-stage market negotiation

mechanism based on the Matthew effect (the rich get richer and the poor get poorer) is designed to solve this problem [150, 151]. The followings are its basic idea.

(a) During the free negotiation phase, a predetermined amount of the reward for each earth observation task is withheld as an incentive for completion. This portion of the reward cannot be allocated by the manager agent and can only be obtained upon the successful completion of the task. The worker agents engage in iterative negotiations to optimize the overall system gain until a Nash equilibrium is achieved. Once the maximum number of free negotiation iterations has been reached, the system enters the monopolistic competition phase.
(b) Monopolistic competition phase: At this stage, the earth observation tasks with a low degree of completion withdraw from the competition and release the occupied resources. The tasks with a high degree of completion can prioritize these resources to meet their requirements fully. After this process, it will enter the reshuffle stage.
(c) Reshuffling phase: In this phase, earth observation tasks that have withdrawn from competition rejoin the consultation environment. However, if a task requirement has been fully met, the satellite agent cannot easily occupy its resources in the negotiation process.

The objective of the free negotiation phase is to establish a just negotiation environment and enhance the algorithm's capacity for global search. In contrast, the monopolistic competition phase leverages the Matthew effect to promote task execution integrity and expedite the algorithm's convergence rate. The final reshuffle phase seeks to strike a balance between task execution integrity and overall system benefits, thereby optimizing the system globally. This phase represents a crucial step toward achieving fine-grained optimization.

(2) Improved bidding strategies based on clustering [68]

In the framework of contract network calculation, each satellite agent selects some bids according to the set criteria, such as the maximum priority strategy, the minimum remaining scheduling time strategy, the maximum task satisfaction strategy, etc. The core of these criteria is calculating the evaluation value of bids based on some evaluation function and selecting the one with the higher benefit value [152]. The advantage of these strategies is simple, intuitive, and efficient. Still, the disadvantage is that the difference and diversity of bid selection are often insignificant, and it is easy to fall into local optimization. The top-ranked bids may differ only in a few earth observation meta-tasks for a given satellite agent. If the satellite agents aim to maximize their interests, their top-ranked bids may prioritize the observation activities of high-priority tasks. It will make the low-priority tasks challenging to plan because many resources are already occupied by high-priority tasks, which also leads to the loss of bid diversity and may cause the scheduling process to fall into local optimal solutions. In contrast, the clustering algorithm can divide the bids into several groups based on similarity, with some variability between each group.

Therefore, all possible scenarios of observation activities within the agent can be clustered, and representatives from each class can be selected as bids for the tender.

(3) Improved bid evaluation strategies based on intelligent optimization [153]

Under the framework of contract network calculation, the manager agent usually adopts accurate search algorithms such as backtracking, branch and bound method to evaluate bids. The computation time is not long when the number of agents is small, and the number of bids is small so that it can meet the requirements. However, the computation time of manager agent evaluation bids grows exponentially with the increase in the number of agents and bids of each satellite, making it challenging to meet the requirements of fast dynamic response for online scheduling. The intelligent optimization algorithms simulate the evolutionary process of organisms in nature, which has good optimality and high solution efficiencies, such as genetic algorithm and particle swarm optimization algorithm. So these intelligent optimization algorithms can be introduced into the bid evaluation process of the contract network calculation to search the optimal combination of bids for each satellite agent. Taking the genetic algorithm as an example, the manager agent constructs chromosomes by integer coding for each satellite agent's bids and then selects one or no bids from each satellite agent's bids according to the selection, crossover, and mutation evolution process.

5.3.2 Distributed Task Scheduling Based on Blackboard and Evolutionary Computation

1. Blackboard model

The concept of the blackboard was first introduced by Newell in 1962. In the early 1970s, Carnegie-Mellon University proposed a blackboard model for problem-solving and developed the HEARSAY-II speech understanding system, the first expert system based on the blackboard model. The blackboard model relies on multiple human or subject experts collaborating to solve a problem, using a shared workspace known as the blackboard. The problem-solving process begins when the problem and initial data are entered onto the blackboard. Experts share information through the blackboard and utilize their specialized knowledge to solve the problem. When an expert determines that there is enough information on the blackboard to support further problem-solving, they record their results on the blackboard. This newly added information allows other experts to continue their work, repeating the process until the problem is solved and the final results are obtained. The working process of the blackboard model is shown in Fig. 5.8.

The following are the three essential components of the blackboard model.

(1) Knowledge Sources (KS). The application domain is divided into some independent experts according to their expertise in solving the problem, and these experts are called knowledge sources (i.e., subjects).

(2) Blackboard. A shared problem-solving workspace. It is generally organized in a hierarchical structure. It mainly stores the information the knowledge source requires, the solution data during the problem-solving process, and sometimes the control data. During the problem-solving process, the knowledge sources continuously modify the blackboard. Communication and interaction between knowledge sources take place through the blackboard.
(3) Monitoring Mechanism. According to the problem-solving status on the blackboard and the solving skills of each knowledge source, the appropriate knowledge sources are dynamically selected and activated based on some control strategy so that the knowledge sources can respond to the blackboard changes on time.

The following are the key features of the blackboard model for implementing distributed collaborative problem-solving [154, 155].

- The subjects (i.e., knowledge sources) are independent, and there are no interactions between subjects.
- The blackboard structures enable flexible representation of information.
- Use of a common language of interaction.
- An independent monitoring mechanism.
- Blackboard structures are suitable for describing and processing problems at multiple levels of abstraction.
- Opportunity problem-solving mechanisms, especially for complex problems where the order of problem-solving cannot be determined in advance.

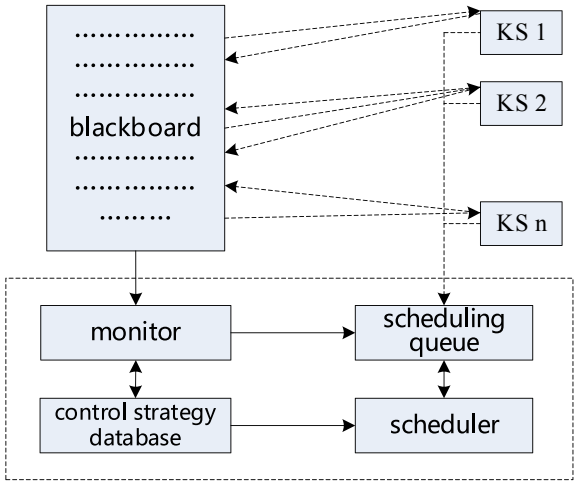


Fig. 5.8 Working process of the blackboard model

- The blackboard model provides a way to integrate existing software.

2. Agent collaboration process based on blackboard model and evolutionary computation

To solve the distributed satellite task scheduling problem, this approach combines the blackboard model with an intelligent optimization algorithm. Following the multi-agent system architecture, each satellite is treated as an agent, with the manager agent serving as the blackboard's function. Each satellite agent schedules its tasks independently and communicates with the blackboard instead of directly with other agents. During each evolutionary calculation, each satellite agent retrieves the optimal solutions of other agents from the blackboard and employs them to guide the evaluation, selection, and restoration of the population individuals in its evolutionary computation.

The Distributed Satellite Task Scheduling algorithm based on Blackboard Model and Genetic Algorithm (DSTS-BMGA) is designed using the genetic algorithm as an example. Each satellite agent initializes its subpopulation based on the earth observation meta-task information received from the manager agent. Each individual in the subpopulation represents the satellite agent's current single-satellite earth observation programme. During each evolutionary calculation, the satellite agent performs crossover, variation, and constraint adjustment operations on its subpopulation and interacts with the manager agent to retrieve the individual representatives sent to the center by the other satellite agents. The satellite agent then calculates the adaptation value of each individual in the subpopulation and sends the optimal individual to the manager agent. This process is repeated, with the manager agent searching for a more optimal observation solution from each satellite agent until a satisfactory solution is found to the problem. The algorithm's steps are as follows:

Algorithm name: DSTS-BMGA
Input: the set of earth observation meta-tasks TASK
Output: the set of earth observation meta-tasks performed by each satellite $TASK_{do}$

(continued)

(continued)

begin
1 the manager agent gets the set of earth observation meta-tasks to be scheduled (TASK)
2 for each $k \in$ SAT
3 manager agent send the TASK to satellite agent k
4 satellite agent k random initialization of subpopulations
5 satellite agent k select the best individuals in the subpopulation to send to the blackboard
6 end for
7 for each $k \in$ SAT
8 satellite agent k do crossover and mutation operations on subpopulations to generate next-generation populations
9 receive the other satellite agent's individuals (solution) from the blackboard and calculate the adaptation values for each individual of each next-generation population
10 elite archiving
11 send the optimal individual (solution) to the blackboard
12 end for
13 manager agent determines if the algorithm has reached the convergence condition, terminate the algorithm if yes, otherwise go to 7.
14 each satellite agent decodes and sends the assumed tasks to the manager agent to generate the final solution (TASK_{do})
end

Where the convergence condition of the algorithm can be designed as follows.

(1) The locally optimal solutions of each satellite agent are identical and have undergone a certain number of generations of iterations, the exact number of generations being obtained empirically.
(2) Algorithm iterations exceed the maximum number of iterations for scheduling.
(3) Algorithm runtime exceeds the maximum runtime.

The workflow of the DSTS-BMGA algorithm is shown in Fig. 5.9.

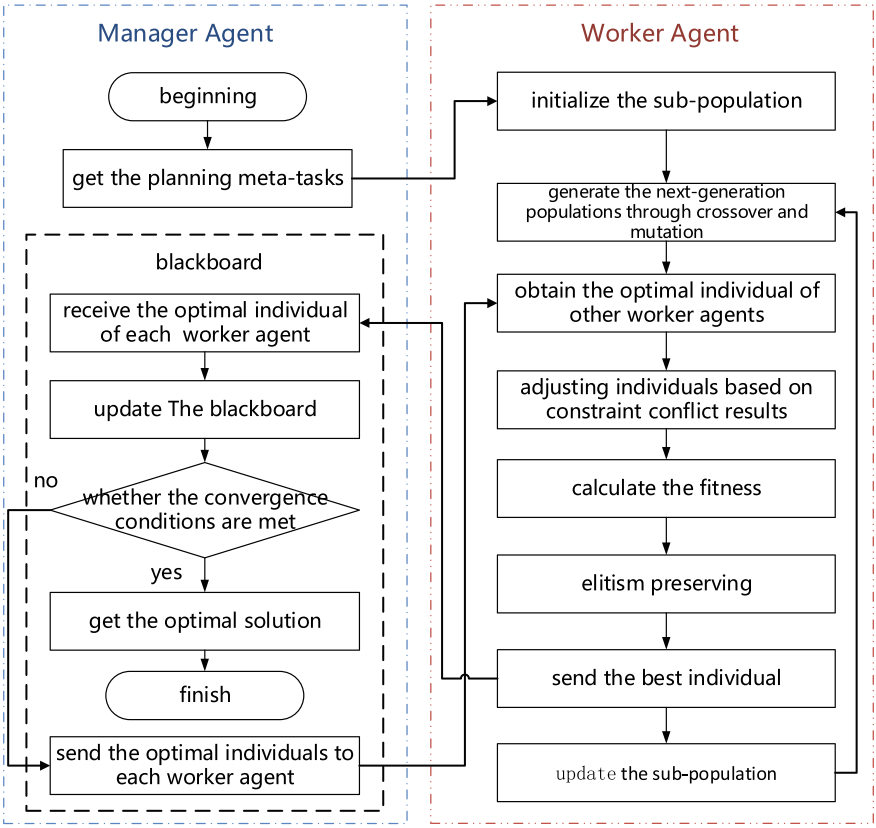


Fig. 5.9 Distributed satellite task scheduling algorithm flow based on blackboard model and genetic algorithm

Chapter 6
Satellite Onboard Autonomous Task Scheduling Models and Methods

The scheduling of satellite tasks is generally performed on the ground. The EOSOC generates a daily executable programme an for each satellite, which is then translated into instructions and transmitted to the satellite via the TT&C equipment. The satellite follows these instructions without any modifications. However, recent advancements in artificial intelligence and space science technology have led to the emergence of a new generation of satellites with onboard data processing and autonomous task scheduling capabilities. In this book, these satellites are referred to as autonomous operation satellites. An autonomous operation satellite can carry out routine operations without human intervention, monitoring the system, detecting faults, and performing self-adaptation, self-regulation, self-tolerance, and self-recovery functions in the event of faults or unknown environments. Such a satellite is no longer a passive executor, but rather a decision-maker that can actively generate or modify observation programme based on changes in the space environment, equipment status, and available resources. Its capability to respond quickly to uncertain space environments and unexpected satellite states surpasses that of traditional satellites. In the future, autonomous operation satellites will become one of the primary deployment forms of earth observation satellites [72].

The limited computing power of the onboard processor is a significant weakness of the autonomous operation satellite. This processor is primarily an embedded processor that must consider the impact of space radiation, energy, temperature, weight, and other factors. As a result, it operates at a low frequency, has limited memory, and relatively weak computing power compared to the personal desktop computer or server typically used in the ground system. Furthermore, the onboard processor is responsible for operation environment awareness, state monitoring, attitude adjustment and control, onboard fault tolerance, and self-recovery, among other tasks, which further stretch the scarce onboard computing resources. Due to the limitation of the onboard processor's computing power, the task scheduling algorithms used in ground computing devices cannot be applied onboard. Therefore, new methods and mechanisms must be investigated to adapt to these changes. This

H. Chen et al., *Earth Observation Satellites*,
https://doi.org/10.1007/978-981-99-3565-9_6

chapter introduces some mainstream satellite onboard autonomous task scheduling models and techniques.

6.1 Satellite Onboard Autonomous Task Scheduling Problem

Satellite onboard autonomous task scheduling refers to the process by which an earth observation satellite automatically develops an onboard programme with minimal reliance on ground personnel intervention. This process is based on received tasks, equipment status, energy, and storage resources. Satellites capable of onboard autonomous task scheduling are more intelligent than traditional ground-based task scheduling methods. They no longer act as simple command execution terminals but rather as task decision-makers. The technology of satellite onboard autonomous task scheduling offers several advantages.

(1) Autonomous satellite onboard task scheduling can enhance the rapid response capabilities of satellites to emergencies. The satellite can detect anomalies on the ground through onboard data processing or receive observation requests from the ground sensor network or users and adjust its onboard observation programme accordingly [156]. For example, it can quickly adjust the onboard observation sensor to observe unexpected events such as volcanic eruptions, earthquakes, tsunamis, and military conflicts in sensitive areas.
(2) The technology can also improve the utilization of satellite resources. Optical imaging satellites rely on optical sensors to scan ground targets, which are often obstructed by clouds. As a result, around 80% of the ground targets of optical imaging satellites fail to be observed due to cloud cover obstruction [157]. However, autonomous operation satellites can adjust their observation programme based on the cloud cover information uploaded by the ground station or obtained from onboard analysis results [77]. By doing so, the satellites can use their resources more efficiently to observe high-value targets and produce high-quality remote sensing images.
(3) The technology can improve the granularity and flexibility of satellite task execution. Ground-based satellite task scheduling systems are subject to stringent constraints to ensure operational safety, which can limit satellite capabilities. For instance, energy constraints are usually based on cumulative working time and cannot accurately calculate the energy state onboard in real-time. Autonomous operation satellites can create more detailed programme based on real-time resource and equipment status and can modify the observation programme according to changes in resources and tasks, increasing the flexibility of satellite task execution.
(4) Finally, autonomous task scheduling on satellites can reduce ground operators' workload and staffing costs. By transferring part of the work undertaken by the ground control center to the satellite platform, the technology reduces staff

involvement. It decreases the workload and staffing costs of maintaining and managing satellite operations.

In the following section, we will describe the satellite onboard autonomous task scheduling process, and analyze its difficulties and characteristics.

6.1.1 Process of Satellite Onboard Autonomous Task Scheduling

The description of the satellite onboard autonomous task scheduling problem is illustrated in Fig. 6.1. The autonomous operation satellite orbits the earth and calculates the access time window for each ground target. It then generates corresponding earth observation meta-tasks within the autonomous scheduling horizon. The earth observation meta-task with the earliest access start time within the autonomous scheduling horizon is referred to as the critical task, and the critical scheduling duration is the time between the access start time of the critical task and the current onboard system time. The satellite onboard autonomous tasking process needs to calculate the scheduling result of each earth observation meta-task in the autonomous scheduling horizon within the critical scheduling duration. As time passes, new earth observation meta-tasks continuously come into the autonomous scheduling horizon. The satellite decides whether to execute each earth observation meta-task within the autonomous scheduling horizon based on onboard energy and storage resources, earth observation meta-task distribution information, and other factors. The first decision that must be made is whether the critical task should be executed. Suppose the task scheduling algorithm has not calculated whether the critical task should be executed while the onboard system time reaches the access start time of the critical task. In that case, the current scheduling is deemed failed, and the satellite misses the observation opportunity. The goal of satellite onboard autonomous task scheduling is to maximize user requirements while adhering to satellite constraints.

The autonomous task scheduling process onboard satellites can establish a more detailed mathematical model compared to ground-based task scheduling. This is

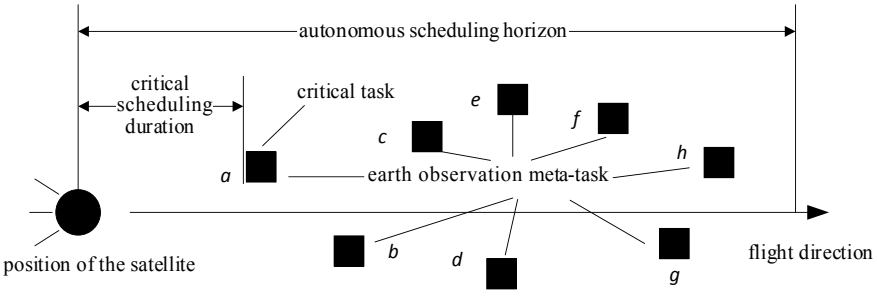


Fig. 6.1 Description of the satellite onboard autonomous task scheduling problem

because the satellite can obtain real-time information about energy, storage, payload, and other resources and equipment status. However, satellite task scheduling is a complex problem that involves numerous resources, diverse application requirements, and complex constraints. Typically, the satellite task scheduling problem consists of two subproblems: observation task scheduling and data transmission task scheduling. In this chapter, we focus only on the solution for the observation task scheduling problem, assuming that the data transmission resources used by the autonomous operation satellite have already been determined either by the ground or onboard.

6.1.2 Challenges of Satellite Onboard Autonomous Task Scheduling

1. Uncertainty in earth observation demands

The uncertainty in earth observation demands refers to the dynamic changes in observation demand during satellite operations, as described in Sect. 4.1.1.

2. Limited computing resources for task scheduling

Earth observation satellites operate in the near-earth orbit and are susceptible to cosmic radiation and temperature differences, which put severe demands on onboard processors [158]. Therefore, aerospace-grade CPUs with low computing power are used to ensure reliability and stability. For instance, the RAD 6000 processor used by NASA on the Deep Space 1 satellite has a main frequency of 25 MHz and a computing power of no more than 100 Million Instructions Per Second (MIPS) [74]. The latest domestic Longxin 1E300 processor and foreign RAD750 processor have a main frequency of about 200 MHz and a computing power of approximately 240–400 MIPS [158, 159]. The main frequency of the latest domestic Longxin 1E300 processor and foreign RAD750 processor is about 200 MHz, and the computing power is about 240 ~ 400 MIPS. The limited computational resources pose a significant challenge to onboard autonomous task scheduling.

3. Timeliness of task scheduling

Unlike the offline mode of "scheduling before execution" on the ground, the online mode of "scheduling while executing" on the satellite requires higher computational timeliness for the task scheduling algorithm. The algorithm must provide a practical and feasible programme before the execution of an observation task to avoid missing observation opportunities and reducing the satellite resource utilization.

4. Uncertainty in the successive implementation of subsequent tasks

Several factors, such as energy, storage, satellite attitude, and target distribution, influence the execution of observation tasks during an autonomous operational satellite's flight. Figure 6.2a shows that the meta-task c cannot be executed in observation programme P1 due to insufficient energy after performing several meta-tasks.

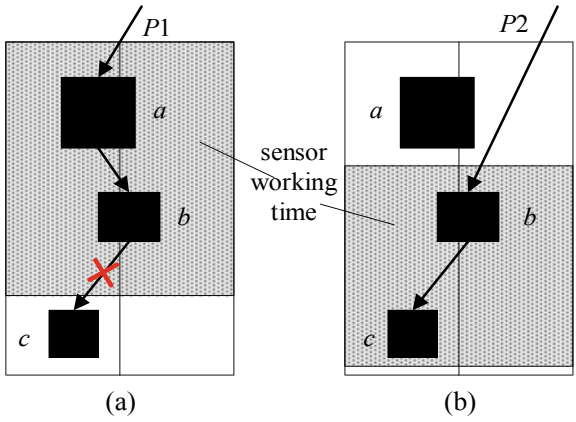


Fig. 6.2 Schematic diagram of the uncertainty of continuous mission execution during the onboard autonomous task scheduling process

However, in Fig. 6.2b, the meta-tasks b and c can execute in observation programme P2, which demonstrates the uncertainty in the successful execution of subsequent tasks during the task scheduling process.

6.2 Path Searching Approach for Satellite Onboard Autonomous Task Scheduling

6.2.1 Rolling Optimization Strategy

The rolling optimization strategy (Fig. 1.3) is a general approach for onboard autonomous task scheduling. As outlined in Sect. 1.1.4, the onboard planner schedules earth observation meta-tasks dynamically within the autonomous scheduling horizon, represented as $[t_{\text{sys}} + \text{Tp}, t_{\text{sys}} + \text{Th}]$ ($\text{Tp} \ll \text{Th}$), where t_{sys} denotes the onboard system time, Tp indicates the preparation time for task execution, and Th signifies the autonomous scheduling horizon's duration. The autonomous scheduling horizon constantly shifts as the autonomous operation satellite proceeds, with new earth observation meta-tasks entering and existing ones departing due to execution, expiration, or cancelation. This necessitates the onboard planner's continuous scheduling of future tasks.

Within the autonomous scheduling horizon, two triggers are established. Any dynamic changes to the earth observation meta-tasks will prompt the onboard planner to replan and revise the earth observation programme. Figure 6.3 depicts the onboard autonomous task scheduling process under the rolling scheduling strategy.

(1) Earth observation meta-tasks enter the autonomous scheduling horizon at the system moment t_{sys}. This scenario can be further divided into two: first, a known earth observation meta-task outside the autonomous scheduling horizon enters; and second, a user requests a new observation target, resulting in the system

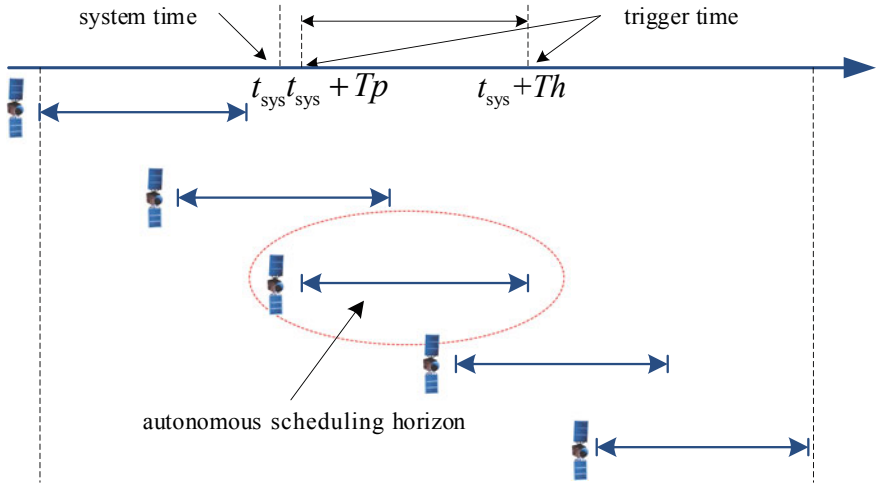


Fig. 6.3 Schematic diagram of rolling task scheduling

generating a new earth observation meta-task with an access time within the autonomous scheduling horizon.

(2) The earth observation meta-task is removed from the autonomous scheduling horizon at system time. This scenario can be categorized into two: first, a canceled earth observation meta-task is removed from the earth observation programme; and second, a earth observation meta-task with an access start time equal to the current system time is executed if included in the observation programme, otherwise, it is disregarded.

In the following step, we propose to integrate the rolling optimization approach with a graph model to schedule the earth observation meta-task within the autonomous scheduling horizon effectively. This integration will be achieved by utilizing a path search algorithm, which will aid in optimizing the scheduling strategy.

6.2.2 Directed Acyclic Graph Model

To begin, we will organize all earth observation meta-tasks within the autonomous scheduling horizon in ascending order based on their access start times. Subsequently, we will employ a directed acyclic graph $G = (V, E)$ (as depicted in Fig. 6.4) to model the satellite observation task scheduling problem, where V, E represents the set of vertices and the set of directed edges in the graph G, respectively. For any given vertex $v_i \in V$ corresponding to the earth observation meta-task task_i, we can describe it as and can be described as $(t_b^i, t_e^i, \text{mod}_i, \psi_i, \text{eng}_i, \text{mem}_i)$, where t_b^i, t_e^i, mod_i, ψ_i are, respectively, denotes the access start time, access end time, work mode, and priority of the task_i, and eng_i, mem_i represents the energy and storage resources required to

execute the task$_i$. Each directed edge $(i, j) \in E$ in the graph indicates a connection between two vertices v_i and v_j, that is, the earth observation meta-task task$_i$ and task$_j$ satisfy the work mode switching constraint that $t_{\mathrm{b}}^{j} - t_{\mathrm{e}}^{i} \geq \mathrm{trans}_{\mathrm{mod}_i,\mathrm{mod}_j}$. $\Delta\mathrm{eng}_{ij}$ is the weights of directed edge (i, j), it denotes the maximum energy that can be replenished from the access end time of earth observation meta-task task$_i$ to access start time of earth observation meta-task task$_j$. To facilitate problem-solving, we will add two virtual vertices to the directed acyclic graph model: the start vertex v_{s} and the end vertex v_{t}. We will then map the satellite earth observation task scheduling problem as a path search problem, with the objective of identifying a path from the start vertex v_{s} to the end vertex v_{t} that maximizes the evaluation value while satisfying the energy and storage constraints.

The directed acyclic graph model utilizes several relevant symbols and decision variables, which are defined as follows.

- $x_{ij} \in \{0, 1\}$: the decision variable, the $x_{ij} = 1$ indicates that a directed edge (i, j) is selected, otherwise it is not, $0 \leq i < j \leq n - 1$ (n denotes the number of vertices in graph G, which contains the start vertice and end vertice).
- $y_{ij} \geq 0$: the decision variable, which represents the total amount of energy spilled while charging from the access end time of task$_i$ to the access start time of task$_j$, $0 \leq i < j \leq n - 1$.
- $z_{ij} \geq 0$: the decision variable, which represents the total amount of data overflow from the access start time to the access end time of the data transmission task task$_j$, $0 \leq i < j \leq n - 1$.

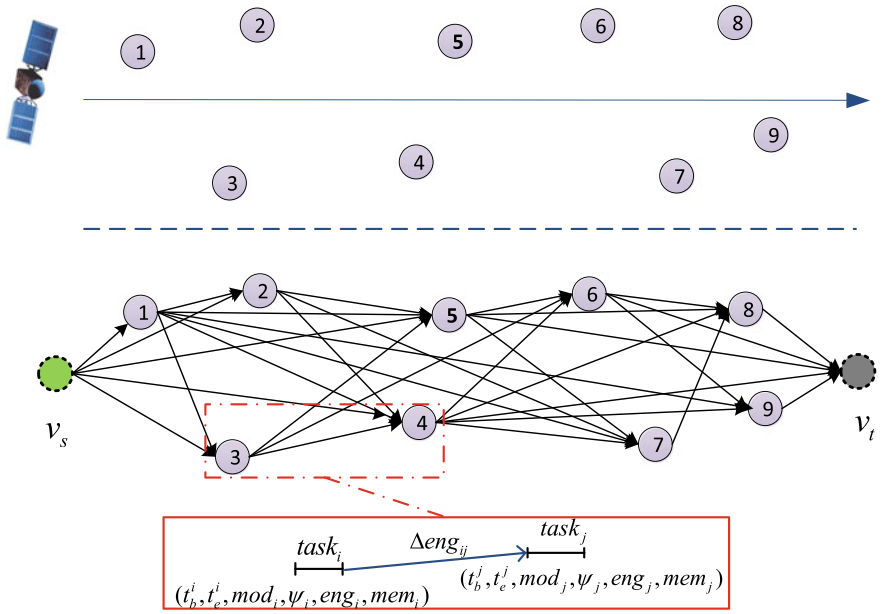


Fig. 6.4 Directed acyclic diagram model for earth observation task scheduling

- eng_0: the energy state at the initial moment.
- mem_0: the memory state at the initial moment.
- Engy: the upper bound of onboard battery capacity onboard.
- Memy: the upper bound of the storage space.

Given the problem description and the mathematical notation and decision variables provided above, we can now construct a mixed integer programming model for the problem. The model is as follows.

Optimization objectives.

$$V_{\text{imp}} = \max\left\{\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}\psi_j \cdot x_{ij}\right\}. \tag{6.1}$$

Constraints.

$$\sum_{j=1}^{n-1} x_{0j} = 1 \tag{6.2}$$

$$\sum_{i=0}^{n-2} x_{i(n-1)} = 1 \tag{6.3}$$

$$\sum_{i=0}^{j-1} x_{ij} \le 1,\ 1 \le j \le n-1 \tag{6.4}$$

$$\sum_{i=0}^{j-1} x_{ij} = \sum_{k=j+1}^{n-1} x_{jk},\ 1 \le j \le n-2 \tag{6.5}$$

$$\begin{aligned}&\text{eng}_0 + \sum_{j=1}^{k-1}\sum_{i=0}^{j-1}\left(\left(\Delta\text{eng}_{ij} - \text{eng}_i\right) * x_{ij} - y_{ij}\right)\\ &\quad + \sum_{i=0}^{k-1}\left(\Delta\text{eng}_{ik} * x_{ik} - y_{ik}\right) \le \text{Engy},\ 1 \le k \le n-1\end{aligned} \tag{6.6}$$

$$\begin{aligned}&\text{eng}_0 + \sum_{j=1}^{k-1}\sum_{i=0}^{j-1}\left(\left(\Delta\text{eng}_{ij} - \text{eng}_i\right) * x_{ij} - y_{ij}\right)\\ &\quad + \sum_{i=0}^{k-1}\left(\left(\Delta\text{eng}_{ik} - \text{eng}_k\right) * x_{ik} - y_{ik}\right) \ge 0,\ 1 \le k \le n-1\end{aligned} \tag{6.7}$$

$$\text{mem}_0 + \sum_{j=1}^{k-1}\sum_{i=0}^{j-1}\left(\text{mem}_j * x_{ij} + z_{ij}\right)$$

$$+\sum_{i=0}^{k-1}\mathrm{mem}_k * x_{ik} \le \mathrm{Memy},\ 1 \le k \le n-1 \tag{6.8}$$

$$\mathrm{mem}_0 + \sum_{j=1}^{k-1}\sum_{i=0}^{j-1}\left(\mathrm{mem}_j * x_{ij} + z_{ij}\right) + \sum_{i=0}^{k-1}\left(\mathrm{mem}_k * x_{ik} + z_{ik}\right) \ge 0,\ 1 \le k \le n-1 \tag{6.9}$$

$$y_{ik} = x_{ik} * \max\left\{0, \mathrm{eng}_0 + \sum_{j=1}^{k-1}\sum_{i=0}^{j-1}\left(\left(\Delta\mathrm{eng}_{ij} - \mathrm{eng}_i\right) * x_{ij} - y_{ij}\right) + \sum_{i=0}^{k-1}\Delta\mathrm{eng}_{ik} * x_{ik} - \mathrm{Engy}\right\} \tag{6.10}$$

$$z_{ik} = x_{ik} * \max\left\{0, -\left(\mathrm{mem}_0 + \sum_{j=1}^{k-1}\sum_{i=0}^{j-1}\left(\mathrm{mem}_j * x_{ij} + z_{ij}\right) + \sum_{i=0}^{k-1}\mathrm{mem}_k * x_{ik}\right)\right\}. \tag{6.11}$$

The mathematical model presented above comprises an optimization function and several constraints. The optimization function, defined as Eq. (6.1), aims to maximize the cumulative priority of the earth observation meta-tasks executed by the autonomous operation satellite. Constraint (6.2) mandates that the out-degree of the starting vertex v_s must be equal to 1, meaning that only one task can be executed after the starting vertex v_s. Similarly, constraint (6.3) requires that the entry degree of the end vertex, denoted by v_t must also be 1, indicating that only one task can point to the end vertex v_t. Constraints (6.4) and (6.5) ensure that the in-degree of any intermediate vertex must be the same as the out-degree and less than or equal to 1, indicating that each task has only one predecessor and one successor task. Together, these constraints guarantee the uniqueness of the solution. Constraint (6.6) stipulates that the energy state at the access start time of a task must not exceed the upper bound of the battery capacity. In this equation, the summation of the first two terms ($\mathrm{eng}_0 + \sum_{j=1}^{k-1}\sum_{i=0}^{j-1}((\Delta\mathrm{eng}_{ij} - \mathrm{eng}_i) * x_{ij} - y_{ij})$) denotes the energy state after the previous mission of task_k is executed, while the third term ($\sum_{i=0}^{k-1}(\Delta\mathrm{eng}_{ik} * x_{ik} - y_{ik})$) represents the amount of energy changed between the access end time of the previous mission of task_k and the access start time of task_k. Similarly, constraint (6.7) specifies that the energy state at the access end time of a mission cannot be lower than the lower bound of the energy state, where the third part of the equation ($\sum_{i=1}^{k-1}((\Delta\mathrm{eng}_{ik} - \mathrm{eng}_k) * x_{ik} - y_{ik})$) represents the amount of energy changed from the end of the previous task execution to the end of the current task execution. Finally, constraints (6.8) and (6.9) mandate that the storage state at the end of any task execution cannot exceed the upper storage bound or fall below

the lower storage bound, respectively. The energy level of a satellite can be restored during idle periods. However, updating the energy level by utilizing the maximum charged energy may result in surpassing the battery's capacity, thus necessitating a correction. The amount of correction needed when the energy is restored can be determined using the constraint (6.10). Similarly, when updating the storage state with the maximum quantity of data that can be downlinked, the storage state may turn negative and therefore requires correction. The constraint (6.11) provides the necessary correction amount for the storage state during data downlinking.

6.2.3 Exact Search Algorithm Based on Path Label Updating

Utilizing the directed acyclic graph model presented previously, we employ the label updating algorithm [160] to tackle the path search issue in the graph. Initially, we provide definitions and theorems that will be utilized throughout the solution process.

Definition 6.1: Path dominance Let $P1, P2 \in \mathrm{PATH}_{s,i}$, if it satisfies $\psi(P1) < \psi(P2)$, $E(P1) \geq E(P2)$, $M(P1) \geq M(P2)$, then the path is said to be $P2$ dominates $P1$, denoted as $P1 \triangleleft P2$, where $\mathrm{PATH}_{s,i}$ is the set of all paths from the vertex v_s to the vertex v_i. $\psi(P)$, $E(P)$, and $M(P)$ refer to the evaluation metrics, which consist of the cumulative sum of task priorities, energy consumption, and storage consumption for the earth observation programme represented by the given path P.

Definition 6.2: Non-dominated and dominated paths Let $P \in \mathrm{PATH}_{s,i}$, if $\nexists Q \in \mathrm{PATH}_{s,i}$, such that $P \triangleleft Q$, then the path P is a non-dominated path. Otherwise P is a dominated path.

Definition 6.3: Optimized path Let $P \in \mathrm{PATH}_{s,i}$, if P is a non-dominated path, then we call P is an optimized path.

Lemma 6.1 *In the loop-free directed graph model, the subpaths of a non-dominated path are non-dominated paths.*

Proof We can demonstrate this assertion via a proof by contradiction. Let the path P be a non-dominated path consisting of $v_s \rightarrow v_i \rightarrow v_j$. Then the path $P1 \in \mathrm{PATH}_{s,i}$ is a subpath of P and $\exists P2 \in \mathrm{PATH}_{s,i}$, such that $P1 \triangleleft P2$. Let $P3 \in \mathrm{PATH}_{i,j}$ be the subpath of $P \backslash P1$, and since $P1 \triangleleft P2$, then we have $E(P1) \geq E(P2)$ and $M(P1) \geq M(P2)$. If we combine the path $P2$ with path $P3$ to form a new path Q. It must be satisfied $\psi(P) = \psi(P1) + \psi(P3) < \psi(P2) + \psi(P3) = \psi(Q)$, $E(P) = E(P1) + E(P3) \geq E(P2) + E(P3) = E(Q)$, and $M(P) = M(P1) + M(P3) \geq M(P2) + M(P3) = M(Q)$, then we can get $P \triangleleft Q$, so the path P is a contradiction of the non-dominated path. The original proposition is proved.

Theorem 6.1 *In the loop-free directed graph model, the optimized path's subpaths are non-dominated.*

Proof According to Definition 6.3, the optimal path is a non-dominated path from vertex v_s to vertex v_i. Utilizing Lemma 6.1, it can be established that the subpaths of the non-dominated path are also non-dominated paths. Therefore, the subpaths of an optimized path are non-dominated paths as well, and the initial proposition is thereby validated.

The label updating algorithm is utilized to locate the optimal paths by exploring all non-dominated paths from the vertices v_s in graph. The algorithm eliminates the dominated paths from v_s to each vertex, continually, effectively reducing the number of paths explored. The fundamental principle is to sustain a collection of tokens for each vertex in the graph, recording the vertex-to-vertex route. Each vertex is analyzed sequentially based on its topological order, and the label values of all its succeeding vertices are updated. When the update procedure arrives at vertex v_t, all the optimized paths from vertex v_s to vertex v_t are found. The graph's loop-free directed feature guarantees that only the set of labels of the successor vertices of the presently analyzed vertex needs to be updated, without considering its forward converging vertices. Consequently, each vertex in the graph is visited only once, significantly enhancing the efficiency of the optimized path search. The Label Updating-based Exact Path Searching algorithm (LUEPS) is outlined below.

Algorithm name: LUEPS
Input: directed acyclic graph $G = (V, E)$
Output: from vertex v_s to the vertex v_t the set of optimized paths Label(t)

begin
Label$(s) = \{0, 0, 0, s, \text{Null}\}$
Label$(t) = \{0, 0, 0, s, \text{Null}\}$
Label$(i) = \{\psi(i), E(i), M(i), s, \text{Null}\}, \forall v_i \in V(G)\backslash\{v_s, v_t\}$
for $i = 1, 2, \ldots, t$
  for each $Q \in$ Label(i)
    for each $j \in$ succ(Q)
      for each $P' =$ append(Q, j)
        if $P \lhd P'$ then
          Label$(j) =$ Label$(j)\backslash\{l_P\}$
          Label$(j) =$ Label$(j) \cup \{V(P'), E(P'), M(P'), i, \text{point}_{P'}\}$
        else if $P' \lhd P$ then
          continue
        else
          Label$(j) =$ Label$(j) \cup \{V(P'), E(P'), M(P'), i, \text{point}_{P'}\}$
        end if
      end for
    end for
  end for
end for
end

The LUEPS algorithm employs the following notation: Let Label(i) be the set of labels of vertex v_i. Each label in Label(i) represents a path from the

vertex v_s to vertex v_i. $\forall l_P \in \text{Label}(i)$, the label l_P is expressed as $l_P \equiv \{\psi(P), E(P), M(P), \text{pre}^i(P), \text{point}_P\}$. The meaning of $\psi(P)$, $E(P)$, and $M(P)$ has been illustrated in Definition 6.1. $\text{pre}^i(P)$ denotes the predecessor vertex of v_i in path P. Note that, vertex v_s has no predecessor vertex. point_P is a pointer to the path P that records all the vertices in the path P from vertex v_s to $\text{pre}^i(P)$. In statement 3, $\psi(i)$, $E(P)$ and $M(P)$ denote the evaluation value, consumed energy, and consumed storage resources in the programme, which are represented by the path from vertices v_s to vertex v_i. $\text{succ}(P)$ denotes the set of all successor nodes of the last vertex in path P. Due to the dynamic topological properties of the loop-free directed graph (see Fig. 6.2), different paths may have different sets of successor vertices, even if their last vertex is the same. Therefore, it is necessary to compute $\text{succ}(P)$ in real-time based on the path P while searching. If the last vertex of the path P is vertex v_t, then $\text{succ}(P) = \emptyset$. The function $\text{append}(P, i)$ adds vertex v_i to the end of the path P and generates a new path.

LUEPS is an accurate search algorithm that can find all optimized paths within the distance from the current position of the satellite to the satellite's autonomous scheduling horizon. The relevant lemmas are presented first.

Lemma 6.2 *According to the LUEPS algorithm, for any vertex $v_i \in V(G)\backslash\{v_s\}$, let $P \in \text{PATH}_{s,i}$ be an non-dominated path, and once the path P added into the set of tokens* $\text{Label}(i)$*, it will always remain in* $\text{Label}(i)$.

Proof From statements 10 to 11 of the LUEPS algorithm, if v_i is a vertex in the path P and its label l_P is removed (l_P is the label of path P), then there must exist another path P' such that $P \lhd P'$. So if a label representing a non-dominated path once enters the label set $\text{Label}(i)$, it will always remain in $\text{Label}(i)$. The original proposition is proved.

Lemma 6.3 *According to the LUEPS algorithm, for any vertex $v_i \in V(G)\backslash\{v_s\}$ that* $\text{Label}(i)$ *contains all the non-dominated paths from the vertex v_s to the vertex v_i.*

Proof The proof is carried out by mathematical induction. Based on the topological ordering of the vertices in a loop-free directed graph, it may be helpful to set the vertex order to be $s, 1, 2, \ldots, t$.

Step 1: When $i = 1$, (where $v_i \in V(G)\backslash\{v_s\}$). The vertex v_1 has only one predecessor node v_s, and so there can be only a unique path from the vertex v_s to vertex v_1. Statement 3 in the LUEPS algorithm is initialized with the guarantee that Label(1) contains all vertices v_s to the vertex v_1 of non-dominated paths.

Step 2: Suppose that when $i \leq m$ (where m is the ordinal number of the last vertex in the path), the original proposition holds. Then when $i = m + 1$, suppose there exists a non-dominated path $P \in \text{PATH}_{s,m+1}$ (whose label is l_P), and $l_P \notin \text{Label}(m + 1)$. Let v_j is the predecessor node of v_{m+1} in path P, and $1 \leq j < m$ (since statement 3 guarantees that $\text{Label}(m + 1)$ contains all the labels of paths whose predecessor node is v_s, so we only consider the case of $1 \leq j < m+1$). Let $P = \text{append}(Q, v_j)$,

then $Q \in \mathrm{PATH}_{s,j}$ be the subpath of P that excludes v_{m+1}. From Lemma 6.1, it is clear that Q is also a non-dominated path. It follows from the assumption that Label(j) contains a subpath representing the path Q of the label l_Q. Then, according to the LUEPS algorithm statements 7–16, the path P will be generated and added to Label($m+1$). Then by Lemma 6.2, it can be seen that it will always be kept in Label($m+1$). Contrary to the assumption $l_P \notin$ Label($m+1$) contradicts. So when $i = m+1$ time, the original proposition holds. In conclusion, the original proposition was proved.

Lemma 6.4 *According to the LUEPS algorithm, for any vertex $v_i \in V(G)\backslash\{v_s\}$, the* Label($i$) *only contains the non-dominated paths from the vertex v_s to vertex v_i.*

Proof Let Label(i) be the label set of vertices v_i, it contains the label l_P that representing path P, where $P \in \mathrm{PATH}_{s,i}$ and P is a dominated path. Let $\exists Q \in \mathrm{PATH}_{s,i}$ and $P \triangleleft Q$. It is known that the token l_Q of path Q must also be contained in Label(i) by Lemma 6.3. According to statements 7–12 of the LUEPS algorithm, it is known that l_P has been removed from Label(i) (statements 8 ~ 9) or will not be added into Label(i) (statements 11 ~ 12), contradicting the assumption $l_P \in$ Label(i). The original proposition is proved.

Theorem 6.2 *The result of running the LUEPS algorithm contains all optimized paths and only those paths.*

Proof The result of running the LUEPS algorithm is the set of contained paths Label(t). By Lemmas 6.3 and 6.4, it is known that when the vertex $i = t$ is reached, then Label(t) contains and only contains all non-dominated paths from v_s to v_t. Then Label(t) contains and only contains all optimization paths. The original proposition is proved.

The ultimate objective of satellite autonomous task scheduling is to find an optimized path with the highest evaluation value from the scheduling start time to the scheduling end time. The purpose of reserving all non-dominated paths is to reuse the search results of the LUEPS algorithm for rescheduling when a new earth observation meta-task is added to the directed acyclic graph model. An optimized path represents a feasible solution for autonomous satellite operation at a given moment T. The LUEPS algorithm guarantees obtaining all optimized solutions within the autonomous scheduling horizon as per Theorem 6.2. Usually, the autonomous operation satellite should select the optimized solution with the highest evaluation value. However, in some scenarios (such as when the satellite is close to running out of energy), the path with the highest onboard evaluation value (which consumes more energy) may not necessarily be the most optimal solution for the current situation. A more scientific approach is for the satellite to utilize a rule-based expert system to autonomously select an optimized solution from the set of optimized solutions and execute it.

To accommodate onboard scheduling characteristics, we adopt a scheduling strategy with a variable satellite autonomous scheduling horizon.

We consider the node position farthest from the satellite's current position to the set of generated labels that can be calculated within the time of the satellite's overflight critical task as the range of the autonomous scheduling horizon. The autonomous scheduling horizon may be shorter when earth observation meta-tasks are more densely distributed and longer when they are more sparsely distributed, but always less than or equal to the maximum visible distance in the forward direction of the satellite.

In this section, we analyze the time complexity of the LUEPS algorithm, which is influenced by the topological relations of the directed acyclic graph model. The time complexity of LUEPS for a problem size of n is $O\left(n^2L^2\right)$, where $L = \max(|\text{Label}(i)|),\ v_i \in V(G)\backslash\{v_s\}$. In the worst-case scenario where directed edges connect each vertex, and none of the paths are dominated by each other, the vertex with the highest number of labels is vertex v_t. By applying knowledge of permutations and combinations, $\max(|\text{Label}(t)|) = 1 + C_n^1 + C_n^2 + \cdots C_n^n = 2^n$, the worst-case time complexity of the LUEPS algorithm is $O(2^n)$. This confirms that the satellite task scheduling problem is NP-complete, as previously shown in [1]. Although previous studies have indicated that the number of non-dominated paths from vertex v to an intermediate vertex is relatively insignificant in practice [161], the computational resources onboard the satellite are limited. As a result, the algorithm's exponential time complexity may cause a short autonomous scheduling horizon or even scheduling failures. Therefore, efficient algorithms or heuristics are required to reduce the computational time and improve the scheduling horizon.

6.2.4 *Label Updating-Based Approximation Search Algorithm*

The LUEPS algorithm is faced with the challenge of saving all non-dominated paths from the starting vertex to every other vertex, leading to a substantial increase in search time as the number of vertices increases. Consequently, it is only appropriate for solving satellite task scheduling problems in small-scale scenarios. The Label Updating-based Approximate Path Searching algorithm (LUAPS) has been proposed to address this challenge. The LUAPS algorithm introduces the concept of path approximation, which decreases the number of labels assigned to each vertex by relaxing the path dominance relations and eliminating many similar paths. This reduction in the search space saves search time, thereby enhancing the efficiency of the algorithm. To establish a common understanding, the relevant definitions are presented below.

Definition 6.4: Path approximation dominance Let $P1, P2 \in \text{PATH}_{s,i}$, path $P2$ is dominant to $P1$ if it satisfies one of the following two conditions and is denoted as $P1 \prec P2$.

(1) $\psi(P1) < \psi(P2), E(P1)(1+\varepsilon_1) \geq E(P2), M(P1) \geq M(P2),$

(2) $\psi(P1) < \psi(P2), E(P1) \geq E(P2), M(P1)(1+\varepsilon_2) \geq M(P2),$

where $\varepsilon_k = \delta_k/2n$, δ $(0 < \delta < 1)$ is the approximation factor, and n is the problem size (the number of tasks in the autonomous scheduling horizon), the $k = \{1, 2\}$. From Definitions 6.1 and 6.4, it can be shown that if $P1 \triangleleft P2$, then we have $P1 \prec P2$. $\text{PATH}_{s,i}$ is the set of all paths from vertex v_s to vertex v_i. $\psi(P)$, $E(P)$ and $M(P)$ denote the evaluation value, the consumed energy, and the consumed storage for the plan represented by path P, respectively.

The LUAPS algorithm is built upon the LUEPS algorithm. The primary difference between the two algorithms is that the LUAPS algorithm incorporates the function TrimLabel (Label (j)) to eliminate unnecessary labels from vertex j based on the path approximation dominance relation (refer to Definition 6.4). TrimLabel (Label (j)) function is integrated into the LUEPS algorithm after operation 16. The algorithm TrimLabel algorithm is described as follows.

Algorithm name: TrimLabel
Input: Label(j) (label set of vertex j)
Output: Label(j) (label set of vertex j after pruning)

begin
1 LB = Label(j)
2 Label(j) = $\emptyset$
3 Sort(LB)
4 l_P = LB[0]
5 $i = 1$
6 while $i <$ |LB|
7 $l_{P'}$ = LB[i]
8 if $P' \prec P$ then
9 $i = i + 1$
10 else if $P \prec P'$ then
11 $l_P = l_{P'}$
12 $i = i + 1$
13 else
14 Label(j) = Label(j) $\cup \{l_P\}$
15 $l_P = l_{P'}$
16 $i = i + 1$
17 end if
18 end while
19 Label(j) = Label(j) $\cup \{l_P\}$
end

In the algorithm, operation 3 entails sorting all labels in LB according to their energy consumption in ascending order. lb[i] represents the ith label in LB. Operations 1–4 constitute the initialization process, while operations 5–19 follow the path approximation dominance relation (refer to Definition 6.4) for label pruning. Next, we will examine the distinctive features of the LUAPS algorithm.

Theorem 6.3 *The LUAPS algorithm is a polynomial time algorithm.*

Proof From the TrimLabel algorithm, the worst time complexity of the TrimLabel algorithm is $O(L \cdot \ln(L))$, where $L = \max(|\text{Label}(i)|)$, $v_i \in V(G)\backslash\{v_s\}$. Then the worst time complexity of the LUAPS algorithm is $O\left(n^2 L^2 \ln(L)\right)$, where n is the problem size. For the approximate dominance relation condition (1), note that Engy is the maximum level of the onboard battery, then for $\forall P \in \text{PATH}_{s,t}$, we have $E(P) \leq \text{Engy}$. From the TrimLabel algorithm and Definition 6.4, it is known that the approximate dominance relation divides the energy interval [0, Engy] into at most $\left\lfloor \log_{1+\varepsilon_1} \text{Engy} \right\rfloor$ parts ($\lfloor\rfloor$ is a downward integer function), and adding 0 and Engy two elements, then for $\forall v_i \in V(G)\backslash\{v_s\}$, it satisfied:

$$|\text{Label}(i)| \leq \log_{1+\varepsilon_1} \text{Engy} + 2 \tag{6.12}$$

$$\log_{1+\varepsilon_1} \text{Engy} = \frac{\ln \text{Engy}}{\ln(1+\varepsilon_1)} \leq \frac{1+\varepsilon_1}{\varepsilon_1} \ln \text{Engy} = \frac{2n \cdot \ln \text{Engy}}{\delta_1} + \ln \text{Engy} \tag{6.13}$$

By substituting Eq. (6.12) into Eq. (6.13) yields.

$$|\text{Label}(i)| \leq \frac{2n \cdot \ln \text{Engy}}{\delta_1} + \ln \text{Engy} + 2. \tag{6.14}$$

In Eq. (6.14), δ_1 and $\ln \text{Engy}$ are constants, so the worst time complexity of LUAPS algorithm is $O\left(n^2 \cdot n^2 \cdot \lg(n)\right) = O\left(n^4 \cdot \lg(n)\right)$. So LUAPS is polynomial time algorithm.

Similarly, it can be shown that the complexity of satisfying the approximation relation condition (2) is $O\left(n^4 \cdot \lg(n)\right)$.

The original proposition is proved.

In comparison to the LUEPS algorithm, the LUAPS algorithm may not always identify the optimal path with the highest evaluation value within the autonomous scheduling horizon. Nevertheless, due to the LUAPS algorithm's better computational time complexity, it typically features a longer autonomous scheduling horizon for satellites than the LUEPS algorithm. Research has shown that larger autonomous scheduling horizons tend to result in better scheduling outcomes for satellites [91]. Consequently, in some cases, the LUAPS algorithm can deliver superior scheduling results to the LUEPS algorithm.

As previously mentioned, the earth observation meta-tasks within the satellite's autonomous scheduling horizon can shift at any time, with two possibilities: earth observation meta-task joining and earth observation meta-task dropping out. In the case of earth observation meta-task joining, the algorithm only needs to verify whether the vertices within their corresponding graph can be added to the path of each vertex, one by one, and update the label value accordingly. Conversely, for earth observation meta-task dropping out, it suffices to delete the path containing the associated vertex of the task in each vertex. The specific details of these actions are not repeated here.

6.3 Machine Learning Methods for Satellite Onboard Task Decision-Making

6.3.1 Observation Task Sequential Decision-Making Model

Section 6.1.1 describes the satellite onboard autonomous tasks scheduling problem, revealing that it is unnecessary to simultaneously compute the observation programme for all the earth observation meta-tasks in the autonomous scheduling horizon. Instead, it suffices to identify the subsequent earth observation meta-task to execute after the current one is completed, which is a classic sequential decision-making process. To address this, we have introduced a novel solution, the sequential decision-making model for the satellite onboard autonomous task scheduling problem (Fig. 6.5). The core principle underlying this model is that, while executing the current earth observation meta-task, the autonomous operation satellite sequentially determines decisions about future earth observation meta-tasks chronologically until it identifies the next one to be completed. This model's advantage lies in the satellite's ability to make real-time decisions on upcoming earth observation meta-tasks based on the real-time status information, such as energy status, memory status, satellite attitude, and target distribution information. Thus, it can swiftly respond and adapt to changes in the earth observation meta-tasks, significantly improving the satellite's responsiveness to a highly dynamic operating environment.

Let t_{sys} denote the system time, task_0 denote the earth observation meta-task to be decided, H_{d} denote the minimum decision duration, and $\left[t_{\text{sys}}, t_{\text{sys}} + H_{\text{d}}\right]$ denote the task decision horizon. The sequential decision-making process proceeds as follows: while executing the current earth observation meta-task, the autonomous operation satellite sequentially decides on the earth observation meta-tasks in the task decision horizon to identify the first earth observation meta-task to be executed. If there is no earth observation meta-task in the task decision horizon, or no decision on the earth observation meta-task to be completed, the end time of the decision horizon becomes the next decision time point. Figure 6.5 illustrates the onboard planner sequentially making decisions for earth observation meta-tasks 5, 6, 7, and 8 in the task decision horizon. The decision outcome for tasks 5 and 6 is "false," while the decision outcome for 7 is "true," making task 7 the subsequent task to execute. The algorithm then exits the decision process, awaiting the next round.

Algorithm name: sequential decision-making for earth observation tasks
Input: $\text{TASK}_{\text{decision}}$: the set of earth observation meta-tasks in the task decision horizon, t_{sys}: the system time onboard
Output: task_{do}: the next task to be performed, t_{next}: the next decision time point

---

(continued)

(continued)

begin
1 $t_{\text{next}} \leftarrow t_{\text{sys}} + H_{\text{d}}$
2 $\text{task}_{\text{do}} \leftarrow$ null // Initialize the next execution task
3 for $\text{task}_i \in \text{TASK}_{\text{decision}}$ do
4 **if** constraintcheck(task_i) then // determine if the earth observation meta-task violates the constraint
5 continue
6 end if
7 do $\leftarrow$ makedecision(task_i) // Decision on whether the current earth observation meta-task is executed
8 if do is True, then
9 $\text{task}_{\text{do}} \leftarrow \text{task}_i$ // Update the next implementation task
10 $t_{\text{next}} \leftarrow \text{ta}_i$ // Update next decision moment
11 break
12 end if
13 end for
end

The sequential decision model for observation tasks aims to determine when and which earth observation meta-tasks to decide on. Once this is established, the subsequent inquiry pertains to the algorithm best suited to determine the performance of the earth observation meta-task. Supervised learning [162] is a significant class of machine learning methods that can learn a functional mapping of inputs to outputs from training samples. By applying the supervised learning method to learn implicit decision knowledge from actual task scheduling data, we can effectively combine it with the sequential decision model of observation tasks to achieve airborne autonomous real-time decision-making of earth observation meta-tasks.

To achieve this goal, it is necessary to analyze the classification features that can be employed in the machine learning method for the sequential decision-making model. Additionally, we will introduce the corresponding sequential decision algorithms for the observation mission represented by ensemble learning and deep neural networks, as illustrated in Fig. 6.6.

Training an observation mission decision-making model requires substantial computing resources and numerous training samples. Therefore, training the model on the ground and then uploading it to the satellite is more feasible.

6.3.2 Features for Observation Task Decision-Making

The extraction and selection of decision features are crucial in representing the raw data when employing machine learning methods for decision-making regarding whether a earth observation meta-task can be executed. From a global perspective, earth observation meta-tasks compete for limited energy, storage, and equipment resources onboard. Performing more tasks can result in the payload working longer and consuming more resources, leading to insufficient recharge time and

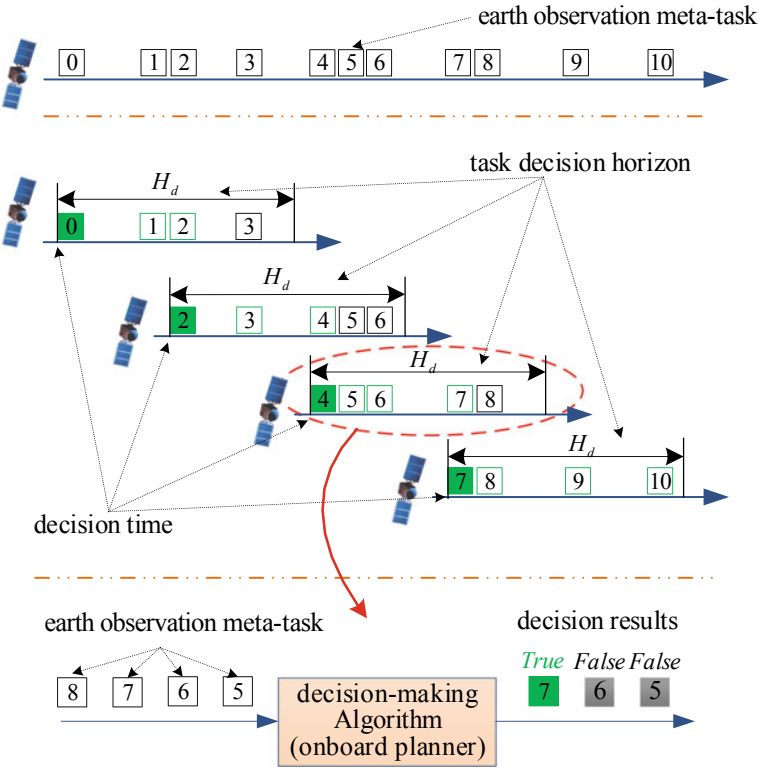


Fig. 6.5 Sequential decision-making for satellite earth observation missions model

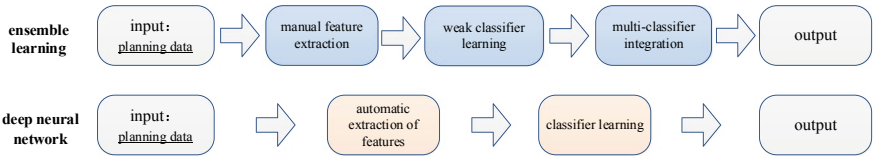


Fig. 6.6 Workflow of ensemble learning and deep neural networks

fewer resources for subsequent earth observation meta-tasks. Conversely, underutilization of satellite resources and ineffective earth observation can occur if fewer tasks are performed. Thus, when deciding whether to execute a earth observation meta-task, attributes and distribution of subsequently planned meta-tasks should also be considered, in addition to its features and available resources.

To fully represent earth observation meta-task distribution information, resource competition relationship, and conflict degree, several relevant features were identified. We designed five types of features: resource and equipment state features,

features of pending earth observation meta-task, features of conflicting earth observation meta-tasks, features of non-conflicting earth observation meta-tasks, and optimal neighborhood features.

(1) **Resource and equipment state features**

Resource and equipment state features provide a comprehensive overview of the state of resources and equipment of the earth observation satellite, such as energy levels, storage capacity, and attitude. These features are crucial in determining the feasibility of executing a given task.

(2) **Pending earth observation meta-task features**

The pending earth observation meta-task refers to the current task under consideration, and its features describe the attributes of this task. These features mainly include the observation priority, observation duration, energy consumption, memory consumption, and other relevant characteristics of the earth observation meta-task.

(3) **Conflicting earth observation meta-task features**

Conflicting earth observation meta-tasks refer to a set of earth observation meta-tasks that cannot be executed after the execution of the pending earth observation meta-task due to constraints such as work mode switching, energy, and storage. Their features include the sum of priorities and the total number of these conflicting tasks.

(4) **Non-conflicting earth observation meta-task features**

As previously discussed, a potential competition exists among earth observation meta-tasks for the limited energy, storage, equipment, and other satellite resources. In order to ensure the optimality of the solution and prevent the task sequential decision-making algorithm from making non-optimal decisions due to a "short-sighted" approach, it is imperative to incorporate information on the distribution of all earth observation meta-tasks during the autonomous scheduling horizon.

(5) **Neighborhood optimal features**

The feature quantity known as "neighborhood optimal" describes whether a property of the current earth observation meta-task is the largest in its spatiotemporal neighborhood. For example, this feature can be used to determine whether the priority of the earth observation meta-task is the highest in the neighborhood, whether the observation duration is the longest, whether the sum of the conflicting target priorities is the largest, whether the number of conflicting earth observation meta-tasks is the largest, whether the energy consumption is the largest, and whether the storage space consumption is the largest.

It is important to note that overly complex decision network models may pose challenges for autonomous task decision-making due to limited onboard computing resources. The features mentioned above, such as prior knowledge of task distribution, can help to reduce the complexity of the decision network by facilitating the learning of potential features and minimizing computational demands.

6.3.3 *Ensemble Learning-Based Sequential Decision-Making Approach*

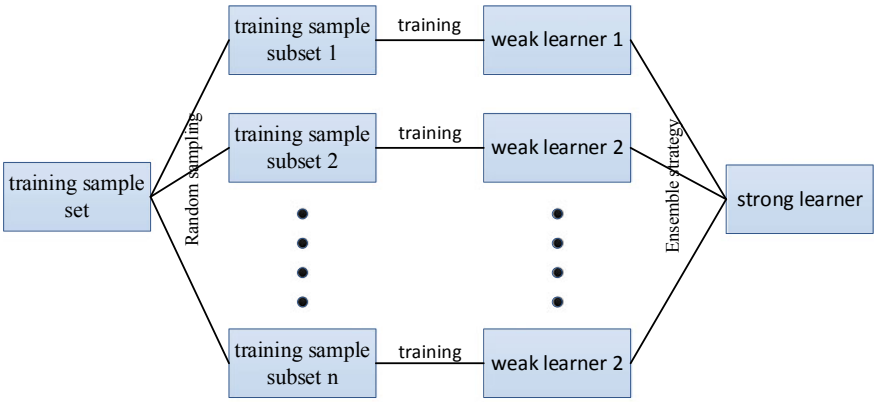
Ensemble learning [162] is a well-established machine learning technique that involves generating several weak classifiers by altering the distribution of the original training sample. These classifiers are then combined to produce a stronger, more robust classifier. Compared to simpler supervised learning methods, such as support vector machine, neural network, and decision tree, ensemble learning frequently outperforms these methods by combining the classification results of multiple classifiers.

Ensemble learning methods can be divided into two categories based on the different ways of selecting training samples and combining classifiers: bagging and boosting (as shown in Fig. 6.7). The bagging method employs a random sampling method with replacement to create multiple independent training sample subsets. These subsets are then used to train several independent classifiers, and their results are combined using a combination strategy to produce the final decision. Random forest (RF) algorithm is a typical representative of the bagging method. On the other hand, the boosting approach trains classifiers serially in steps, and the training samples of the current classifier are related to the learning results of previously trained classifiers. In each training iteration, the weights of misclassified samples are increased, and the weights of accurately classified samples are decreased according to their classification results. This approach alters the data distribution by updating the weights of the training samples. Finally, the results of multiple classifiers are synthesized using a combination strategy to produce the final decision. Gradient Boosting Decision Tree (GBDT) is a typical representative of the boosting method.

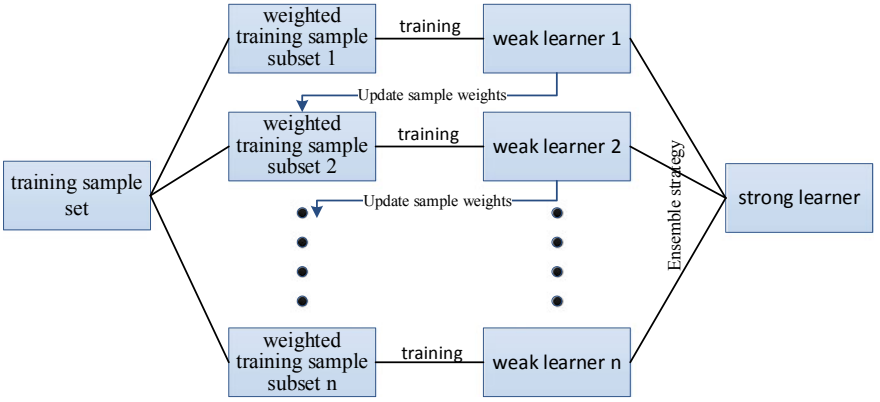
In conclusion, ensemble learning is a powerful technique combining multiple classifiers' results to create a stronger, more accurate classifier. The bagging and boosting methods provide different ways to generate weak classifiers and combine their results. These methods have been widely used in various fields and have successfully solved practical problems.

Both the bagging and boosting methods are ensemble learning techniques that combine multiple weak learners into a strong learner through a combination strategy. The most common combination strategies include the voting method, averaging method, and learning method. The voting method is mainly used for classification problems, where the final output is the category with the most votes among the outputs of multiple learners. The averaging method is primarily for regression prediction problems, where the final result is obtained by weighing the sum of the outputs of multiple learners. The learning method involves training a layer of learners to integrate the outputs of weak classifiers, and it is often referred to as the stacking integration method [163].

The base classifiers of bagging and boosting methods are typically decision trees and neural networks. The simpler the base classifier, the stronger the generalization ability of the integrated model, and the less likely overfitting occurs. In the bagging integration method, each base classifier is independent of the other, and there is no



(a) Bagging



(b) Boosting

Fig. 6.7 Schematic diagram of bagging, boosting ensemble model

dependency. Its training goal is to reduce the bias of the classifiers, so the depth of the decision tree model is usually set large. In contrast, the boosting method fits each primary classifier to the data based on the trained basic classifiers, reducing the model's bias in each iteration. Only the basic classifier with minor variance is selected to avoid model overfitting as much as possible. Therefore, the basic model of the boosting method must be weak, and the depth of the decision tree model is usually set small.

We can use ensemble learning algorithms such as RF and GBDT to train the earth observation meta-task decision-making model. Since these algorithms are mature,

they are not described in detail here. Interested readers can refer to [162] for further information on RF and GBDT models.

1. Decision-making model training process

The training of the decision-making model for observation tasks requires a large number of samples in the shape of $(\boldsymbol{x}, y)$, where $\boldsymbol{x}$ is the input feature vector for the earth observation meta-task, and $y \in \{0, 1\}$ is the decision result. However, a given satellite task scheduling instance comprises a series of earth observation meta-tasks and scheduling results. Therefore, it is necessary to convert the task scheduling instances and earth observation meta-tasks into samples for model training.

To address this, we propose an algorithm for generating training samples based on the online decision process for the earth observation meta-task. The algorithm computes the features for each earth observation meta-task in time order, from earliest to latest, and sets its label value to 1 if it is included in the programme; otherwise, it is set to 0. It is important to note that earth observation meta-tasks that cannot be executed due to constraint violations (such as energy, storage, and work mode switching) are not required to be decided in the real-time decision process. Therefore, earth observation meta-tasks that cannot be executed due to constraint violations will be discarded and not included in the training samples.

Algorithm name: Training sample generation algorithm
Input: mission set (TASK), earth observation programme (P), ensemble learning model (model)
Output: labeled training samples

begin
samples ← null
for $\text{task}_i \in$ TASK do
if not constraintscheck(task_i) then // determine if the earth observation meta-task violates the constraint
continue
end if
$\text{TASK}_{\text{decision}}$ ← tasksindecisionhorizon() // Obtain earth observation meta-tasks in the task decision horizon
feature_i ← getfeature(task_i, $\text{TASK}_{\text{decision}}$)
if $\text{task}_i \in P$ then
label_i ← 1
else
    label_i ← 0
end if
    samples ← samples ∪ (feature_i, label_i)
end for
end

6.3.4 Deep Neural Network-Based Sequential Decision-Making Approach

Deep neural networks, which can automatically extract features through specific network structures and avoid complex feature engineering, have significantly progressed in recent years [164–167]. Convolutional neural networks (CNNs), as a type of deep neural networks, have become a significant tool for extracting image features automatically through multiple convolutional layers. CNNs are widely used in various computer vision fields, such as image classification, segmentation, and generation. In image classification, CNNs can accurately classify images into different categories based on their features. In image segmentation, CNNs can identify and segment different objects within an image. Furthermore, CNNs can also be used in image generation, where they can generate new images based on a given set of inputs. Therefore, CNNs have proved to be a versatile and effective technique for various computer vision tasks, making them a popular choice among researchers and practitioners [168, 169]. The achieved results are usually better than traditional machine learning methods based on feature engineering. Recurrent neural network (RNN) can automatically extract serial data features through a network structure with feedback and is used in natural language processing (NLP) problems such as speech recognition, machine translation, picture naming, etc. [170, 171]. The advent and utilization of novel techniques and methodologies, such as deep neural networks, present fresh perspectives for our research while simultaneously posing new challenges. In this section, we propose sequential decision-making algorithms for satellite earth observation tasks, based on two prevalent types of deep neural networks: convolutional neural networks and recurrent neural networks. These algorithms are designed to cater to the specific requirements of satellite earth observation tasks, by effectively utilizing the capabilities of convolutional and recurrent neural networks. The proposed algorithms aim to enhance the accuracy and efficiency of satellite earth observation tasks, thereby enabling a more comprehensive understanding of the earth's surface and its dynamics.

1. Deep neural network decision model for earth observation tasks

(1) Decision-making model for earth observation tasks based on convolutional neural networks

Figure 6.8 illustrates the decision-making model for earth observation tasks, which is based on a convolutional neural network. The encoding network comprises two convolutional layers and two pooling layers, which are employed to automatically extract the earth observation meta-task features through multi-layer convolutional operations. The classification network consists of a three-layer fully connected neural network with two output nodes, which produce the probability of each category (1 for execution and 0 for non-execution) based on the abstract features derived from the encoding network. Finally, the category with the highest output probability is selected as the ultimate decision outcome. The proposed decision-making model

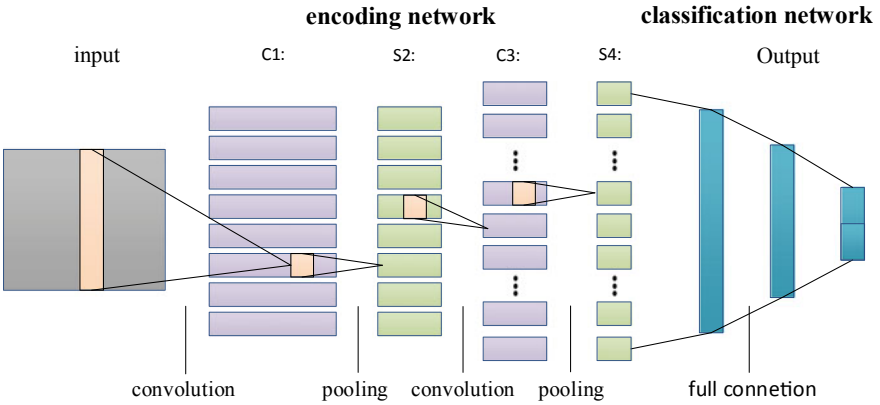


Fig. 6.8 Decision-making model for earth observation tasks based on convolutional neural networks

provides an efficient and accurate approach to address earth observation tasks by utilizing the capabilities of convolutional neural networks.

(2) Decision-making model for earth observation tasks based on recurrent neural network

The decision-making model for earth observation tasks based on a recurrent neural network consists of coding and classification networks as shown in Fig. 6.9. The coding network uses a recurrent neural network structure to extract the related features of the earth observation meta-task. Then the output of the last hidden layer is input into the classification network. Finally, the classification network calculates the probability of each category by a fully connected neural network and uses it to determine the execution status of the task.

Among coding networks, recurrent neural networks differ depending on the structure of the recurrent neurons employed. Long short-term memory (LSTM) [172] is

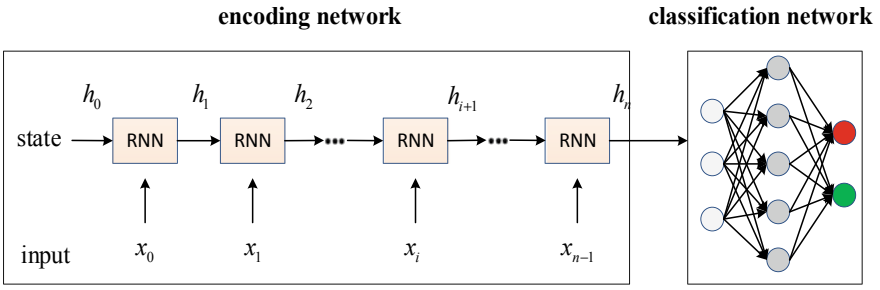


Fig. 6.9 Decision-making model for earth observation tasks based on recurrent neural network

a classical recurrent neural network. Due to its unique gate structure design, it effectively solves the long-term dependence problem of sequence data and achieves satisfactory results in processing time-series data. There are many textbooks and literature on convolutional neural networks and recurrent neural networks, the details of which can be found in the literature [167] and will not be repeated in this section.

The two deep neural networks proposed for earth observation tasks decision-making have slight variations in their applicability to different scenarios. The earth observation meta-task, which can be considered time-series data in the satellite task scheduling problem, is more effectively handled by the recurrent neural network-based decision model, which generally produces better decision results but requires longer training time. In contrast, the convolutional neural network-based decision model has the inherent parallel computational properties of convolution and pooling, resulting in higher training efficiency. Hence, if the onboard processor has sufficient computing power, using the recurrent neural network to make observation task decisions onboard would be a suitable choice. However, in cases where computational resources are limited, it is recommended to use a convolutional neural network model.

2. The input of the task decision-making model

Using deep neural networks to decide the scheduling state of a earth observation meta-task requires converting the raw meta-task data into recognizable features. For convolutional neural networks, the input comprises multi-dimensional matrix data, while for recurrent neural networks, the input is sequence data. Determining whether a earth observation meta-task can be executed depends on the present state of resources, such as energy and storage, and the distribution of earth observation meta-tasks within the autonomous scheduling horizon. The earth observation meta-task in the autonomous scheduling horizon can be considered a sequence of earth observation meta-tasks in chronological order, which can also be viewed as two-dimensional matrix data. Since the input to the convolutional neural network must be matrix data of a specific size, the length of the input earth observation meta-task sequence can be set to a fixed value to meet its usage requirements simultaneously. In cases where the input sequence length is insufficient, the data is processed with complementary zeros to compensate.

The set $\mathrm{TASK}_{\mathrm{input}} = \{\mathrm{task}_0, \mathrm{task}_1, \ldots, \mathrm{task}_{k-1}\}$ is the sequence of earth observation meta-tasks arranged in chronological order from earliest to latest, and task_0 is the earth observation meta-task to be decided. For $\forall \mathrm{task}_i \in \mathrm{TASK}_{\mathrm{input}}$, its feature vector x_i can be expressed as $(\psi_i, \mathrm{eng}_i, \mathrm{mem}_i, t_{\mathrm{e}}^i - t_{\mathrm{b}}^i, t_{\mathrm{b}}^i - t_{\mathrm{e}}^{i-1}, \Delta\mathrm{eng}_{(i-1)i}, \Delta\mathrm{mem}_{(i-1)i}, c_{(i-1)i}, c_{0i}, \mathrm{BST}_i)$. The meaning of each element is as follows.

- ψ_i: The priority of the earth observation meta-task task_i.
- eng_i: The energy to be consumed if the earth observation meta-task task_i is executed.
- mem_i: The storage to be consumed if the earth observation meta-task task_i is executed.
- $t_{\mathrm{e}}^i - t_{\mathrm{b}}^i$: The duration of the earth observation meta-task task_i.
- $t_{\mathrm{b}}^i - t_{\mathrm{e}}^{i-1}$: The time interval between the earth observation meta-task task_i and previous earth observation meta-task task_{i-1}.

- $\Delta\text{eng}_{(i-1)i}$: The amount of energy replenished during the time interval from the earth observation meta-task task_i to earth observation meta-task task_{i-1}. If task_i is the first earth observation meta-task of the input sequence, then $\Delta\text{eng}_{(i-1)i}$ is the energy state at the current moment.
- $\Delta\text{mem}_{(i-1)i}$: The amount of storage state change from the earth observation meta-task task_i to earth observation meta-task task_{i-1}. The default value of $\Delta\text{mem}_{(i-1)i}$ is 0. If task_i is the first earth observation meta-task of the input sequence, then $\Delta\text{mem}_{(i-1)i}$ replaced by the storage state at the current moment onboard.
- $c_{(i-1)i}$: If the time interval from the access end time of task_{i-1} to the access start time of task_i satisfies the work mode switching constraint, then $c_{(i-1)i} = 0$, otherwise $c_{(i-1)i} = 1$.
- c_{0i}: If earth observation meta-tasks task_i and the earth observation meta-task task_0 satisfy all constraints, then $c_{0i} = 0$, otherwise $c_{0i} = 1$.
- BST_i: A Boolean variable set that represents whether some characteristics of the earth observation meta-task task_i are the largest in the neighborhood, including: priority, observation duration, sum of conflicting target priorities, number of conflicting targets, energy consumption, storage consumption, etc.

Based on the feature representation of the earth observation meta-task, we can construct the original input data for the convolutional neural networks and recurrent neural networks as shown in Fig. 6.10. In Fig. 6.10, each column corresponds to a earth observation meta-task, and each row corresponds to the feature value of the earth observation meta-task in that dimension.

3. Loss function

We use the cross-entropy loss function (see Eq. 6.15) [162] and Adam [173] optimization algorithm to train a deep neural network-based task decision-making model.

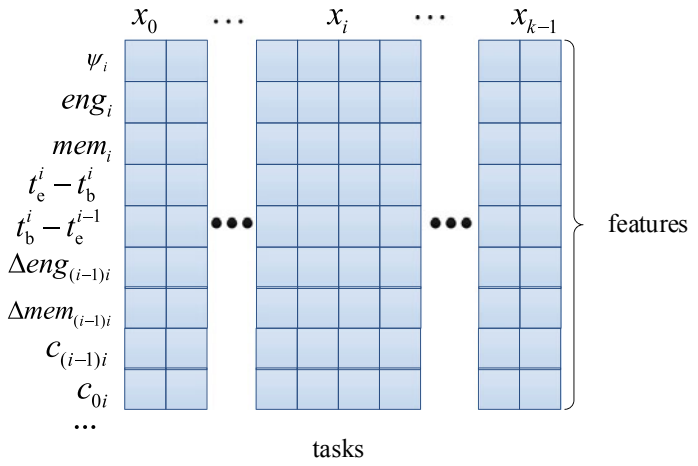


Fig. 6.10 Input representation of deep neural network

$$L = -\left[y \cdot \log \hat{y} + (1 - y) \log\left(1 - \hat{y}\right)\right], \tag{6.15}$$

which y and $\hat{y}$ denote the sample labels' real value and the model output's predicted value, respectively.

The training samples for the deep neural network-based task decision-making model are obtained like the ensemble learning-based task decision-making model. For details, please refer to Sect. 6.3.3.

4. Deep neural network compression

Despite significant advancements and improvements in onboard computing platforms in recent years, their computing and storage capacities are still relatively weak compared to ground-based systems. As a result, decision-making times may be prolonged, and even decisions may fail if the task decision-making network parameters are too numerous or the number of layers is excessively deep. Therefore, it is essential to design neural networks that are appropriate for the onboard computational capacity's depth and complexity rather than solely focusing on the decision-making effect without considering the decision-making network's complexity.

A primary point of view is that the shallow neural network compressed from the trained deep neural network has certain advantages over the direct trained shallow neural network regarding model representation, generalization, and robustness. For the satellite onboard autonomous task decision-making problem, the deep learning network training on the ground should be compressed into a shallow neural network that matches the onboard computing ability to operate normally in the resource-constrained embedded environment.

Deep neural network compression is an essential branch of machine learning research, and many research results have been achieved in recent years. The following methods are suitable for onboard autonomous task decision-making network compression.

(1) Network pruning method

The network pruning method is more intuitive and aims to discover those redundant connections in the deep neural network and remove them so that they are no longer involved in the network's forward or backward operations, reducing the amount of network computation. The removed neurons and the corresponding connections are also no longer stored, thus also reducing the number of parameters for the model.

The network pruning process results in a sparse deep neural network by removing some connections from the initially dense neural network. After completing the deep neural network training, the importance of each connection is defined using an evaluation criterion, such as the absolute value of the magnitude of the connection weights. This criterion is widely used since the smaller the weight value, the smaller the contribution of the corresponding neuron to the network output result. As a result, unimportant neurons in the network can be removed along with their corresponding connections. However, removing too many low-contributing neurons and connections may weaken the network output results and lead to error accumulation, reducing the network's decision accuracy. To restore the network performance,

fine-tuning the pruned network is a widely adopted treatment. The pruned network is alternately pruned and fine-tuned until achieving an optimal balance between model size and model performance. Network pruning methods are typically more effective in compressing fully connected layers due to their higher redundancy compared to convolutional layers.

(2) Weight quantification methods

Weight quantization methods compress neural networks by reducing the number of bits representing each weight. The idea of quantization is straightforward. It is to cluster the weight values (e.g., k-means clustering, etc.). The range of values of the network weights and activations are counted. After finding the maximum and minimum values, a min–max mapping is performed to map all weights and activations to the 8-bit integer range (-127 to 128). By quantizing the weights, the storage space occupied by the model can be reduced. If tens of millions of parameters are mapped from a 32-bit representation of a floating-point type to an 8-bit integer type, the size of the parameters is reduced to 1/4 of the original size. The size of the whole model is also reduced to 1/4 of the original size. Also, with the reduction of the model after parameter quantization, the computational resources required in the forward operation stage of the network will be significantly reduced.

Obviously, weight quantization will lose accuracy, which is equivalent to introducing noise to the network, but deep neural networks are generally less sensitive to noise. The impact on the final decision task accuracy can be very small as long as the degree of quantization is controlled. From the architectural point of view, another benefit of weight quantization is energy saving and chip area. Each number uses fewer bits and fewer data to carry when doing operations, reducing the access memory overhead (energy saving). At the same time, the number of multipliers required is also reduced (reducing chip area). The weight quantization method, therefore, is more suitable for use in an embedded environment.

(3) Low-rank approximation method

From a mathematical perspective, deep neural network computations can be seen as a series of matrix operations. The low-rank decomposition method optimizes the model computation process by breaking down these matrix operations into smaller components that have a precise mathematical interpretation. This approach is an effective way to reduce model redundancy and accelerate operations, particularly for fully connected layers. In convolutional neural networks, most of the computation is concentrated in the convolutional layer, where the parameters are typically saved in a multi-dimensional matrix format. The low-rank decomposition method, which is based on linear algebra theory, decomposes the large parameter matrix into smaller matrix combinations. This approach maintains the expressive ability of the original convolution layer while significantly reducing the amount of computation required. Therefore, the low-rank approximation method can approximate each convolution layer from shallow to deep with low-rank approximation orders, while ensuring a certain level of accuracy. This dramatically reduces the space and computation required for parameter storage. In summary, the low-rank decomposition method is an

effective way to optimize deep neural network computations. By decomposing matrix operations into smaller components with a precise mathematical interpretation, this method reduces model redundancy and accelerates operations, particularly for fully connected layers. In convolutional neural networks, this method significantly reduces the amount of computation required for parameter storage while maintaining the expressive ability of the original convolution layer. The disadvantages of the low-rank approximation are also more apparent. The decomposition is computationally expensive when the network is extensive. The low-rank approximation can only be performed layer-by-layer and cannot perform global parameter compression.

(4) Knowledge distillation method

Knowledge distillation is a current research area that has attracted significant attention with various implementation proposals. The basic idea is to use a pretrained large network, referred to as the teacher network, to train a smaller network, called the student network. The objective is to transfer the knowledge learned by the teacher network to the student network, which can then be used for classification, thus completing network compression. The underlying principle is to transfer knowledge from a cumbersome model to a smaller and more deployable one that is better suited to transfer learning. The process of knowledge distillation involves training the student network to fit the teacher network and learn knowledge from it. Unlike the three network compression methods mentioned earlier, the teacher and student networks can be completely different from each other. However, the distillation effect is usually better when the network structures are similar. It is important to note that during knowledge distillation training, the output distribution of the student network must be consistent with that of the teacher network. To achieve this, the training loss function commonly uses Kullback–Leibler (K-L) divergence or mean square error (MSE). In summary, knowledge distillation is a promising approach to network compression that involves transferring knowledge from a pretrained large network to a smaller network, which is better suited for deployment and transfer learning. During the distillation process, the student network is trained to fit and learn from the teacher network. The output distribution of the student network is required to be consistent with that of the teacher network, which is ensured using K-L divergence or MSE as the training loss function.

For the compression of the neural network, the specific compression method can be determined according to expert experience or experiment.

Chapter 7
Satellite Task Scheduling System

In Chaps. 3–6, some typical satellite task scheduling methods for centralized EOS task scheduling, dynamic rescheduling, distributed scheduling, and onboard autonomous scheduling are described in detail, which will eventually be applied to the satellite task planning and scheduling system. This chapter introduces several typical satellite task scheduling systems and simulation tools involved in the practice and focuses on analyzing more typical distributed task scheduling systems.

7.1 Typical Satellite Task Scheduling Systems and Tools

The satellite task scheduling system is an essential component of the earth observation system and the core component for efficiently utilizing satellite resources. Currently, all major spacefaring nations have designed and developed satellite task scheduling systems or simulation experiment tools for the management and task scheduling of satellites, such as NASA's Automated Scheduling and Planning Environment (ASPEN) system [111, 174, 175]. ESA's ESTRACK Planning System (EPS) [176, 177] and the generic space mission analysis tool—STK (Satellite Tool Kit) [178]. The following is a brief description of ASPEN and STK.

7.1.1 ASPEN/CASPER

ASPEN is an object-oriented task scheduling system (Fig. 7.1) that automatically generates spacecraft-executable low-level action instructions by coding spacecraft

H. Chen et al., *Earth Observation Satellites*,
https://doi.org/10.1007/978-981-99-3565-9_7

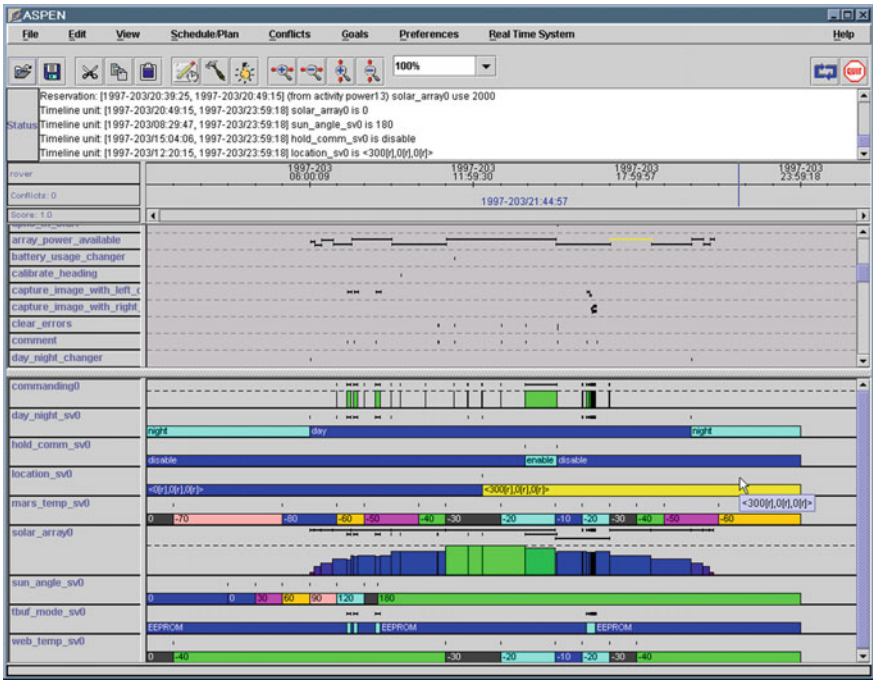


Fig. 7.1 Graphical visualization interface of ASPEN

operability constraints, flight rules, spacecraft hardware models, science experiment objectives, and operational procedures. ASPEN provides reusable software components that implement complex planning and scheduling systems, specifically.

(1) A uniform and easily expressible constraint modeling language: it allows users to customize the application and complete secondary development.
(2) The constraint management system represents and maintains spacecraft operability, resource constraints, and activity requirements.
(3) A range of search strategies for programme generation and repair.
(4) A language for representing programme preferences and optimizing those preferences.
(5) The real-time planning/replanning system-Continuous Activity Scheduling Planning Execution and Replanning (CASPER).
(6) A temporal reasoning system for expressing and maintaining constraints on satellite usage.
(7) A visual graphical interface for the presentation of planning results.

CASPER is the core of the ASPEN system, which uses local, heuristic iterative search methods for satellite task scheduling. It allows the spacecraft to adapt to changes in resources, tasks, and generate executable solutions quickly and rapidly. The core ideas are described in Sect. 1.2.1 and will not be repeated here.

7.1.2 Satellite Tool Kit—STK

STK is a commercial space mission analysis and simulation software developed by Analytical Graphics, Inc. It covers the entire space mission cycle of concept, requirements, design, manufacturing, testing, launch, operations, and applications. Its main features include.

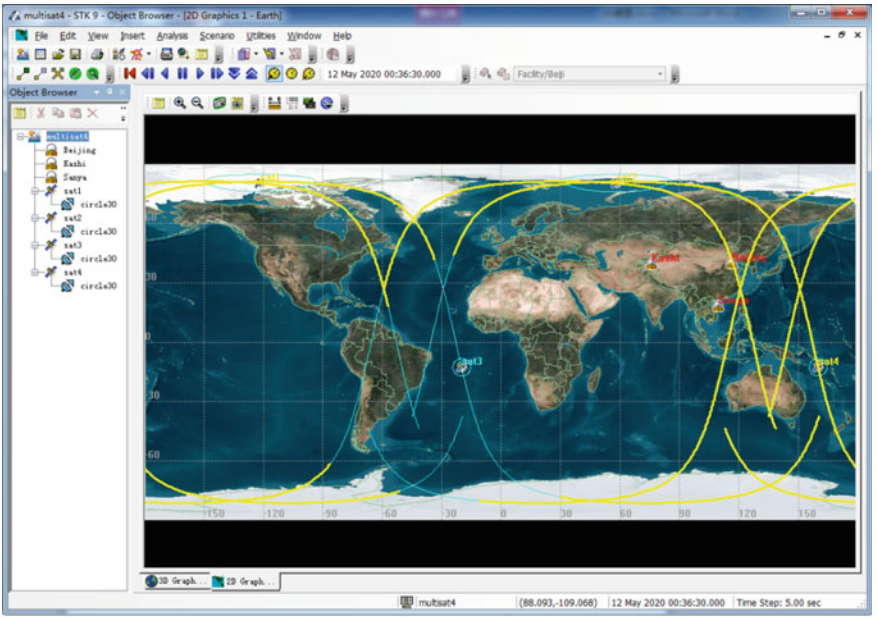
(1) Analysis capability: it can calculate the position and attitude of the satellite at any moment in time and the coverage area of the satellite or ground station remote sensor.
(2) Orbit generation: it provides satellite orbit generation wizards to help users create typical orbits, such as geosynchronous orbit, near-earth orbit.
(3) Visibility analysis: it supports calculating access times between space objects such as launch vehicles, missiles, aircraft, ground vehicles, targets, with the ability to add geometric constraints (e.g., visual range, minimum elevation angle, etc.) between objects for detailed simulation.
(4) Visual display: it supports visualization on 2D maps (see Fig. 7.2a), 3D maps (see Fig. 7.2b) visualization module to display all time-based information, with multi-window real-time display capability for task scene changes, etc.
(5) Comprehensive data reporting: it provides over a hundred types of reporting information in a text or graphical form, supporting user customization.
(6) STK can perform spacecraft task scheduling through heuristic algorithms, and the scheduling interface is shown in Fig. 7.2c.

7.2 Distributed Satellite Task Scheduling Systems

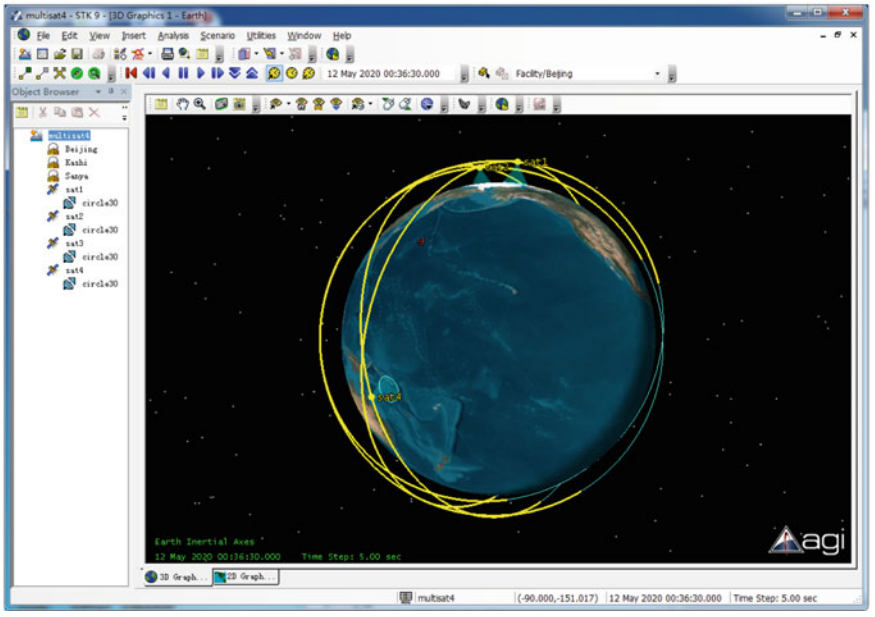
A distributed satellite task scheduling system based on the multi-agent system has emerged as a promising solution to achieve joint planning and scheduling of many earth observation satellites with good scalability and the ability to adapt to changes in tasks, ground resources, and satellites. In this section, we introduce the architecture and human–computer interaction interface of a typical distributed satellite task scheduling system, mainly referring to the work of literature [125, 136, 153, 179, 180].

7.2.1 System Architecture Design

The distributed satellite task scheduling system adopts a client–server system architecture (see Fig. 7.3). The server is deployed on the server cluster, responsible for receiving earth observation requirements and related resource status from external systems, running multi-satellite collaborative task scheduling algorithms, and generating satellite earth observation programmes. The clients are divided into central-user client and end-user clients. The central-user client provides general functions

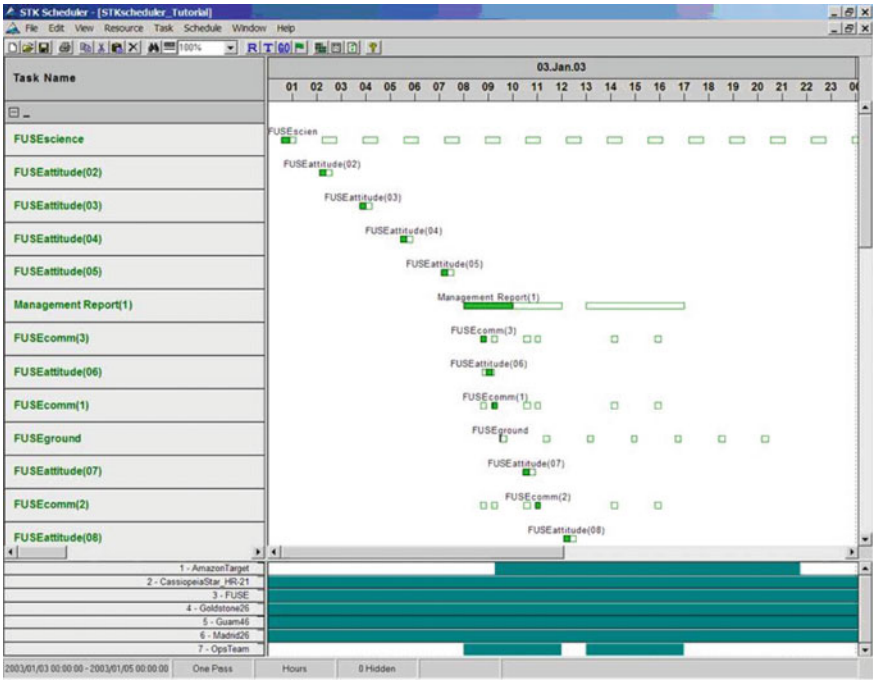


(a) Two-dimensional visualization interface



(b) Three-dimensional visualization interface

Fig. 7.2 Graphical visualization interface of STK



(c) Space resource planning interface

Fig. 7.2 (continued)

for task scheduling, including the whole scheduling result viewing and revision, etc. The end-user clients support various mobile platforms, including handheld and vehicle-mounted devices. The end-users can remotely log in to the satellite operation center through the end-user clients, submit their earth observation requirements and formulate satellite earth observation programme related to them.

According to the system hardware architecture, the composition of the distributed satellite task scheduling system software is shown in Fig. 7.4.

The distributed satellite task scheduling system's server is designed based on the multi-agent system architecture, comprising a cooperative scheduling agent module and several single-satellite scheduling agent modules. The system's client primarily integrates the earth observation solution display and human–computer interaction functions. This section focuses on the server module, while the subsequent section will introduce the client module.

The collaborative scheduling agent is responsible for managing each single-satellite scheduling agent and collaborating with each to complete the allocation of earth observation tasks. The functions of the modules are as follows.

- Process management and monitoring unit: its primary function is to control and manage the collaborative process of each module, monitor the operational status

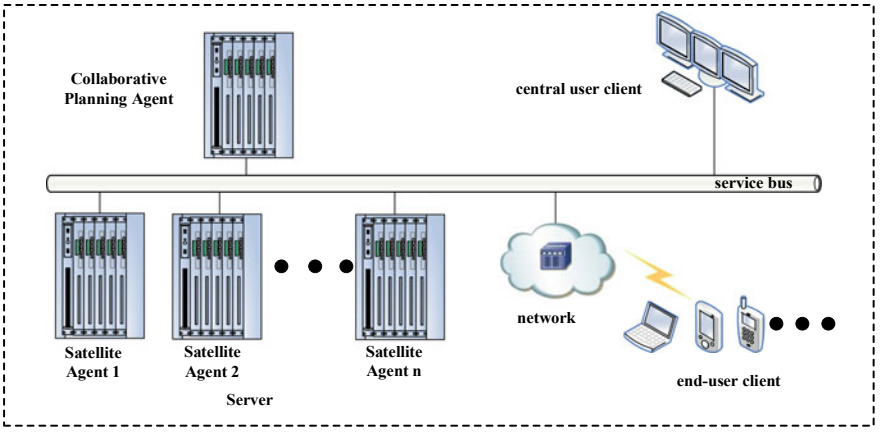


Fig. 7.3 Hardware components of the distributed satellite task scheduling system

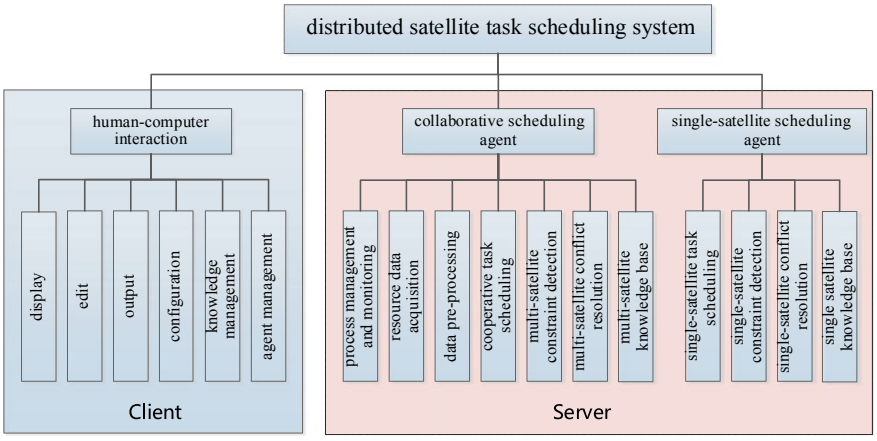


Fig. 7.4 Main modules of the distributed satellite task scheduling system

of each module and generate monitoring information, monitor operational faults, and be responsible for alarms.

- Resource data acquisition unit: its primary function is to obtain information on system operation parameters, earth observation requirements, satellite resources, ground station resources, measurement and control resources, satellite orbit data, etc. It is also used to calculate resource accessibility, including the time window for satellite access to the target, the time window for satellite download data to the ground station, and the time window for the measurement and control system to upload the commands to the satellite.
- Data preprocessing unit: it mainly consists of two parts: mission detection filtering and working mode reasoning. The mission detection filtering specifically detects

the legitimacy of the input mission information based on the expert system knowledge base. It filters the satellite earth observation missions and data transmission resources that do not meet the requirements (e.g., optical satellites cannot perform tasks at night). The working mode decision module is responsible for acquiring information about earth observation missions and data transmission resources and reasoning about the working mode of satellite payloads based on the rules in the domain knowledge base as shown in Fig. 7.5.

- Cooperative task scheduling unit: its primary function is to coordinate multiple single-satellite scheduling agents for scheduling and calculation based on the earth observation tasks, ground station resources, and other ground resources. The distributed collaborative scheduling algorithm decomposes tasks and resources into individual single-satellite scheduling agents, receives the earth observation programme generated by multiple single-satellite scheduling agents, and then evaluates and fuses multiple single-satellite earth observation programme to form a complete satellite earth observation solution.
- Multi-satellite constraint detection unit: it is responsible for detecting whether the use of satellites violates the set constraints.
- Multi-satellite conflict resolution module: it applies conflict resolution algorithms to resolve the conflicts between tasks.
- Multi-satellite knowledge base: it stores and manages the knowledge rules required in processes such as data preprocessing and multi-satellite collaborative scheduling. The expert system knowledge base contains the rule categories is shown in Fig. 7.6.

The collaborative scheduling agent of distributed satellite task scheduling system provides a display interface for users to view the status of scheduling calculations as shown in Fig. 7.7.

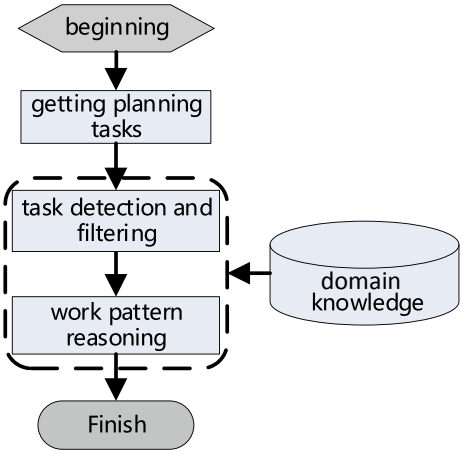


Fig. 7.5 Workflow of the data preprocessing unit

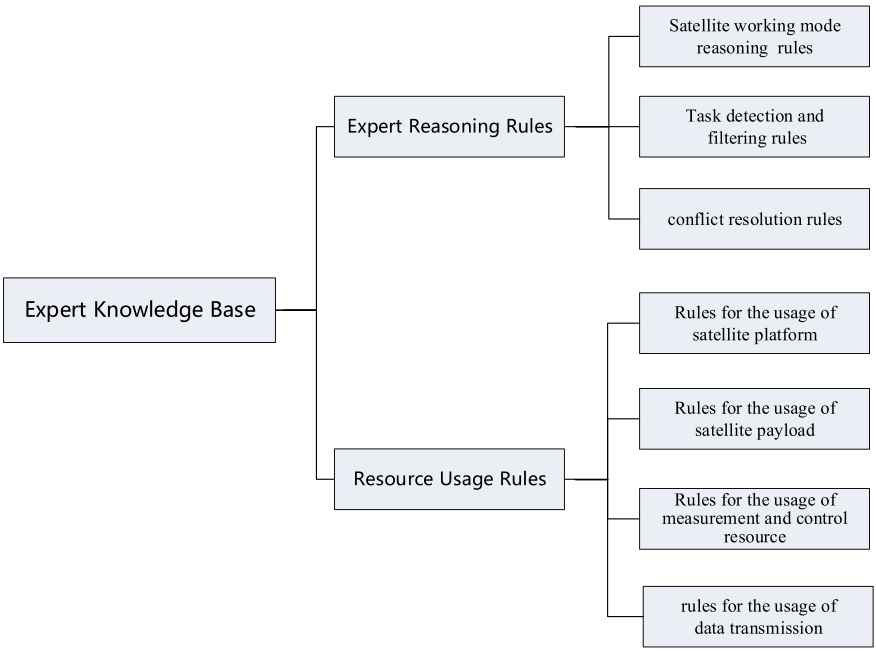


Fig. 7.6 Structure of the multi-satellite knowledge base

```
2015/9/22 11:22:39......start task planning.
2015/9/22 11:22:42......task planning completed.
2015/9/22 11:22:42......start resource planning.
2015/9/22 11:22:44......resource planning completed.
2015/9/22 11:22:44......start constraint detection and plan correction.
2015/9/22 11:22:45......constraint detection and plan correction completed.
2015/9/22 11:22:46......generating satellite activity sequence.
2015/9/22 11:22:46......finished.
```

Fig. 7.7 Interface of the collaborative scheduling agent

The single-satellite scheduling agent module completes single-satellite task scheduling under the organization of the collaborative scheduling agent module and submits the scheduling results to the collaborative scheduling agent module, whose submodules function as follows.

- Single-satellite task scheduling unit: according to the tasks and resources assigned by the collaborative scheduling agent, the single-satellite scheduling algorithm is invoked to perform scheduling calculations, generate a single-satellite earth observation programme, and send it to the collaborative scheduling agent.

```
start task planning
task planning period: 2015/9/23 00:00:00-2015/9/23 23:59:59
tender information received, target id: 14500010
winning the bid.
access time window: 196, 2015/9/23 01:36:17-2015/9/23 01:40:50
access time window: 199, 2015/9/23 22:20:42-2015/9/23 22:27:35

tender information received, target id: 14500011
winning the bid.
access time window: 105, 2015/9/23 18:17:20-2015/9/23 18:20:46
access time window: 106, 2015/9/23 19:59:01-2015/9/23 20:02:44

tender information received, target id: 14500012
missing the bid.

tender information received, target id: 14500013
winning the bid.
access time window: 131, 2015/9/23 15:41:53-2015/9/23 15:43:57

tender information received, target id: 14500014
missing the bid.
```

Fig. 7.8 Interface of the single-satellite scheduling agent

- Single-satellite constraint detection unit: it is used to detect the constraint conflict in the single-satellite earth observation programme based on the constraints of that satellite.
- Single-satellite resolution unit: it resolves the conflict in the single-satellite earth observation plan based on conflict-solving rules.
- Single-satellite knowledge base: it stores and manages the knowledge rules required in single-satellite scheduling calculations.

The single-satellite scheduling agent of the distributed satellite task scheduling system provides a display interface for users to view the status of this single-satellite scheduling agent module as shown in Fig. 7.8.

7.2.2 *Human–Computer Interaction Interface Design and Presentation*

The system provides users with several human–computer interactive interfaces for managing the satellite earth observation programme and maintaining the distributed satellite task scheduling system, such as programme viewing and editing, satellite resource management, knowledge base management.

1. Human–computer interface for viewing and editing of earth observation programme

The human–computer interface for viewing and editing provides three types of human–computer interaction: tables, electronic maps, and Gantt charts. The table primarily displays the attributes of the satellite earth observation meta-task, while the electronic map primarily shows the spatial distribution of the satellite earth observation meta-task. The Gantt chart primarily presents temporal relationship information of the earth observation meta-task.

(1) Tables

The table allows the details of each attribute element in the earth observation programme to be displayed or edited directly as shown in Fig. 7.9.

(2) Gantt chart

A Gantt chart visually represents the sequence and duration of specific activities using an activity list and a time scale. Gantt chart-based visual representations are designed for earth observation, data reception, and data transmission activity sequences.

The Gantt chart's advantage lies in its ability to visually represent the time-series correlation of earth observation tasks and resources. Users can adjust the satellite earth observation programme using the Gantt chart interface. Once the programme is modified, the scheduling system automatically performs constraint

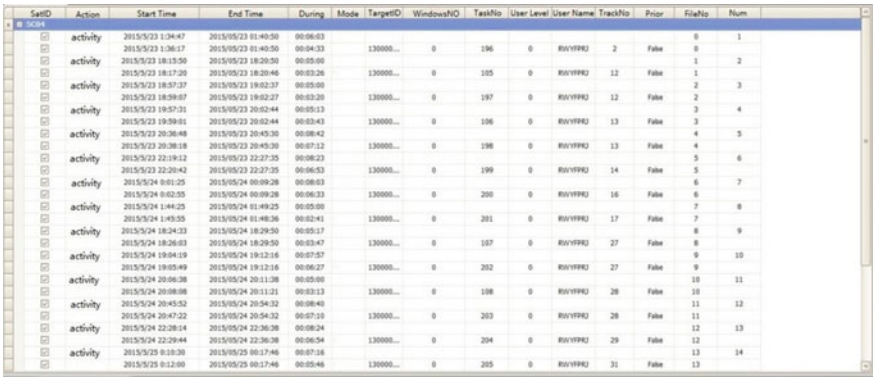


Fig. 7.9 Table for visualizing human–computer interaction of earth observation programme

detection on the adjusted scheduling results. If the adjustments violate the constraints, the scheduling system displays a constraint violation alert message and provides feasible adjustment suggestions.

However, the weakness of the Gantt chart visualization approach is that it does not effectively represent the relationship between the satellite and ground station or the spatial location of observation targets. To address this limitation, it is necessary to introduce an electronic map-based visual representation of scheduling results as a complementary approach to the Gantt chart visual representation.

(3) Maps

The electronic map displays the spatial relationship between satellite orbits and resources, such as targets and data transmission. The system visually represents scheduling results based on geographic information system (GIS) technology. The electronic map's base map is typically a high-precision Mercator projection of the world map. Planners can adjust the satellite earth observation programme on the electronic map, and the adjustment process is similar to that in the Gantt chart interface.

2. Human–computer interaction interface for satellite resource management

Users can complete the registration and logout functions of the single-satellite scheduling agent on the client side. The interface is shown in Fig. 7.10. Among them, the "sleep" state represents that the satellite scheduling agent has been deployed but has not formally entered the state of participating in scheduling calculation, and only the satellite scheduling agent in the "active" state can participate in scheduling process.

3. Knowledge base management human–computer interaction interface

The distributed satellite task scheduling system offers a human–computer interface for knowledge base management to maintain and manage the knowledge and rules

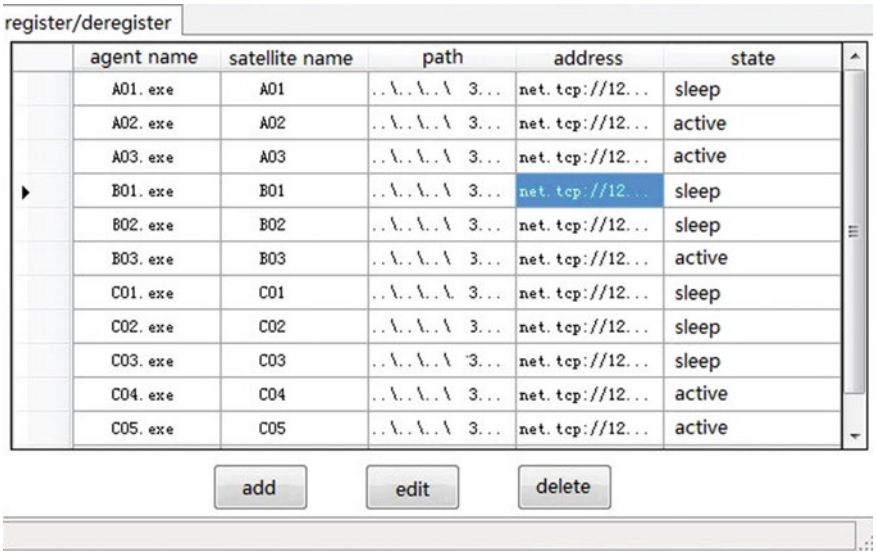


Fig. 7.10 Interface for the management of satellite agent status

required for data preprocessing and multi-satellite collaborative scheduling. This module includes adding, deleting, modifying, and retrieving rules and knowledge. It helps users understand the rules and facts the expert system uses and enables knowledge engineers to adapt the rules to actual needs.

Chapter 8
Summary and Prospect

With the rapid development of the global aerospace industry, the number and types of satellites in orbit and supporting resources are increasing, resulting in unprecedented attention to EOSs task scheduling in both academic and industrial circles. As a result, various new techniques and methods are emerging in this field. One of the main objectives of this book is to organize and present the current existing technologies and to explore future technological trends in this area.

8.1 Summary of This Book

This book presents the current research and practical progress in the field of earth observation satellite (EOS) task scheduling in a systematic order of "Problems (Chaps. 1 and 2)—Models and Algorithms (Chaps. 3–6)—Applications (Chap. 7)." The book aims to provide readers, whether a novice in the field, an academic deeply involved in the industry, or an experienced engineer, with a macro-understanding of the subject matter and the ability to select chapters of interest for further study.

Chapter 1 introduces the background and significance of EOS task scheduling and outlines the current development of various branches of EOS task scheduling. The chapter also briefly introduces the writing ideas and chapter composition of the book.

Chapter 2 provides a definition of the satellite task scheduling problem and introduces relevant concepts and terminology, as well as the challenges posed to planning and scheduling by the various elements of the EOS system from a planning and scheduling perspective. The purpose of this chapter is to present the issues and analyze the difficulties. Subsequent chapters address the satellite task scheduling problem in different scheduling scenarios.

H. Chen et al., *Earth Observation Satellites*,
https://doi.org/10.1007/978-981-99-3565-9_8

Chapter 3 describes the centralized task scheduling model and algorithm for satellites in static scenarios, which assumes that neither the earth observation tasks nor the satellite resources will not change once scheduling computation begins. This traditional and the most commonly used satellite task scheduling scenario is presented using evolutionary computation-based methods, while other meta-heuristic algorithms (e.g., tabu search algorithm, ant colony algorithm, simulated annealing algorithm, particle swarm algorithm, fireworks algorithm, etc.) follow a similar solution process.

Chapter 4 describes EOS task rescheduling methods in dynamic scenarios, which are used after a satellite earth observation programme has been developed, due to task changes or failure of satellite resources. In practice, such methods are often used for satellite earth observation task scheduling in emergency situations.

Chapter 5 describes the distributed satellite task scheduling method, which is one of the future development trends. Earth observation satellite task scheduling will gradually develop from a static offline centralized batch processing approach to a dynamic online distributed on-demand processing approach. This chapter establishes a distributed model for EOS task scheduling based on the idea of a multi-agent system and proposes scheduling algorithms based on contract network negotiation, blackboard model, and evolutionary computation, respectively. This type of method is still a hot research issue and has achieved initial results in practical application.

Chapter 6 introduces task scheduling models and algorithms for autonomous earth observation satellites with onboard processing capabilities. The scheduling algorithms based on graph theory and the sequential decision methods based on machine learning are introduced, and their application scenarios are analyzed. The development of autonomous satellites has received attention worldwide, but currently, they are mainly used in application scenarios where satellite earth communication is limited, such as deep space exploration, and the applications of autonomous earth observation satellites are still in their infancy. The methods presented in this chapter have good research value and application prospects.

Chapter 7 describes the architecture, main functions, and human–machine interfaces of some typical satellite task scheduling systems and aims to facilitate the application of the key technologies related to this book in practice.

8.2 Future Promising Technologies

In the author's opinion, the future of satellite task scheduling in both research and practical applications will be defined by the following trends.

8.2.1 Preference-Based Multi-objective Optimization for EOS Task Scheduling

In essence, the satellite task scheduling problem is a multi-objective optimization problem, which involves the optimization of multiple objectives such as task priority, task timeliness, and task completion. The ideal situation is to optimize all three objectives simultaneously. However, in practice, there is often a trade-off between these objectives. For example, increasing task priority may lead to a decrease in task completion, as more low-priority tasks may not be observed. Multi-objective optimization aims to provide a set of optimized solutions that are not dominated by each other, known as the set of non-dominated solutions or Pareto solutions. Ultimately, the decision-maker selects one or a small number of solutions as final solutions based on actual needs or user preferences.

Compared to single-objective optimization algorithms, multi-objective optimization algorithms are more computationally intensive and take longer computation time, making them less used in engineering practice and still at the theoretical research stage. However, if the decision-maker's potential preference information can be used to guide the algorithm's search direction, the algorithm can focus on generating non-dominated solutions that meet the decision-maker's preferences, thereby improving algorithm performance and reducing computation time. The main challenges in this process are modeling, extracting, and representing user preference information, and designing preference-based multi-objective optimization algorithms.

8.2.2 Multi-satellite Onboard Autonomous Cooperative Task Scheduling

In the context of earth observation systems, satellite onboard autonomous task scheduling is an important trend that enhances the system's rapid response capability. However, scheduling multiple satellites to work together presents more significant challenges compared to single-satellite autonomous task scheduling.

The first challenge arises from the need for stable and uninterrupted communication links between satellites, which are crucial for cooperative task scheduling. During the movement of satellites, the dynamic changes in the network topology of the satellite communication system can cause communication link instability and time delays, which can significantly impact the autonomous task cooperative scheduling process.

The second challenge involves the diversity of earth observation satellites, which exhibit varying constraints, working characteristics, and levels of intelligence. Heterogeneous satellite clusters require efficient and reasonable task assignment methods to members with different levels of intelligence, a considerable challenge for multi-satellite autonomous cooperative task scheduling.

Lastly, the complexity and variability of earth observation requirements put high demands on the robustness and efficiency of the multi-satellite autonomous cooperative task scheduling methods. Thus, the development of such methods still requires further research.

In conclusion, multi-satellite autonomous cooperative task scheduling is an essential area of research that requires a solution for the challenges posed by the dynamic communication network topology, heterogeneity of satellite clusters, and complexity of earth observation requirements.

8.2.3 Observation Task Cooperative Scheduling for Heterogeneous Platforms

This book is dedicated to exploring the theories, models, and methods associated with satellite task scheduling. With the rapid advancement of the aerospace and aviation industries, collaborative observation using air/space-based earth observation resources, such as earth observation satellites, unmanned aerial vehicles, and airships, has garnered increasing attention. While these resources operate and observe differently (see Table 8.1), they offer excellent complementarity. If we can seamlessly integrate satellites, UAVs, airships, and other air and space resources for earth observation, we will be able to further enhance the task completion rate, temporal resolution of observation data, and timeliness of such data, thereby maximizing the comprehensive benefits of these valuable resources. However, efficiently and optimally scheduling earth observation tasks to make full use of heterogeneous air and space-based earth observation resources remains a challenging issue.

8.2.4 Schedulability Prediction for Earth Observation Tasks

In the current multi-user satellite earth observation system, each user independently submits earth observation requirements to the satellite operation and control center. Each user has a specific number of high-, medium-, and low-priority slots for earth observation requirements. The operation and control center coordinates the scheduling of earth observation tasks based on the requirements submitted by users. However, due to the limited resources of earth observation satellites, only a portion of the tasks can be performed, which can result in potential competition among independent users. At the same time, if multiple users request observations of the same area, those observation requirements are combined, and the priority of that combined observation task is increased, creating a win–win situation for multiple users under certain conditions.

The existing satellite scheduling process is opaque to users, and they are only notified of the observation task acceptance (or rejection) after the scheduling computation

Table 8.1 Comparison of operational and observational characteristics of satellites, UAVs, and airships

Platform	Earth observation satellite	Unmanned aerial vehicle	Air ship
Payload resolution	High	Extremely high	High (decreases as roll angle increases)
Running mode	Orbiting as intended	Settable track	Settable track
Aerial hovering capability	Cannot	Weak	Strong
Revisiting observation capability	Revisiting observations by cycle	Circle around and perform continuous observation	Hover and stare
Movement speed	Rapid	Medium	Slow
Regional coverage capacity	Large	Medium	Small
Endurance	High	Low	Medium
Platform security	High	Medium–low	Medium

is complete. This results in users repeatedly modifying and resubmitting observation requirements through trial and error to ensure their interests are maximized, causing a significant computational burden on the satellite operation control center, and decreasing the overall efficiency of the satellite earth observation system.

Therefore, is it possible to find a method to predict quickly, inexpensively, and with high accuracy the subset of earth observation tasks that can be performed without performing scheduling computation? This can guide the user to propose more reasonable observation requirements, such as priority, temporal resolution, thereby increasing user participation in the scheduling process. Schedulability prediction of earth observation tasks can achieve this objective.

From the perspective of granularity, the earth observation task schedulability prediction can be divided into three levels: target-level forecast, satellite-level forecast, and earth observation meta-task-level forecast. The target-level forecast predicts whether an individual ground target can be observed. The satellite-level forecast determines which satellite observes which target and requires not only completing the target-level forecast but also considering the satellite that performs the observation. The earth observation meta-task-level forecast predicts whether an earth observation meta-task is performed directly, and it not only completes the satellite-level forecast but also predicts when a particular satellite will observe a specific target. If a high-accuracy earth observation meta-task-level forecast can be achieved, the prediction can be used as a complete earth observation programme, which can provide a high-quality initial solution for the scheduling algorithm or directly as a solution in an emergency.

The finer the prediction, the more challenging it becomes, as the schedulability of an earth observation task depends not only on the capabilities and constraints of the entire earth observation satellite system, but also on its geographical location,

priority, observation range requirements, observation time resolution requirements, observation urgency requirements, observation duration requirements, observation timeliness requirements, and other complex factors. Moreover, there is an implicit dependence on the direction of flight of the satellite and the presence of other high-priority targets in its vicinity on whether the task can be responded to. Thus, extracting the features from these complex elements and modeling the dependencies between them to improve the prediction accuracy is a challenging problem.

References

1. Hall N G, Magazine M J. Maximizing the Value of a Space Mission [J]. European Journal of Operation Research, 1994, 78: 224–241.
2. Lemaitre M, Verfaillie G, Jouhaud F, et al. Selecting and scheduling observations of agile satellites [J]. Aerospace Science and Technology, 2002, 6: 367–381.
3. Globus A, Crawford J, Lohn J. A Comparison of Techniques for Scheduling Earth Observing Satellites [C]. In the Proceedings of the 16th Conference on Innovative Applications of Artificial Intelligence, San Jose, USA, 2004.
4. Bianchessi N, Cordeau J F, Desrosiers J, et al. A Heuristic for the Multi-Satellite, Multi-Orbit and Multi-User Management of Earth Observation Satellites [J]. European Journal of Operational Research, 2007, 177: 750–762.
5. Jufang Li. Research on task scheduling of multi-satellite and multi-ground station [D]. National Defense University of Science and Technology, Changsha, Hunan, 2005. (In Chinese).
6. Jun Wang. Research on integrated task scheduling model and optimization method of imaging satellite [D]. National Defense University of Science and Technology, Changsha, Hunan, 2007. (In Chinese).
7. Xiaoshan Jin. Research on satellite ground integrated scheduling technology of imaging satellite [D]. National Defense University of Science and Technology, Changsha, Hunan, 2009. (In Chinese).
8. Kai Sun. Research on task scheduling model and optimization algorithm of agile Earth Observation Satellite [D]. National Defense University of Science and Technology, Changsha, Hunan, 2013. (In Chinese).
9. Wu K, Zhang D, Chen Z, et al. Multi-type multi-objective imaging scheduling method based on improved NSGA-III for satellite formation system [J]. Advances in Space Research, 2019, 63(8): 2551–2565.
10. Longmei Li. Preference based evolutionary multi-objective optimization and its application in satellite mission planning [D]. National Defense University of Science and Technology, Changsha, Hunan, 2018. (In Chinese).
11. Gabrel V, Vanderpooten D. Enumeration and Interactive Selection of Efficient Paths in a Multiple Criteria Graph for Scheduling an Earth Observing Satellite [J]. European Journal of Operational Research, 2002, 139: 533–542.
12. Fan Zhang. Research on constraint modeling and optimization solution technology in imaging satellite planning [D]. National Defense University of Science and Technology, Changsha, Hunan, 2005. (In Chinese).
13. Potter W. A Photo Album of Earth: Scheduling Landsat 7 Mission Daily Activities [C]. In the Proceedings of the 5th International Symposium on Space Mission Operations and Ground Data Systems, Tokyo, Japan, 1998.

H. Chen et al., *Earth Observation Satellites*,
https://doi.org/10.1007/978-981-99-3565-9

14. Yamaguchi Y, Kawakami T, Kahle A B, et al. ASTER Task scheduling and Operations Concept [C]. In the Proceedings of the 5th International Symposium on Space Mission Operations and Ground Data Systems, Tokyo, Japan, 1998.
15. Muraoka H, Cohen R H, Ohno T, Doi N. ASTER Observing Scheduling Algorithms [C]. In the Proceedings of the 5th International Symposium on Space Mission Operations and Ground Data Systems, Tokyo, Japan, 1998.
16. Chien S, Rabideau G, Knight R, et al. ASPEN - Automated Planning and Scheduling for Space Mission Operations [C]. In the Proceedings of the SpaceOps 2000, Toulouse, France, 2000.
17. Rabideau G, Knight R, Chien S, et al. Iterative Repair Planning for Spacecraft Operations in the ASPEN System [C]. In the Proceedings of 5th International symposium on Artificial Intelligence Robotics and Automation in Space, Noordwijk, Netherlands, 1999.
18. Frank J, Jonsson A, Morris R, et al. Planning and Scheduling for Fleets of Earth Observing Satellites [C]. In the Proceedings of the 6th International Symposium on Artificial Intelligence, Robotics, Automation and Space, Montreal, Canada, 2002.
19. Dungan J, Frank J, Jonsson A, et al. Advances in Planning and Scheduling of Remote Sensing Instruments for Fleets of Earth Orbiting Satellites [C]. In the Proceedings of the 2nd Earth Science Technology Conference, 2002.
20. Vasquez M, Hao J K. Upper Bounds for the SPOT 5 Daily Photograph Scheduling Problem [J]. Journal of Combinatorial Optimization, 2003, 7: 87–103.
21. Wolfe W J, Sorensen S E. Three Scheduling Algorithms Applied to the Earth Observing Systems Domain [J]. Management Science, 2000, 46(1): 148–168.
22. Lin W C, Liao D Y, Liu C Y, et al. Daily Imaging Scheduling of an Earth Observation Satellite [J]. IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans, 2005, 35(2): 213–223.
23. Renjie He. Research on imaging satellite scheduling problem [D]. National Defense University of Science and Technology, Changsha, Hunan, 2004. (In Chinese).
24. Yuhua Guo. Research on Key Technologies of integrated task planning of multiple types of Earth Observation Satellites [D]. National Defense University of Science and Technology, Changsha, Hunan, 2009. (In Chinese).
25. Wang P, Reinelt G, Gao P, et al. A model, a heuristic and a decision support system to solve the scheduling problem of an earth observing satellite constellation [J]. Computers & Industrial Engineering, 2011, 61: 322–335.
26. Chen X, Reinelt G, Dai G, et al. Priority-based and conflict-avoidance heuristics for multi-satellite scheduling [J]. Applied Soft Computing, 2018, 69: 177–191.
27. Chen H, Yang S, Li J, et al. Exact and heuristic methods for observing task-oriented satellite cluster agent team formation [J]. Mathematical Problems in Engineering, 2018.
28. Liu X, Laporte G, Chen Y, et al. An adaptive large neighborhood search metaheuristic for agile satellite scheduling with time-dependent transition time [J]. Computers & Operations Research, 2017, 86: 41–53.
29. He L, Liu X, Laporte G, et al. An improved adaptive large neighborhood search algorithm for multiple agile satellites scheduling [J]. Computers & Operations Research, 2018, 100: 12–25.
30. Lemaître M, Verfaillie G, Jouhaud F, et al. Selecting and scheduling observations of agile satellites [J]. Aerospace Science and Technology, 2002, 6(5): 367–381.
31. Cordeau J F, Laporte G. Maximizing the value of an earth observation satellite orbit [J]. Journal of the Operational Research Society, 2005, 56(8): 962–968.
32. Bianchessi N, Cordeau J F, Desrosiers J, et al. A heuristic for the multi-satellite, multi-orbit and multi-user management of earth observation satellites [J]. European Journal of Operational Research, 2007, 177(2): 750–762.
33. Tangpattanakul P, Jozefowiez N, Lopez P. A multi-objective local search heuristic for scheduling Earth observations taken by an agile satellite [J]. European Journal of Operational Research, 2015, 245(2): 542–554.
34. Wang P, Reinelt G, Gao P, et al. A model, a heuristic and a decision support system to solve the scheduling problem of an earth observing satellite constellation [J]. Computers & Industrial Engineering, 2011, 61(2): 322–335.

35. Jian Yang. Research on imaging scheduling method of Earth Observation Satellite Based on Area Target Decomposition [D]. National Defense University of Science and Technology, Changsha, Hunan, 2010. (In Chinese).
36. Xu Y, Liu X, He R, et al. Multi-satellite scheduling framework and algorithm for very large area observation [J]. Acta Astronautica, 2020, 167: 93–107.
37. Waiming Zhu. Multiple Earth Observation Satellites Cooperative Observing Area Covering Optimization Method [D]. Hefei Polytechnic University, Hefei, Anhui, 2019. (In Chinese).
38. Berry P E, Pontecorvo C, Fogg D. Optimal Search, Location and Tracking of Surface Maritime Targets by a Constellation of Surveillance Satellites [R]. DSTO-TR-1480, 2002.
39. Yuanzhuo Ci. Research on multi-satellite mission planning for moving target search [D]. National Defense University of Science and Technology, Changsha, Hunan, 2008. (In Chinese).
40. Pan Lu. Research on imaging satellite mission planning for marine moving targets [D]. National Defense University of Science and Technology, Changsha, Hunan, 2007. (In Chinese).
41. Yifan Xu, Yuejin Tan, Renjie He. Multi-model Prediction for Maritime Moving Target Motion [J]. Fire Control and Command Control, 2012, 37(3): 20–25. (In Chinese).
42. Hailong Zhang. Research on Multi-satellite Cooperative Search Method for Moving Target on Multi-obstacle Sea [D]. Hefei Polytechnic University, Hefei, Anhui, 2019. (In Chinese).
43. Bo Yuan. Research on ocean moving target analysis method for satellite resource scheduling [D]. National Defense University of Science and Technology, Changsha, Hunan, 2010. (In Chinese).
44. Li J F, Yao F, et al. Using multiple satellites to search for maritime moving targets based on reinforcement Learning [J]. Journal of Donghua University (English Version), 2016, 33(5): 749–754.
45. Guanlin Mei, Xiaomin Ran, Liang Fan, et al. Research on Satellite Sensor Scheduling Technology for Moving Target [J]. Journal of Information Engineering University, 2016, 17(5): 513–517. (In Chinese).
46. Pemberton J C, Greenwald L G. On the Need for Dynamic Scheduling of Imaging Satellites [C]. In the Proceedings of Pecora 15/Land Satellite Information IV/ISPRS Commission I/ FIEOS 2002 Conference, Colorado, USA, 2002.
47. Verfaillie G, Jussien N. Constraint Solving in Uncertain and Dynamic Environments: A Survey [J]. Journal of Constraints, 2005, 10: 253–281.
48. Verfaillie G, Schiex T. Solution Reuse in Dynamic Constraint Satisfaction Problems [C]. In the Proceedings of the 12th Conference of the American Association of Artificial Intelligence, Seattle, USA, 1994.
49. Verfaillie G, Bensana E, et al. Dealing with Uncertainty when Managing an Earth Observation Satellite [C]. In the Proceedings of the 5th International Symposium on Artificial Intelligence, Robotics, and Automation for Space, Noordwijk, Netherlands, 1999.
50. Liao D Y, Yang Y T. Satellite Imaging Order Scheduling with Stochastic Weather Condition Forecast [C]. In the Proceedings of the 2005 IEEE International Conference on Systems, Man and Cybernetics, 2005.
51. Billups S C. Satellite Mission Scheduling with Dynamic Tasking [R]. Final Report of the UCDHSC Mathematics Clinic, 2005.
52. Yang Liu. Research on Dynamic Rescheduling model, algorithm and application of imaging satellite [D]. National Defense University of Science and Technology, Changsha, Hunan, 2004. (In Chinese).
53. Kramer L A, Smith S F. Task Swapping for Schedule Improvement: A Broader Analysis [C]. In the Proceedings of the 14th International Conference on Automated Planning and Scheduling, Whistler, Canada, 2004.
54. Kramer L A, Smith S F. Task Swapping: Making Space in Schedules for Space [C]. In the Proceedings of the 4th International Workshop on Planning and Scheduling for Space, Darmstadt, Germany, 2004.

55. Lining Zhang, Xiaojun Huang, Dishan Qiu, et al. Heuristic dynamic adjust of task scheduling for earth observing satellite [J]. Computer Engineering and Applications, 2011, 47(30): 241–245. (In Chinese).
56. Wang J, Zhu X, Qiu D, et al. Dynamic Scheduling for Emergency Tasks on Distributed Imaging Satellites with Task Merging [J]. IEEE Transactions on Parallel and Distributed Systems, 2014, 25(9): 2275–2285.
57. Wang J, Demeulemeester E, Qiu D. A pure proactive scheduling algorithm for multiple earth observation satellites under uncertainties of clouds [J]. Computers & Operations Research, 2016, 74: 1–13.
58. Wang MC, Dai GM, Vasile M. Heuristic Scheduling Algorithm Oriented Dynamic Tasks for Imaging Satellites [J]. Mathematical Problems in Engineering, 2014.
59. Ping Jian, Peng Zou, Wei Xiong. Analyze on Task Planning of Early Warning System of LEO Under Dynamic Disturbances [J]. Acta Electronica Sinica, 2014, 42(10): 1894–1900. (In Chinese).
60. Jianyin Liu. The Research on Multi-satellite Scheduling Method Oriented to Forest Resource Observation [D]. Central South University of forestry Science and Technology, Changsha, Hunan, 2018. (In Chinese).
61. Ming Zhang, Bo Wei, Jindong Wang. Satellite Reactive Scheduling Based on Heuristic Algorithm [J]. Computer Science, 2019. (In Chinese).
62. Haiquan Sun, Wei Xia, Xiaoxuan Hu, et al. Earth observation satellite scheduling for emergency tasks [J]. Journal of Systems Engineering and Electronics, 2019, 30(5): 931–945.
63. Torreño A, Onaindia E, Sapena Ó. An approach to multi-agent planning with incomplete information [J]. arXiv preprint arXiv:1501.07256, 2015.
64. Ni Gao. Research on Task Cooperation of Earth Observation Distributed Satellite System [D]. National Defense University of Science and Technology, Changsha, Hunan, 2007. (In Chinese).
65. Wang C, Li J, Jing N, et al. A Distributed Cooperative Dynamic Task Planning Algorithm for Multiple Satellites Based on Multi-agent Hybrid Learning [J]. Chinese Journal of Aeronautics, 2011; 24(4): 493–505.
66. Chong Wang, Ning Jing, Jun Li, et al. An Algorithm of Cooperative Multiple Satellites Mission Planning Based on Multi-agent Reinforcement Learning [J]. Journal of National University of Defense Technology, 2011, 33(1): 53–58. (In Chinese).
67. Jun L, Ning J, Weidong H, et al. A Multi-platform Sensor Coordinated Earth Observing Missions Scheduling Method for Hazard Monitoring [C]. 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), 2013: 554–560.
68. Feng P, Chen H, Peng S, et al. A method of distributed multi-satellite mission scheduling based on improved contract net protocol [C]. 11th IEEE International Conference on Natural Computation (ICNC), 2015: 1062–1068.
69. Bonnet J, Gleizes M P, Kaddoum E, et al. Multi-satellite Task scheduling Using a Self-Adaptive Multi-agent System [C]. IEEE International Conference on Self-Adaptive and Self-Organizing Systems, 2015: 11–20.
70. Du B, Li S. A new multi-satellite autonomous mission allocation and planning method [J]. Acta Astronautica, 2018.
71. Zheng, Z., J. Guo, and E. Gill, Onboard mission allocation for multi-satellite system in limited communication environment [J]. Aerospace Science and Technology, 2018. 79: 174–186.
72. Pingyuan Cui, Rui Xu. State of the Art and Development Trends of On-board Autonomy Technology for Deep Space Explorer [J]. Acta Aeronautica et Astronautica Sinica, 2014 (01): 13–28. (In Chinese).
73. Bernard DE, Dorais GA, Fry C, et al. Design of the Remote Agent Experiment for Spacecraft Autonomy [C]. IEEE Aerospace Conference, 1998: 259–281.
74. Muscettola N, Fry C, Rajan K, et al. On-board planning for New Millennium Deep Space One autonomy [C]. IEEE Aerospace Conference, 1997, 1: 303–318.
75. Chien S, Sherwood R, Tran D, et al. The EO-1 Autonomous Science Agent [C]. IEEE International Joint Conference on Autonomous Agents & Multiagent Systems, 2004: 420–427.

76. Chien S, Knight R, Stechert A, et al. Using Iterative Repair to Improve the Responsiveness of Planning and Scheduling [C]. International Conference on Artificial Intelligence Planning Systems, 2000: 300–307.
77. Beaumet G, Verfaillie G, Charmeau M. Feasibility of autonomous decision making on board an agile earth-observing satellite [J]. Computational Intelligence, 2011, 27(1): 123–139.
78. Enying Zhang. The Research on Model and Algorithm of Autonomous Mission Planning for Multi Imaging Satellites [D]. Harbin Institute of Technology, Harbin, Heilongjiang, 2017. (In Chinese).
79. Rui Miao. Research On Autonomous Task Planning of Imaging Satellite of Formation Flying [D]. Harbin Institute of Technology, Harbin, Heilongjiang, 2016. (In Chinese).
80. Zhijia Xue, Zhong Yang, Jing Li, et al. Autonomous Mission Planning of Satellite for Emergency [J]. Command Control and Simulation, 2015(01): 24–30. (In Chinese).
81. Knight S, Rabideau G, Chien S, et al. Casper: space exploration through continuous planning [J]. IEEE Intelligent Systems, 2001, 16(5): 70–75.
82. Chien S, Knight R, Stechert A, et al. Integrated planning and execution for autonomous spacecraft [C]. IEEE Aerospace Conference, 1999: 263–271.
83. Chien S, Wichman S, Engelhart B. Onboard Autonomy Software on the Three Corner Sat Mission [C]. International Symposium on Artificial Intelligence Robotics & Automation, 2002: 1–7.
84. Sherwood R, Chien S, Castano R, et al. The Thinking Spacecraft: Autonomous Operations through Onboard AI [C]. SpaceOps Conference, 2006: 1–10.
85. Chien S, Sherwood R, Burl M, et al. A Demonstration of Robust Planning and Scheduling in the Techsat-21 Autonomous Sciencecraft Constellation [J]. Ear Nose & Throat Journal, 2014, 86(8): 506–511.
86. Chien, S, Joshua, D, Tran, D. Onboard Task scheduling on the Intelligent Payload Experiment (IPEX) Cubesat Mission. International Workshop on Planning and Scheduling for Space, 2013: 1–5.
87. Chien S, Bue B, Castillorogez J, et al. Agile Science for Primitive Bodies and Deep Space Exploration [C]. International Conference on Space Operations, 2014: 1–7.
88. Chien S, Rabideau G, Tran D, et al. Scheduling Science Campaigns for the Rosetta Mission: A Preliminary Report [J]. International Workshop on Planning and Scheduling for Space, 2013: 1–8.
89. Teston F, Creasey R, Bermyn J, et al. Proba: ESA's Autonomy and Technology Demonstration Mission [J]. 13th AIAA/USU Conference on Small Satellites, 1999: 1–11.
90. Bermyn J. PROBA-project for on-board autonomy [J]. Air & Space Europe, 2000, 2(1): 70–76.
91. Damiani S, Verfaillie G, Charmeau M-C. A continuous anytime planning module for an autonomous earth watching satellite [C]. International Conference on Automated Planning and Scheduling, 2005: 19–28.
92. Verfaillie G, Charmeau MC. A generic modular architecture for the control of an autonomous spacecraft [C]. 5th International Workshop on Planning and Scheduling for Space, 2006: 1–9.
93. Maillard A, Pralet C. Ground and Onboard Decision-Making on Satellite Data Downloads [C]. International Conference on Automated Planning and Scheduling, 2015: 273–281.
94. Verfaillie A M, Pralet C. Postponing Decision-Making to Deal with Resource Uncertainty on Earth-Observation Satellites [C]. 9th International Workshop on Planning and Scheduling for Space, 2015: 1–8.
95. Pralet C, Verfaillie G. Decision upon observations and data downloads by an autonomous Earth surveillance satellite [C]. 9th International Symposium on Artificial Intelligence, Robotics, and Automation in Space, 2008: 1–8.
96. Wille B, Worle M T, Lenzen C. VAMOS-Verification of Autonomous Task scheduling Onboard a Spacecraft [C]. 19th IFAC Symposium on Automatic Control in Aerospace, 2013: 382–387.
97. Li C, Causmaecker P D, Chen Y W. Data-driven Onboard Scheduling for an Autonomous Observation Satellite [C]. Twenty-Seventh International Joint Conference on Artificial Intelligence, 2018: 5773–5774.

98. Xiaoli Liu, Bin Yang, Zhaohui Gao, et al. Research on Rolling Dynamic Mission Scheduling Technique for Remote Sensing Satellites [J]. Radio Engineering of China, 2017, 47(09): 68–72. (In Chinese).
99. Yingwu Chen, Lining Xing, et al. Autonomous mission planning method of imaging satellite for dynamic environment [P]. Chinese patent: 105095643, 2015-11-25. (In Chinese).
100. Chuan He, Dishan Qiu, Xiaomin Zhu, et al. Emergency scheduling method for imaging reconnaissance satellites based on rolling horizon optimization strategy [J]. Systems Engineering-Theory and Practice, 2013, 33(10): 2685–2694. (In Chinese).
101. Ting Xi, Jufang Li. The Rolling Horizon Method for EOS Scheduling with Dynamic Requests [J]. Chinese Journal of Management Science, 2015, 11(23): 269–274. (In Chinese).
102. Song Liu, Yingwu Chen, Lining Xing, et al. Method of agile imaging satellites autonomous task planning [J]. Computer Integrated Manufacturing Systems, 2016, 4(22): 928–934. (In Chinese).
103. Li G, Xing L, Chen Y. A hybrid online scheduling mechanism with revision and progressive techniques for autonomous Earth observation satellite [J]. Acta Astronautica, 2017, 140: 308–321.
104. He L, Liu X L, Chen Y W, et al. Hierarchical scheduling for real-time agile satellite task scheduling in a dynamic environment [J]. Advances in Space Research, 2018, 63(2): 897–912.
105. Hao Chen, Jun Li, Ning Jing, et al. Scheduling Model and Algorithms for Autonomous Electromagnetic Detection Satellites [J]. Acta Aeronautica Et Astronautica Sinica, 2010, 31(5): 1045–1053. (In Chinese).
106. Chu X, Chen Y, Tan Y. An anytime branch and bound algorithm for agile earth observation satellite onboard scheduling [J]. Advances in Space Research, 2017, 60(9): 2077–2090.
107. Chen Wang. Satellite Observing Mission Scheduling Method Based on Case-based Learning [D]. National Defense University of Science and Technology, Changsha, Hunan, 2016. (In Chinese).
108. Wang H J, Yang Z, Zhou W G, et al. Online scheduling of image satellites based on neural networks and deep reinforcement learning [J]. Chinese Journal of Aeronautics, 2019, 32(4): 1011–1019.
109. Peng S, Chen H, Du C, et al. Onboard Observation Task Planning for an Autonomous Earth Observation Satellite Using Long Short-Term Memory, IEEE Access, 2018, (6): 65118–65129.
110. Barbulescu L, Howe A, Whitley D. AFSCN scheduling: How the problem and solution have evolved [J]. Mathematical and Computer Modelling, 2006, 43(9): 1023–1037.
111. Clement B J, Johnston M D. The Deep Space Network Scheduling Problem [C]. In the Proceedings of the 17th Innovative Applications of Artificial Intelligence Conference, Pittsburgh, USA, 2005.
112. Guang Jin, Xiaoyue Wu, Weibin Gao. Conflict Based Resource Scheduling and Capability Analysis of Satellite - Ground Station System [J]. Journal of Chinese Mini-Micro Computer Systems, 2007, 28(2): 310–312. (In Chinese).
113. Xiaoguang Wu. Satellite Data Transmission Mode Decision and Transmission Resource Allocation Method Oriented Emergency Observation [D]. National Defense University of Science and Technology, Changsha, Hunan, 2013. (In Chinese).
114. H Chen, Z Zhong, J Wu, N Jing. Multi-satellite data downlink resource scheduling algorithm for incremental observation tasks based on evolutionary computation [C]. IEEE 7th International Conference on Advanced Computational Intelligence, Wuyi Mountain, China, 2015.
115. H Chen, Y Zhou, C Du, et al. A satellite cluster data transmission scheduling method based on genetic algorithm with rote learning operator [C]. 2016 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2016: 5076–5083.
116. H Chen, L Li, Z Zhong, J Li. Approach for earth observation satellite real-time and playback data transmission scheduling [J]. Journal of Systems Engineering and Electronics 2015; 26(5): 982–992.

117. Xhafa F, Herrero X, Barolli A, Takizawa M. A Hill Climbing Algorithm for Ground Station Scheduling [J]. Information Technology Convergence, 2013.
118. Xhafa F, Herrero X, Barolli A, Takizawa M. A Simulated Annealing Algorithm for Ground Station Scheduling Problem [C]. 16th International Conference on Network-Based Information Systems, 2013.
119. Xhafa F, Herrero X, Barolli A, Barolli L, Takizawa M. Evaluation of struggle strategy in Genetic Algorithms for ground stations scheduling problem [J]. Journal of Computer and System Sciences 2013: 1086–1100.
120. Song Y, Zhang Z, Sun K, et al. A Heuristic Genetic Algorithm for Area targets' Small Satellite Image Downlink Scheduling Problem [J]. International Journal of Aerospace Engineering, 2019, 2019.
121. Du Y, Xing L, Zhang J, et al. MOEA based memetic algorithms for multi-objective satellite range scheduling problem [J]. Swarm and Evolutionary Computation, 2019, 50: 100576.
122. Zhang J, Xing L, Peng G, et al. A large-scale multiobjective satellite data transmission scheduling algorithm based on SVM+NSGA-II [J]. Swarm and Evolutionary Computation, 2019, 50: 1–10.
123. Chen H, Zhai B, Wu J, et al. A Satellite Observation Data Transmission Scheduling Algorithm Oriented to Data Topics [J]. International Journal of Aerospace Engineering, 2020, 2020.
124. Yunfeng Li. Research on satellite-ground station data transmission scheduling model and algorithm [D]. National Defense University of Science and Technology, Changsha, Hunan, 2007. (In Chinese).
125. Hao Chen. Planning and Scheduling Method for the Earth's Surface Electromagnetic Environment Detection Satellite Resources [D]. National Defense University of Science and Technology, Changsha, Hunan, 2009. (In Chinese).
126. Ying Du. Research on application technology of expert system for satellite mission planning [D]. National Defense University of Science and Technology, Changsha, Hunan, 2008. (In Chinese).
127. MacCormick J. What Can Be Computed?: A Practical Guide to the Theory of Computation [M]. Princeton University Press, 2018.
128. Bozorg-Haddad O, Solgi M, Loáiciga H A. Meta-heuristic and evolutionary algorithms for engineering optimization [M]. John Wiley & Sons, 2017.
129. C.A. Coello Coello. Theoretical and Numerical Constraint-Handling Techniques Used with Evolutionary Algorithms: A Survey of the State of the Art [J]. Comput. Methods in Appl. Mech. Eng, 2002(191): 1245–1287.
130. Janet Kolodner. Case-Based Reasoning [M]//Case-based reasoning. Morgan Kaufmann Publishers Inc. 1993.
131. Wang J, Zhu X, Yang L T, et al. Towards dynamic real-time scheduling for multiple earth observation satellites [J]. Journal of Computer and System Sciences, 2015, 81(1): 110–124.
132. Kramer L A, Smith S F. Maximizing availability: A commitment heuristic for oversubscribed scheduling problems [C]. Proceedings of the International Conference on Automated Planning and Scheduling, 2005: 272–280.
133. Junmin Wang, Jufang Li, Yuejin Tan. Research on Model and Algorithm of Multi-Satellite Dynamic Scheduling with New Tasks Insertion [J]. Journal of System Simulation, 2009, 21(12): 3522–3527. (In Chinese).
134. Joslin E D, Clements P D. "Squeaky Wheel" Optimization [C]. In Proceedings of the 15th National Conference on Artificial Intelligence, Madison, USA, 1998.
135. Jun Li. Coordinated Task Planning Research of Space-aeronautics Earth-Observing [D]. National Defense University of Science and Technology, Changsha, Hunan, 2013. (In Chinese).
136. Kai Chen. Research on distributed mission planning model and algorithm of remote sensing satellite [D]. National Defense University of Science and Technology, Changsha, Hunan, 2011. (In Chinese).
137. Chong Wang. Research on Distributed Collaborative Mission Planning of Earth Observation Satellites Based on Agent [D]. National Defense University of Science and Technology, Changsha, Hunan, 2011. (In Chinese).

138. McCarthy J, Hayes P. Some Philosophical Problems from the Standpoint of Artificial Intelligence [M]. Edinburgh: Edinburgh University Press, 1969.
139. Wooldridge M, Jennings N. Intelligent Agents: Theory and Practice [J]. The Knowledge Engineering Review, 1995, 10(2): 115–152.
140. Debugging M. Multi-Agent System [J]. Information and Software Technology, 1995, 37(2): 102–112.
141. Zhongzhi Shi. Advanced Artificial Intelligence [M]. Beijing: Science Press, 1998. (In Chinese).
142. Zhengqiang Zhang. Research on task planning and control of distributed imaging satellite system based on multi-agent system [D]. National Defense University of Science and Technology, Changsha, Hunan, 2006. (In Chinese).
143. Minsky M, Riecken D. A Conversation with Marvin Minsky about Agent [M]. Communication of the ACM, 1994, 37(7): 23–29.
144. Simon H A. Models of Bounded Rationality [M]. London: MIT Press, 1982–1997, 1–3.
145. Minsky M. The Society of Mind [M]. New York: Simon & Schuster, 1985.
146. Schetter T, Campbell M, Surka D. Multiple Agent-Based Autonomy for Satellite Constellations [J]. Artificial Intelligence, 2003(145): 147–180.
147. Smith G, Davis R. Frameworks for Cooperation in Distributed Problem Solving [J]. IEEE Transactions on Systems, Man and Cybernetics, 1981, 11(1): 61–70.
148. Zheng Xiao, Chengrong Wu, Shiyong Zhang. A Survey of Cooperation and Coordination in Multi-agent System [J]. Computer Science, 2007, 34(5): 139–143. (In Chinese).
149. Hao Chen, Ning Jing, Jun Li, et al. An Approach for Autonomous Electromagnetic Detection Satellite Constellation Scheduling Based on Outsourcing Contract Net [J]. Journal of Astronautics, 2009, 30(06): 2285–2291. (In Chinese).
150. Merton R K. The Matthew effect in science [J]. Science, 1968, 159(3810): 56–63.
151. Merton R K. The Matthew effect in science, II: cumulative advantage and the symbolism of intellectual property. ISIS, 1988, 79(4): 606–623.
152. Badawy R, Hirsch B, Albayrak S. Agent-based coordination techniques for matching supply and demand in energy networks [J]. Integrated Computer Aided Engineering, 2010, 17(4): 373–382.
153. Peng Feng. The Study on Key Technologies of Multi-Satellite Mission Scheduling and Scheme Fusion Based on VMOC [D]. National Defense University of Science and Technology, Changsha, Hunan, 2015. (In Chinese).
154. Li Yao, Weiming Zhang, et al. Intelligent Collaborative Information Technology [M]. Beijing: Electronic Industry Press, 2002, 4. (In Chinese).
155. Kerong Ben, Yanduo Zhang. Artificial Intelligence [M]. Beijing: Tsinghua University Press, 2006. (In Chinese).
156. Chien S, Cichy B, Jones J, et al. An Autonomous Earth-Observing SensorWeb [J]. IEEE Intelligent Systems, 2005, 20(3): 16–24.
157. Song Liu. Autonomous Planning for Agile Earth Observing Satellite Integrating Task and Action [D]. National Defense University of Science and Technology, Changsha, Hunan, 2017. (In Chinese).
158. Weiwu Hu. Development Road of Self-Developed CPU and Application in Aerospace Field [J]. Aerospace Shanghai, 2019, 36(01): 1–9. (In Chinese).
159. Haddad N F, Brown R D, Ferguson R, et al. Second generation (200 MHz) RAD750 microprocessor radiation evaluation [C]. European Conference on Radiation & Its Effects on Components & Systems, 2012: 877–880.
160. Guerriero F, Musmanno R. Label Correcting Methods to Solve Multicriteria Shortest Path Problems [J]. Journal of Optimization Theory and Applications, 2001, 111(3): 589–613.
161. Matthias H, Karsten W. Pareto Shortest Path is Often Feasible in Practice [C]. In the Proceedings of the 5th International workshop of Algorithm Engineering, Aarhus, Denmark, 2001.
162. Zhihua Zhou. Machine Learn [M]. Beijing: Tsinghua University Press, 2016. (In Chinese).
163. Wolpert D H. Stacked Generalization [J]. Neural Networks, 1992, 5(2): 241–259.

164. Lecun Y, Bengio Y, Hinton G. Deep learning [J]. Nature, 2015, 521(7553): 436.
165. Junyang Zhang, Huili Wang, Yang Guo, et al. Review of deep learning [J]. Application Research of Computers, 2018, 35(07): 1921–1928+1936. (In Chinese).
166. Schmidhuber, J. Deep learning in neural networks: An overview [J]. Neural Networks, 2015, 61: 85–117.
167. Goodfellow I, Bengio Y, Courville A. Deep learning [M]. MIT Press, 2016.
168. Jin KH, Mccann MT, Froustey E, et al. Deep Convolutional Neural Network for Inverse Problems in Imaging [J]. IEEE Transactions on Image Processing, 2017, 26(9): 4509–4522.
169. Zhang H, Ying L, Zhang Y, et al. Spectral-spatial classification of hyperspectral imagery using a dual-channel convolutional neural network [J]. Remote Sensing Letters, 2017, 8(5): 438–447.
170. Feng W, Guan N, Li Y, et al. Audio visual speech recognition with multimodal recurrent neural networks [C]. 2017 International Joint Conference on Neural Networks, 2017: 681–688.
171. Hayashi Y, Yanagimoto H. Title Generation with Recurrent Neural Network [C]. 5th IIAI International Congress on Advanced Applied Informatics, 2016.
172. Hochreiter S, Schmidhuber J. Long Short-Term Memory [J]. Neural Computation, 1997, 9(8): 1735–1780.
173. Kingma D P, Ba J L. Adam: A Method for Stochastic Optimization [C]. International Conference on Learning Representations, 2015: 1–13.
174. Fukunaga A, Rabideau G, Chien S, et al. Aspen: A framework for automated planning and scheduling of spacecraft control and operations [C]//Proc. International Symposium on AI, Robotics and Automation in Space. 1997.
175. Kianzad V, Bhattacharyya S S, Qu G. CASPER: An integrated energy-driven approach for task graph scheduling on distributed embedded systems [C]//2005 IEEE International Conference on Application-Specific Systems, Architecture Processors (ASAP'05). IEEE, 2005: 191–197.
176. Arentoft M M, Parrod Y, Stader J, et al. OPTIMUM-AIV: A planning and scheduling system for spacecraft AIV [J]. Telematics and Informatics, 1991, 8(4): 239–252.
177. Damiani S, Dreihahn H, Noll J, et al. A planning and scheduling system to allocate ESA ground station network services [C]//The Int'l Conference on Automated Planning and Scheduling. 2007.
178. Xhafa F, Herrero X, Barolli A, et al. Using STK toolkit for evaluating a GA base algorithm for ground station scheduling [C]//2013 Seventh International Conference on Complex, Intelligent, and Software Intensive Systems. IEEE, 2013: 265–273.
179. Chi Zhang. Robust Resource Scheduling Method and Its Application in Satellite Task Planning [D]. National Defense University of Science and Technology, Changsha, Hunan, 2010. (In Chinese).
180. Shuang Peng. Multi-Satellite Collaborative Planning and Activity Sequence Adjusting for Emergency Observation Missions [D]. National Defense University of Science and Technology, Changsha, Hunan, 2014. (In Chinese).