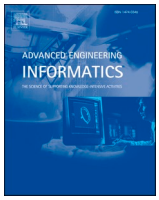
Advanced Engineering Informatics 60 (2024) 102362

Contents lists available at ScienceDirect

Advanced Engineering Informatics

journal homepage: www.elsevier.com/locate/aei



Multi-satellite cooperative scheduling method for large-scale tasks based on hybrid graph neural network and metaheuristic algorithm

Xiaoen Feng, Yuqing Li *, Minqiang Xu

Deep Space Exploration Research Center, Harbin Institute of Technology, Harbin 150001, China

ARTICLE INFO

Keywords:
Earth observation satellites
Large-scale satellite missions
Multi-satellite scheduling
Graph neural network
Metaheuristic algorithm

ABSTRACT

Traditional heuristic optimization algorithms are no longer applicable to the multiple satellites scheduling with large-scale tasks, as they are unable to provide satisfactory performance in terms of convergence and speed when addressing complex constraint conflicts. Therefore, we propose a multi-satellite cooperative scheduling method for large-scale tasks based on a hybrid graph neural network (GNN) and metaheuristic algorithm. By using the representation and extraction capability of GNN for relations in graph, the features of large-scale tasks and their constraint relations were expressed, and the generalized knowledge was extracted. And combined with metaheuristic algorithm, the optimization framework for large-scale satellite task scheduling based knowledge and data was implemented. The method consists of a GNN pre-solver module for static constraint conflicts and a metaheuristic optimization module for dynamic constraint conflicts of satellite missions. In the GNN pre-solver module, the graph sample and aggregate network was used to learn and extract the common knowledge in the static task-conflict graph, and provided effective prior information for the metaheuristic optimization module. The quadratic unconstrained binary optimization loss function was designed with multiple influencing factors, to convert the discrete optimization function into a continuous function; a greedy threshold was also added to improve the task completion rate. Finally, numerical experimental results showed that the proposed method can achieve an efficient solution to the multi-satellite scheduling problem with tens of thousands tasks. Compared with the commonly used multi-satellite scheduling algorithms (the genetic algorithm (GA) and greedy-GA), the proposed method can obtain higher-quality solutions under the same conditions and greatly improve the computational efficiency of large-scale mission planning.

1. Introduction

Earth observation satellites (EOSs) play an important role in various fields, such as the military, economic, and civil society sectors, owing to their wide coverage without airspace boundary constraints [1,2]. The scale of satellite missions is almost increasing exponentially with the demand for information. Consequently, the effective use of satellites to complete more observation tasks is critical to the success of earth observation missions [3,4]. The satellite imaging-task scheduling problem is a crucial component of satellite scheduling, particularly for future large-scale tasks, and constitutes the focus of this study.

The earth observation satellite scheduling problem (EOSSP) is widely known as a nondeterministic polynomial (NP) hard problem with complex constraints [5]. Considerable research has been conducted to solve it [6], wherein the algorithms used usually being classified as exact or heuristic [7,8]. Exact methods, such as branch and bound [9] and mixed-integer linear programming (MILP) [10], have deterministic solving steps and can solve problems in polynomial time in most instances; however, building a deterministic model is practically impossible for larger-scale problems. Heuristic methods can be employed to accelerate the process by searching for a satisfactory solution; however, these methods depend on specific heuristic strategies.

Chen et al. [11] carefully analyzed the interdependency between feasible time intervals to describe the constraints and formulated a multi-satellite scheduling problem as a MILP model. He et al. [12] proposed an edge-computing framework with a rolling mechanism and a heuristic algorithm based on the density of residual tasks, to solve a multiple agile EOSs scheduling problem. These heuristic methods are efficient in obtaining optimum solutions quickly, but they struggle with large-scale problems.

Without depending on problem-specific heuristic strategies, metaheuristic algorithms guided by a set of search strategies are perceived as

* Corresponding author.
E-mail address: bradley@hit.edu.cn (Y. Li).

https://doi.org/10.1016/j.aei.2024.102362
Received 23 August 2023; Received in revised form 22 December 2023; Accepted 12 January 2024
Available online 25 January 2024

Table 1
Symbol definitions.

Symbol	Definition
$SAT = \{sat_1, sat_2, ..., sat_m, ..., sat_M\}$ $m = 1, 2, ..., M$	Satellite collection, where the total number of satellites is M.
$TAR = \{tar_1, tar_2, ..., tar_n, ..., tar_N\}$ $n = 1, 2, ..., N$	Target collection, where the total number of targets is N.
$TASK = \{task_1, task_2, ..., task_k, ..., task_K\}$ $k = 1, 2, ..., K$	Collection of all observation tasks, where the total number of tasks is K.
$WinST_k^{mn}, WinET_k^{mn}$	Start time and end time of $task_k$ when sat_m is observing tar_n.
dur_k^{mn}	Duration of $task_k$ when sat_i is observing tar_n.
pri_k^{mn}	Priority level of $task_k$ when the tar_n is observed by sat_m, which represents the reward earned by completing it.
$tran_{ij}^{m}$	Transition time required between any two consecutive observation tasks $task_i$ and $task_j$ of satellite sat_m.
$STORE_m$	Available storage capacity of satellite sat_m.
str_k^{mn}	sat_m storage space capacity to be occupied by $task_k$ for observing tar_n.
$ENERGY_m$	Total energy of satellite sat_m.
eng_k^{mn}	Energy consumed by $task_k$ when sat_m observes tar_n.
x_k^{mn}	**Decision variable.** 1 for selected, and 0 for deleted because the task violates the constraint: $x_k^{mn} = \begin{cases} 1, & task_k(sat_n \text{ observes } tar_m) \text{ is selected} \\ 0, & \text{otherwise} \end{cases}$

an advanced optimization framework to a wide range of problems [13]. According to different search strategies, metaheuristic algorithms are classified into four branches as evolutionary, physical-based, swarm-based, and human-based algorithms [14]. Specifically, the canonical metaheuristic algorithms with high impact on solving optimization problems includes the genetic algorithm (GA) [15,16], particle swarm optimization (PSO) [17,18], tabu search [19], simulated annealing(SA) [20], etc., which mainly belong to the branches of evolutionary and swarm-based algorithms. They have been the research 'limelight' in decades [21] and widely adopted in some engineering domains such as EOSSPs [22,23], because of the capabilities of offering sufficiently high-quality optimization solutions. For instance, Kim et al. [24] adopted the GA to solve the synthetic aperture radar imaging satellite constellation scheduling problem and reduced the system response time. Niu et al. [25] proposed a multi-objective genetic algorithm called NSGA-II (Non-dominated Sorting Genetic Algorithm-II) for the satellite areal task scheduling in a real disaster scenario. Long et al. [20] combined the GA with the SA algorithm and presented a GA–SA hybrid algorithm with a two-phase planning strategy for the EOSSP, which exhibited better optimization than the GA or SA algorithms alone. Although metaheuristic algorithms can achieve better operational results, they can still inevitably fall into local optima [26] and occasionally fail to offer satisfactory convergence performance, especially when dealing with complex constraints in EOSSPs with intensive tasks.

In recent years, algorithms combining deep learning [27], reinforcement learning [28,29], and other neural networks [30] have been proposed to solve the EOSSPs. Lu et al. [31] presented a learning-based approach that uses supervised learning to train a classifier based on a large volume of historical data and embedded the classifier into an onboard greedy algorithm framework. Huang et al. [32] proposed a task scheduling method based on deep reinforcement learning, in which the deep deterministic policy-gradient algorithm was adopted to solve a time-continuous satellite scheduling problem.

The deep reinforcement learning (DRL) has shown great potential in solving EOSSPs. However, the use of DRL algorithms to solve EOSSPs are still in infancy, and most researches are still at the stage of solving single-satellite problems [33–37]. For multi-satellite scheduling problems, Song et al. [38,39] and Chen et al. [40] adapted DRL to optimize parameters or learn search strategies for metaheuristic algorithms to improve optimization efficiency; another way commonly used in literatures [41–43] is to transform the problem into a Markov decision process (MDP) model and design the state, action space and reward functions according to the characteristics of the problems.

Although these algorithms have great advantages in solving efficiency for EOSSPs, some of them are still inferior to baseline algorithms such as GA in solving quality as the task quantities increasing [44–46]. Moreover, the applicability of these methods to large-scale EOSSPs with tens of thousands of tasks has not been verified and discussed [47–49].

The above studies can manage the EOSSPs to a certain extent; however, challenges with respect to finding an effective approach to handle the EOSSP with large-scale intensive tasks remain.

(1) First, owing to the working characteristics of satellites, the single-satellite scheduling problem itself has high complexity, making it difficult to obtain high-quality solutions using simple greedy algorithms or heuristic strategies.
(2) Furthermore, large-scale multi-satellite scheduling often leads to increases in problem size and task intensity, imposing high requirements on the efficiency of the scheduling algorithm.

The emergence of graph networks has provided new insight into determining solutions for EOSSPs with large-scale intensive tasks. Graph neural networks (GNNs) are emerging intelligent algorithms that have been widely used in search, recommendation, risk control, and other important fields in recent years [50]. They integrate deep-learning and graph-computing algorithms to achieve better cognition and problem-solving abilities. Common graph network algorithms include graph convolutional networks (GCNs) [51], graph auto-encoders (GAEs) [52], and graph attention networks (GATs) [53]. The advantage of graph networks is that they can identify the correlation between "unstructured" data, facilitating a deeper insight into the behavioral trajectories and intentions.

Therefore, we proposed a framework based on a hybrid GNN and metaheuristic algorithm to solve the multi-satellite cooperative scheduling problem for large-scale tasks. By combining a GNN with metaheuristic frameworks, such as the GA, a pre-solver module of the GNN for static constraint conflicts and a metaheuristic optimization module for dynamic constraint conflicts of satellite tasks were proposed. The GNN pre-solver module aims to provide effective prior knowledge with a certain universality and generalization for the metaheuristic optimization module. We designed the structure and loss function of the GNN to learn and extract the conflict topology relationships common among satellite missions more effectively, which can guarantee the provision of effective prior information. The metaheuristic optimization module can handle the dynamic constraint conflicts of satellite tasks and obtain search guidance based on the prior information of static constraints, enabling better and faster search performance.

Specifically, the primary research focus of this study can be summarized as follows.

(1) Based on the satellite task characteristics and constraint associations, we analyzed the graph structure relationship between the static and dynamic constraints of the satellite tasks and constructed a multi-satellite task-conflict graph model.
(2) A generalized multi-satellite cooperative scheduling framework was developed based on a hybrid GNN and metaheuristic optimization algorithm, comprising a GNN pre-solver module for static constraint conflicts and a metaheuristic optimization module for dynamic constraint conflicts of satellite missions. In the GNN pre-solver module, the graph sample and aggregate (graphSAGE) network was used to avoid the over-smoothing problem of GCNs for task node feature representation. The

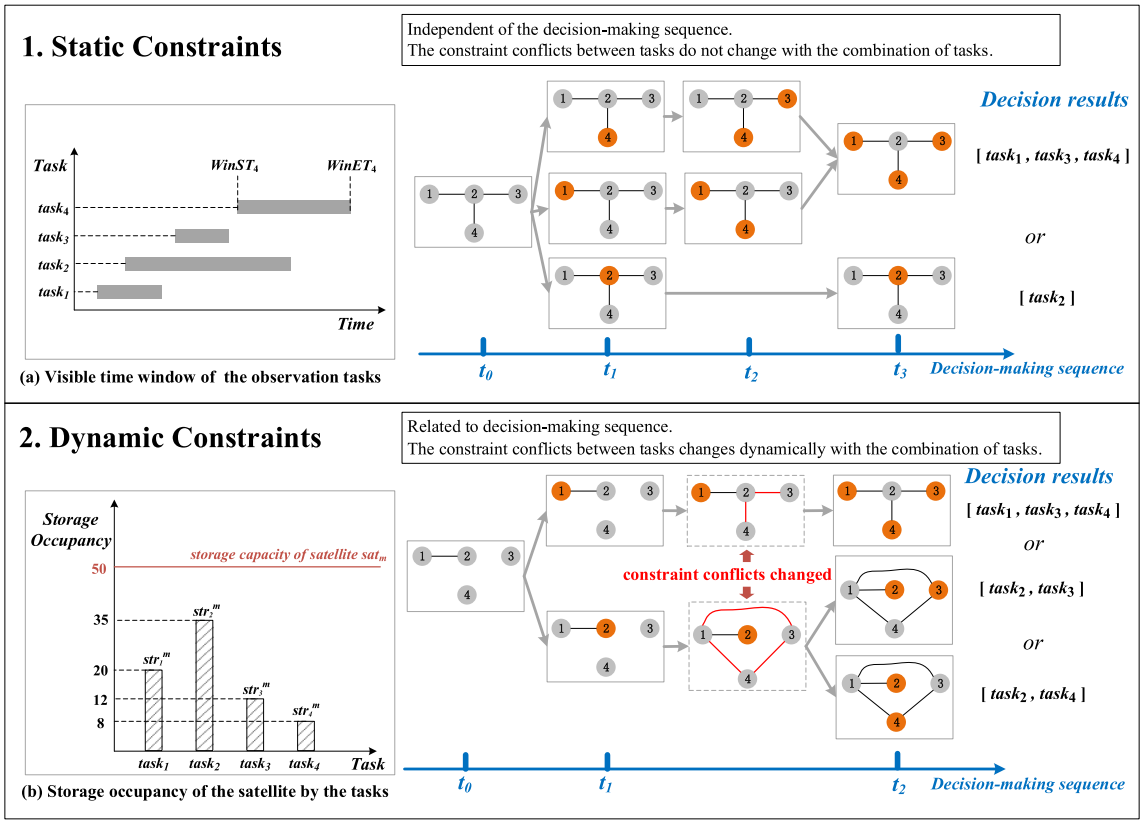


Fig. 1. Each observation task is treated as a node in the graph, and constraint conflicts between tasks are represented by edges.

quadratic unconstrained binary optimization (QUBO) loss function was designed with multiple influencing factors, such as task gain, conflict penalties, and cooperativity requirements, to convert the discrete optimization function into a continuous function. Moreover, we added a greedy threshold to improve the task completion rate.

(3) Numerical experimental results showed that our method can efficiently solve multi-satellite scheduling problems with tens of thousands of tasks. Compared with the commonly used multi-satellite scheduling algorithms, that is, the GA and greedy-GA algorithms, the proposed method can obtain higher-quality solutions under the same conditions and greatly improve the computational efficiency of large-scale mission planning.

The remainder of this paper is organized as follows. Section 2 provides a formal definition of the large-scale EOSSP and analyses of the static and dynamic conflicts of satellite constraints based on a graph representation format. Section 3 introduces the proposed model and method in detail. Section 4 presents the model tests and comparisons conducted at different scales and in various scenarios. Finally, the conclusions are summarized in Section 5.

2. Theory: description of the large-scale EOSSP

2.1. Symbol definitions

For convenience of description, the relevant symbol definitions are listed in Table 1.

2.2. Problem description and simplification of large-scale EOSSP

The large-scale multi-satellite cooperative scheduling problem can be formulated as M satellites cooperatively observing N targets in a planning cycle, such that the objective function is optimal. Essentially, it is a decision-making process on tasks. The final output is a combination of observation tasks that satisfy the constraints and cooperative requirements. Based on an actual satellite system, we simplified the satellite cooperative scheduling problem, as follows:

(1) Each satellite has only one sensor with side-swing capability.
(2) All targets are point targets that can be observed within a visible time window.
(3) The satellite must maintain a stable attitude during the observation task. The transition time, also called the attitude maneuver stabilization time, is the time required for any two consecutive observation tasks in the same orbit.
(4) Considering the demand for high-quality observations, the observation of certain targets requires the collaboration of multiple satellites, that is, a guiding relationship exists among the satellites and a certain sequential relationship in the execution of observation tasks.
(5) Data downloads need not be considered.

2.3. Constraints of large-scale EOSSP

For a large-scale EOSSP, constraints such as time, storage, and energy are considered. The mathematical formulation is as follows.

2.3.1. Task time constraints

A satellite can observe only one target at a time; only one observation task can be executed by a satellite remote sensor at the same time. The following mathematical expression must be satisfied:

$$WinET_k^{mn_1} x_k^{mn_1} + tran_{k(k+1)}^{m} \leqslant WinST_{k+1}^{mn_2} x_{k+1}^{mn_2} \tag{1}$$

In **Eq. (1)**, targets tar_{n_1} and tar_{n_2} occupy satellite sat_m for two adjacent observation tasks, where $task_{k+1}$ can be performed only after $task_k$

ends, and the necessary attitude maneuver is completed. In other words, both $task_k$ and $task_{k+1}$ can be selected for execution only if the above expression is met.

2.3.2. Data storage constraints

Because of the limited storage capacity of the satellite, the data size of all tasks cannot exceed the storage capacity. The mathematical expression representing this constraint is as follows:

$$\sum_{k=1}^{K} str_k^{mn} x_k^{mn} \leqslant STORE_m \tag{2}$$

2.3.3. Energy constraints

For each satellite, the sum of the total energy consumption of the tasks does not exceed the battery energy of the satellite. The mathematical expression is as follows:

$$\sum_{k=1}^{K} eng_k^{mn} x_k^{mn} \leqslant ENERGY_m \tag{3}$$

2.3.4. Cooperative constraints

To improve the quality of target observation, particularly for key target or emergency observation, different types of satellites must usually be dispatched for cooperative observation.

In this study, we set some satellites in the constellation to be guiding satellites and some to be slaving satellites, aiming to achieve cooperative observation of the target in the planning. For example, the target is observed first by the guiding satellite and then by the slaving satellite when conditions allow, so as to achieve high-quality observation of the target. It should be noted that the cooperative constraints were treated as soft rather than hard constraints in this study.

To achieve cooperative observations for as many targets as possible, the constraints can be expressed as follows:

$$\begin{aligned} F(c) &= \max\{\sum_{m_G}^{M_G}\sum_{m_S}^{M_S}\sum_{n=1}^{N} c_n^{m_G m_S} x_i^{m_G n} x_j^{m_S n}\}, (M_G + M_S = M) \\ c_n^{m_G m_S} &= \begin{cases} 1, WinET_i^{m_G n} \leqslant WinST_j^{m_S n} \\ 0, \quad otherwise \end{cases} \\ x_i^{m_G n} &= \begin{cases} 1, task_i(sat_{m_G} observestar_n) isselected \\ 0, \quad otherwise \end{cases} \\ x_j^{m_S n} &= \begin{cases} 1, task_j(sat_{m_S} observestar_n) isselected \\ 0, \quad otherwise \end{cases} \end{aligned} \tag{4}$$

where $WinET_i^{m_G n}$ denotes the end time of the task $task_i$ when tar_n is observed by the guiding satellite sat_{m_G}; $WinST_j^{m_S n}$ denotes the start time of the task $task_j$ when tar_n is observed by the slaving satellite sat_{m_S}; and $x_i^{m_G n}$, $x_j^{m_S n}$ are task decision variables in the scheduling process, representing $task_i$ and $task_j$ selected for execution, respectively.

In the EOSSP, checking and eliminating complex constraint conflicts among tasks affect the algorithm efficiency. Especially for EOSSPs with large-scale tasks, the enormous challenges in handling complex constraint conflicts are self-evident owing to the task intensity.

Essentially, the EOSSP can be regarded as a combinatorial optimization problem with complex constraints. Compared with those of general combinatorial optimization problems in mathematics, most constraints in EOSSPs are hard constraints. Based on their relationship with the task decision sequence, we can divide the above constraints into two categories, namely, static constraints and dynamic constraints. Further, the graph theory to represent constraint conflicts between tasks is shown in Fig. 1.

A graph is usually represented mathematically by $G = (V,E)$, where V denotes the set of nodes and E denotes the set of edges linking the nodes. Considering tasks as nodes and constraint conflicts between tasks as edges, we obtain:$V = \{task_1, task_2, ..., task_k, ..., task_K\}$, $E = \{conflict_{ij} | i, j = 1, 2, ..., K, i \neq j\}$.

Static constraints: These constraints are the inherent properties of tasks independent of the decision sequence. The constraint conflicts between tasks do not change with the combination of tasks, for example, task time constraints and cooperative constraints.

Dynamic constraints: These constraints are related to decision-making sequence. The constraint conflicts between tasks, such as storage and energy constraints, change dynamically with the combination of tasks.

(1) For static constraints (e.g., the time constraints of tasks), as shown in Fig. 1(a), the observation overlap between tasks can be seen when the task is sorted by time. Because only one task can be executed by the same satellite at the same time, time overlap is considered to have a time conflict. The right side of Fig. 1(a) shows the graph of task time conflicts. For static constraints, regardless of what the decision sequence is, the constraint conflict relationship is fixed.

(2) For dynamic constraints (e.g., the data storage constraints), Fig. 1 (b) (left side) shows the storage usage corresponding to each satellite task. At the initial moment of the scheduled decision making, $task_1$ and $task_2$ are in obvious conflict. When a decision is made, different choices at each step affect the conflict relationship between the tasks. For example, in one case, the decision in step t_1 to execute $task_1$, the satellite storage is consumed by $task_1$ (using 20 units), and the remaining 30 units are allocated to $task_2$, $task_3$ and $task_4$. Consequently, the conflicts between $task_2$ and $task_4$ and that between $task_2$ and $task_3$ appear, affecting the decision of the next step. Conversely, if $task_2$ is selected in step t_1, the storage is consumed by $task_2$ (using 35 units), leaving 15 units. At this stage, three conflict edges are added. As distinct from the previous case, different task combination results can be obtained.

2.4. Objective function

The optimization goal of the large-scale EOSSP is to maximize the sum of the observation tasks, which can be expressed as follows:

$$Q = \max(\sum_{k=1}^{K} pri_k^{mn} x_k^{mn}) \tag{5}$$

subject to:

$$WinET_k^{mn_1} x_k^{mn_1} + tran_{k(k+1)}^{m} \leqslant WinST_{k+1}^{mn_2} x_{k+1}^{mn_2}$$

$$\sum_{k=1}^{K} str_k^{mn} x_k^{mn} \leqslant STORE_m$$

$$\sum_{k=1}^{K} eng_k^{mn} x_k^{mn} \leqslant ENERGY_m$$

$$WinET_k^{mn_1}, WinST_{k+1}^{mn_2}, tran_{k(k+1)}^{m}, str_k^{mn}, eng_k^{mn}, ENERGY_m, STORE_m \geqslant 0$$

3. Methods: Large-scale multi-satellite scheduling frame with hybrid graphSAGE and GA

3.1. Preliminaries

To establish our hybrid solver framework for large-scale EOSSPs, we begin with a brief review of combinatorial optimization, GNNs, and metaheuristic algorithms.

Combinatorial Optimization. Generally, a combinatorial

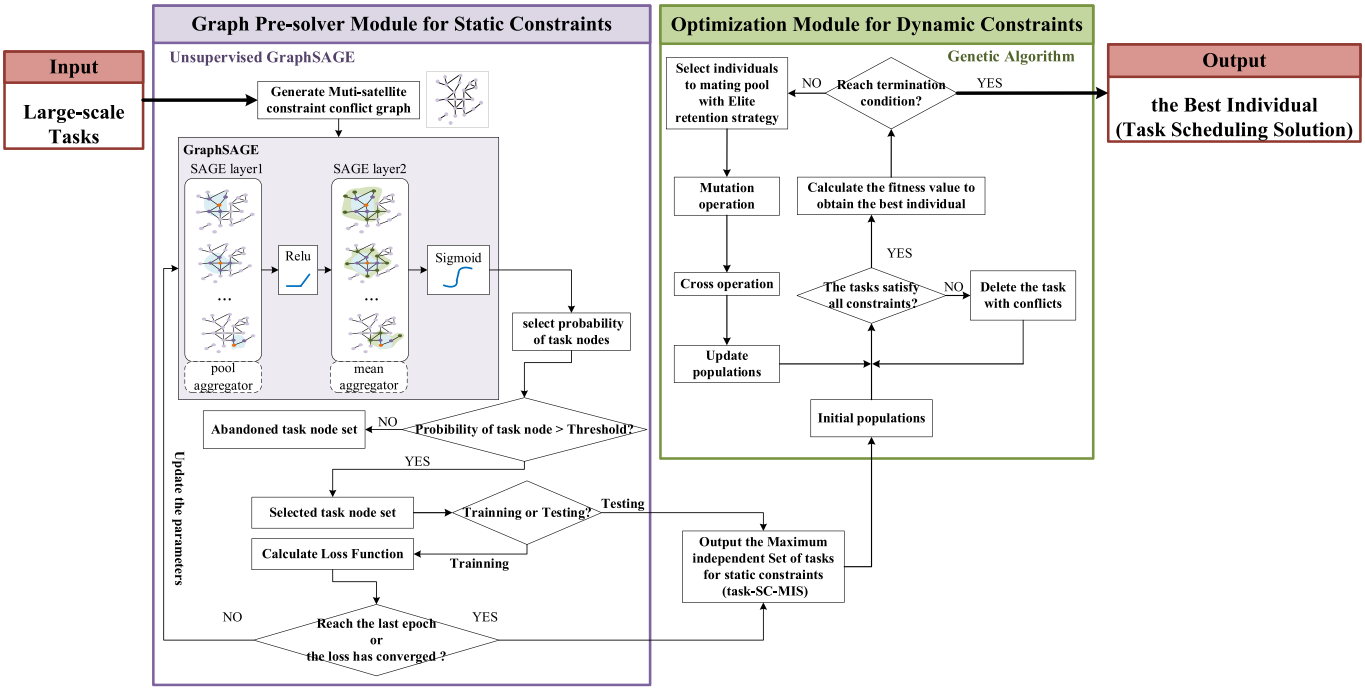


Fig. 2. End-to-end solution architecture with hybrid unsupervised greedy threshold graphSAGE and genetic algorithm (SAGE-gt-GA), including the input module, graph pre-solver module for static constraints, optimization module for dynamic constraints, and output module.

optimization problem can be formulated as a constrained min-optimization program [54]. Canonical combinatorial optimization problems include the maximum cut, maximum independent set (MIS) problems, and Traveling Salesman Problem (TSP). Moreover, the EOSSP can be regarded as a combinatorial optimization problem for tasks. This field is concerned with settings where a large number of yes/no decisions must be made. Each set of decisions yielding a corresponding objective function value, such as a cost or profit value, to be optimized. Exact solutions are not feasible for sufficiently large systems because of the exponential growth of the solution space as the number of variables (n) increases in most instances.

Graph Neural Networks. GNNs form a family of neural networks that are powerful when adopted to exploit non-Euclidean space data [55]. They specialize in representation learning through learning how to aggregate information (including topology and node features) in graphs. Generally, a GNN layer comprises three functions [56], as expressed in **Eq. (6)**: (i) a message passing function (ϕ) that permits information exchange between nodes over edges, (ii) an aggregation function (ρ) that combines the collection of received messages into a fixed-length representation, and (iii) an update activation function (ψ) that produces node-level representations given the previous layer representation and aggregated information.

$$m_e^{(t+1)} = \phi(x_v^{(t)}, x_u^{(t)}, w_e^{(t+1)})$$

$$x_v^{(t+1)} = \psi(x_v^{(t)}, \rho(m_e^{(t+1)})) \tag{6}$$

Here, x_v denotes a feature of node v in the graph, and w_e denotes the weight of the edge (v, u) of nodes v and u.

Typical GNN architectures include the GCN [57], graphSAGE [58], and GAT [59]. They provide processing capabilities for ESSOPs under a task-conflict graph representation.

Metaheuristic Algorithms. Metaheuristic algorithms are among the most widely used optimization algorithms. Typical metaheuristic algorithms, including the tabu search [19], GA [15,16], and PSO [17,18] algorithms. The GA is the most utilized metaheuristic algorithms on solving scheduling problems in literatures [60], which is widely applied to solve EOSSPs and is verified to be efficient in solving certain-scale satellite scheduling problems with specific constraints, but not for large-scale problems in terms of solution efficiency and quality.

3.2. Solver framework with hybrid unsupervised graphSAGE and GA

The large-scale multi-satellite cooperative scheduling problem is a combinatorial optimization problem with complex constraints, including static and dynamic constraints. The consideration and handling of static and dynamic constraint conflicts render the solution process challenging, especially when the task scale is the order of tens of thousands. Consequently, we propose an end-to-end solution architecture combining an unsupervised graphSAGE and GA, enabling decision making for EOSSPs with large-scale tasks. As shown in Fig. 2, the solver framework is divided into an input module, a graph pre-solver module for static constraints, an optimization module for dynamic constraints, and an output module.

Static constraints (e.g., time and cooperative constraints) are inherent attributes of tasks, and some common knowledge may exist in different scenarios. By learning and extracting the common knowledge, the relationships (e.g., the time-conflict or cooperative relationships) between the tasks in new scenarios can be obtained quickly. We attempted to learn and extract knowledge in a constraint conflict graph. In the graph, the tasks and conflicts are represented as nodes and edges, respectively, and two nodes are connected by edges representing two conflicting tasks. As many tasks as possible must be completed while avoiding conflicting tasks in the decision-making process; therefore, this problem with static constraints can be regarded as an MIS problem.

For dynamic constraints (e.g., data storage and energy constraints), the conflicting relationships change dynamically with the decision sequences, and their complex relationships can be difficult to characterize. However, a problem with dynamic constraints can be considered to be similar to the 0/1 knapsack problem. Based on the pre-solving results of the graphSAGE pre-solver module and combined with a metaheuristic algorithm, such as the GA, it is equivalent to using an optimal feasible solution as the starting point and continuing to search for a better solution on this basis.

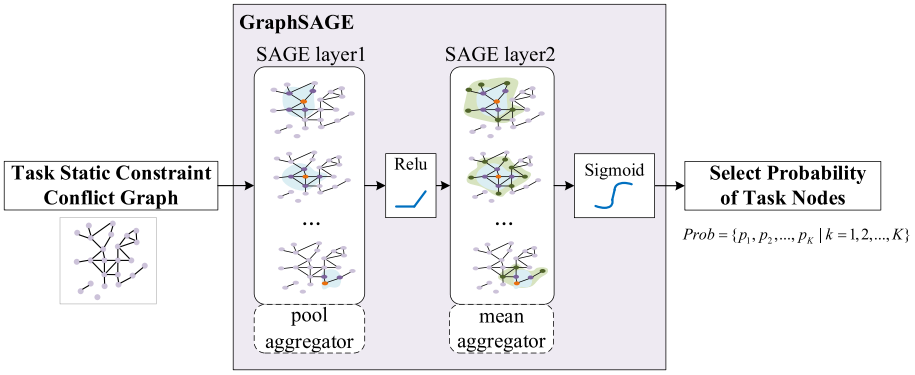


Fig. 3. Network architecture.

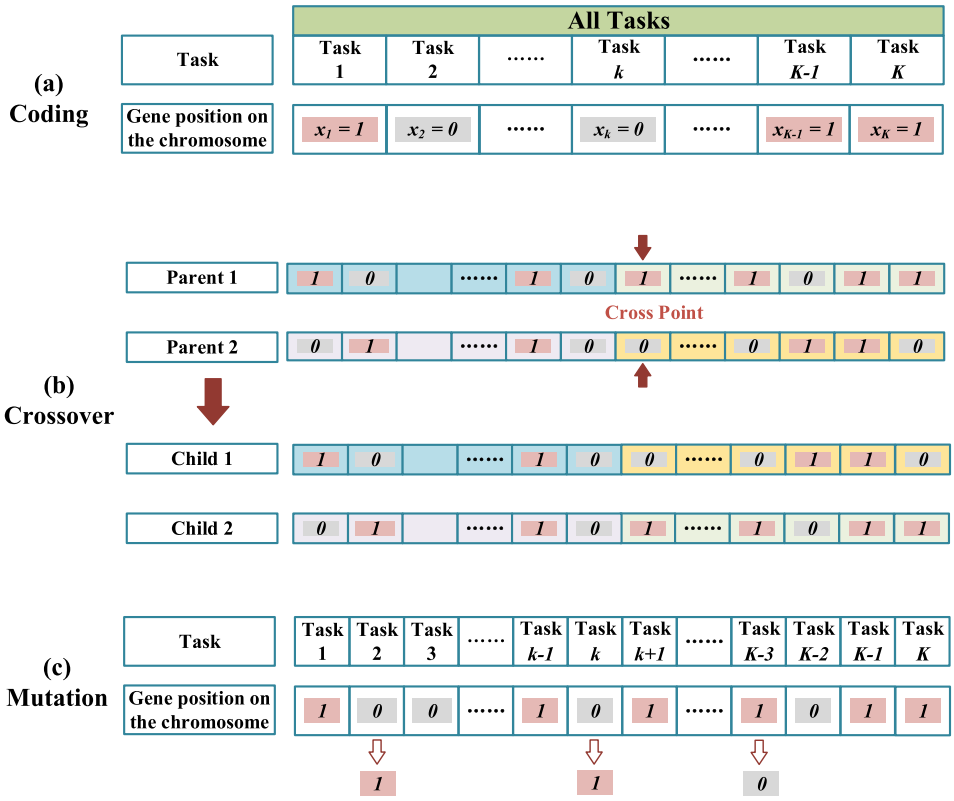


Fig. 4. Schematic of coding, crossover, and mutation.

3.2.1. Input and output module

(1) Input Module: Input large-scale tasks using the original information.

$$Input = \{(task_k, sat_m, tar_n, WinST_k^{mn}, WinET_k^{mn}, dur_k^{mn}, str_k^{mn}, eng_k^{mn}, pri_k^{mn}) | k = 1, 2, ..., K\}$$

(2) Output Module: Output task scheduling.

$$Output = \{x_k^{mn} | k = 1, 2, ..., K\}$$

3.2.2. Graph pre-solver module for static constraints

The primary role of the graph pre-solver module is to manage and filter the static constraint conflicts of the satellite tasks, to provide an excellent reference starting point for the subsequent optimization module.

In this study, the graphSAGE network framework is adopted to avoid the over-smoothing problem of the GCN for task node feature representation. The QUBO loss function is designed with multiple influencing factors such as task reward, conflict penalty, and cooperativity requirements, to convert the discrete optimization function into a continuous function. Additionally, a greedy threshold is added to improve the task execution rate.

Network Architecture. We built a SAGE network with two layers, as shown in Fig. 3. Different aggregation functions can be adopted to characterize the differentiated features of the task nodes more effectively. Here, *pool* is used for the first layer, as expressed in **Eq. (7)**, and *mean* is used for the second layer, as addressed in **Eq. (8)**:

$$AGGREGATE_l^{pool} = \max(\{\sigma(W_{pool} h_u^l + b), \forall u \in N(v)) \quad (7)$$

$$AGGREGATE_l^{mean} = mean(\{h_v^l\} \cup \{h_u^l, \forall u \in N(v)\}) \quad (8)$$

The update function of the node feature can be expressed as follows:

$$h_{N(v)}^{l+1} = AGGREGATE_l(\{h_u^l, \forall u \in N(v)\}) \quad (9)$$

Table 2
Simulation data value settings.

Element	Value
Task	[1, *Task_Num*]
Sat	[1, *Satellite_Num*]
(*GuideSat* and *SalveSat*)	$GuideNum + SlaveNum = Satellite\_Num$
Target	[1, *Target_Num*]
WinST	[0,4]
WinET	$[WinST + 0.03, WinST + 0.08]$
tran	[0, 0.01]
str	[0, 0.03]
eng	[0, 0.03]
pri	[0.1, 0.9]
STORE	0.5
ENERGY	0.5

$$h_v^{l+1} = \sigma(W^l \cdot CONCAT(h_v^l, h_{N(v)}^l)) \tag{10}$$

Loss Function. The loss function determines the solution direction. In recent years, the QUBO model has provided an approach that enables the unification of various types of NP-hard combinatorial optimization problems [61]. This can be expressed using the following Hamiltonian equation:

$$H_{QUBO} = X^T Q X = \sum_{i,j} x_i q_{ij} x_j \tag{11}$$

where $X = \{(x_1, x_2, ..., x_k, ..., x_K) | x_k = 0, 1\}$ denotes a vector of binary decision variables, and Q denotes a square matrix of constant numbers that encodes the actual problem to be solved.

Obviously, the optimization of large-scale EOSSP can be expressed as QUBO model, where the X is the decision variable of tasks, and Q is designed to be a square matrix including task gains, conflict penalties, and cooperativity requirements.

However, the decision variable of tasks in the large-scale EOSSP is defined to be binary and finite, which leads to the discrete objective function in the optimization. To better exert the representation and extraction ability of SAGE network, in the graph pre-solver module, we express the X as the probability values of tasks to be decided to execute, thus converting it into a continuous optimization function.

The loss function for a large-scale task scheduling problem, considering static constraints, such as task time conflicts and cooperativity requirements, as follows:

$$Loss = -X_p^T(\alpha \cdot R + \beta \cdot P + \gamma \cdot C)X_p \tag{12}$$

where $X_p = \{(p_1, p_2, ..., p_k, ..., p_K) | 0 < p_k < 1\}$ denotes a vector of probability values of tasks to be selected; matrix R denotes a square matrix of rewards for tasks (e.g., the priority of each task); matrix P, which is typically symmetric or upper triangular, denotes a square penalty matrix when a conflict (edge in the graph) exists between the tasks; matrix C denotes a square matrix of the cooperative benefits when a cooperative relationship exists between the tasks; α, β, and γ denote the reward, penalty, and cooperative gain coefficients, respectively.

Greedy Threshold. The value of each task decision is either 0 or 1. The outputs of the module, however, are decimals instead of one-hot encoded. We imposed a sigmoid function to obtain the task node

Table 4
Training results of task sets at different scales.

Training	MIS nodes			
Task_Num	GCN	GCN-gt	SAGE	SAGE-gt
2000	160	160	493	493
4000	29	29	562	553
6000	156	156	1073	1077
8000	56	56	1100	1101
10,000	16	16	1125	1145
20,000	39	39	2187	2239
30,000	67	67	3418	3421
40,000	121	121	4474	4499
50,000	117	117	5897	5957
	Calculation Time (s)			
Task_Num	GCN	GCN-gt	SAGE	SAGE-gt
2000	305.193	306.952	114.015	114.785
4000	180.632	178.289	164.199	127.065
6000	296.803	298.153	222.045	213.296
8000	188.456	186.039	247.013	222.921
10,000	240.145	244.171	224.245	250.913
20,000	619.397	604.882	410.237	489.457
30,000	1234.145	1225.682	1078.822	1068.811
40,000	1835.672	1846.813	1822.546	2128.419
50,000	3962.554	4012.129	3397.679	3531.101

Table 3
Datasets settings.

Training datasets					
Task_Num	Satellite_Num	Target_Num	Node_Num	Edge_Num	Average_Degree
2000	10	500	2000	5277	5.277
4000	10	500	4000	22,023	11.012
6000	20	500	6000	24,639	8.213
8000	20	500	8000	43,510	10.878
10,000	20	500	10,000	68,375	13.675
20,000	40	800	20,000	136,388	13.639
30,000	60	1200	30,000	204,722	13.648
40,000	80	1600	40,000	272,602	13.630
50,000	100	2000	50,000	340,183	13.607
Test Datasets					
Task_Num	**Satellite_Num**	**Target_Num**	**Node_Num**	**Edge_Num**	**Average_Degree**
2000	10	1000	2000	5453	5.453
4000	15	2000	4000	14,606	7.303
6000	20	2000	6000	24,767	8.25566
8000	20	2000	8000	43,355	10.83875
10,000	25	2000	10,000	54,311	10.8622
20,000	40	3000	20,000	137,261	13.7261
30,000	50	3000	30,000	246,231	16.4154
40,000	80	3000	40,000	272,907	13.64535
50,000	100	2000	50,000	342,494	13.699

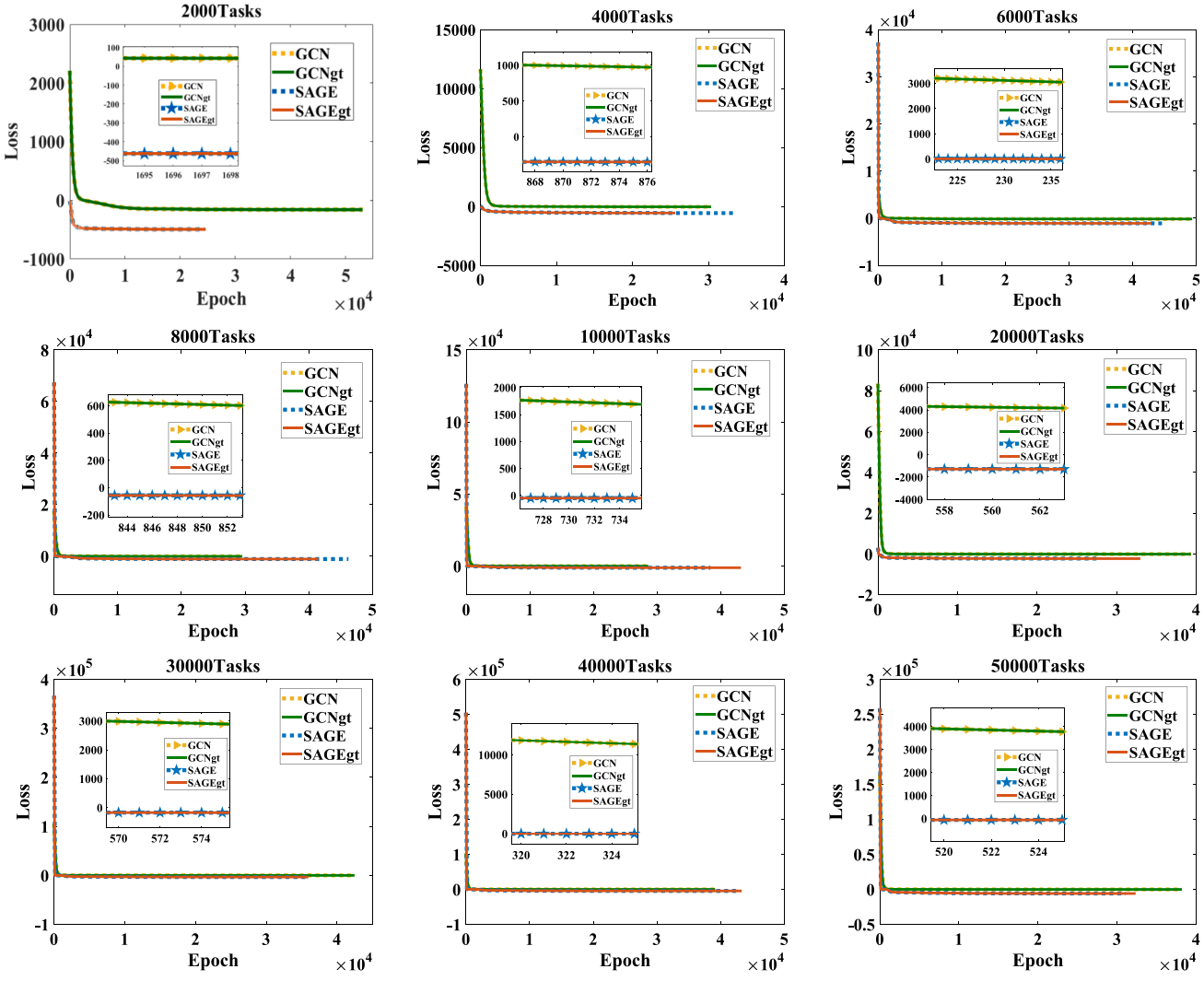


Fig. 5. Training loss-function curves of each method at different task scales. The loss functions of all methods converge under different satellite-task training sets.

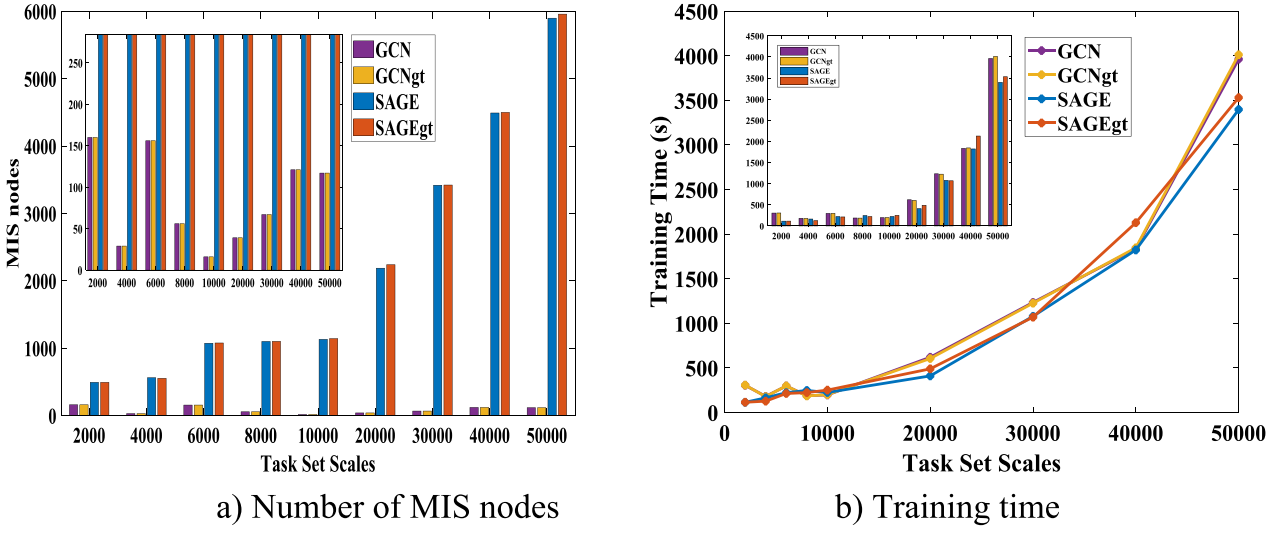


Fig. 6. Performance curves of each method at different (training) task quantity scales.

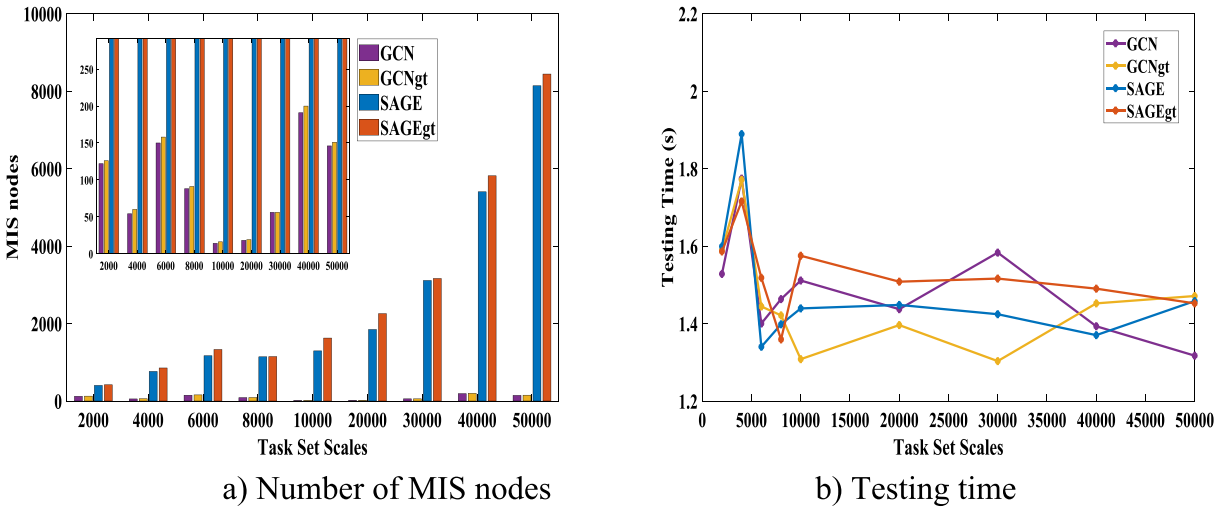


Fig. 7. Performance curves of each method at different (test) task quantity scales.

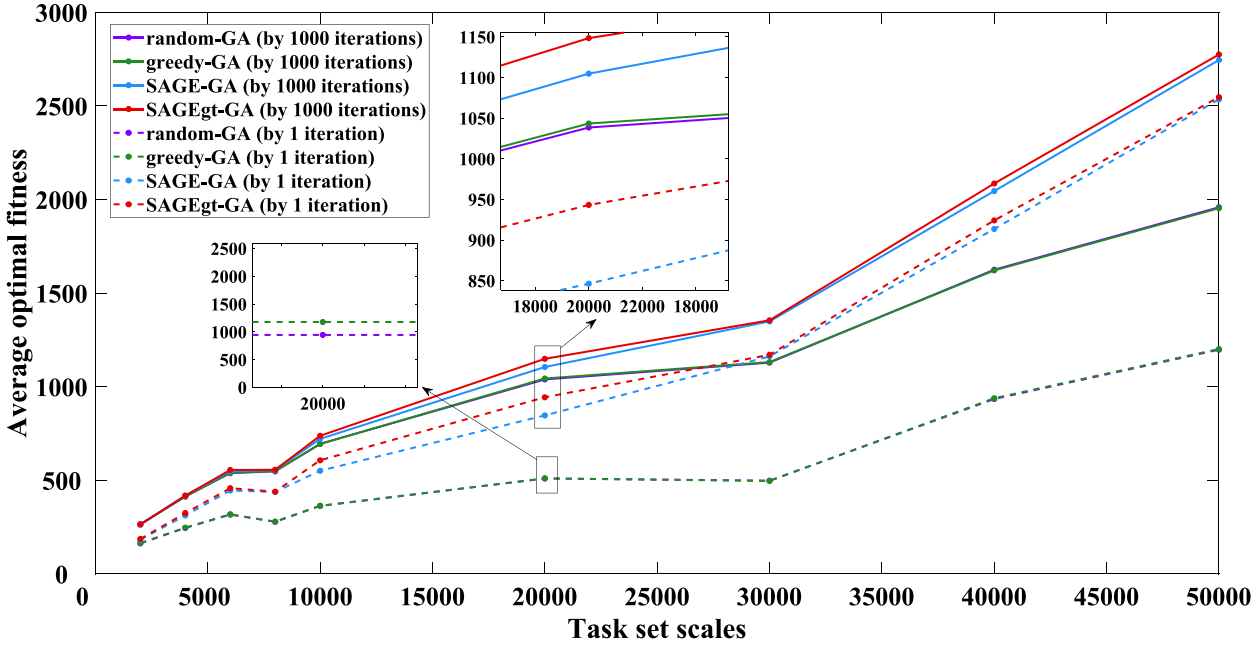


Fig. 8. Average optimal fitness values.

selection probabilities output by the module and then modified the values to 0 and 1 with the threshold. We designed a greedy threshold to complete as many tasks as possible while satisfying the constraints to improve the task execution rate.

$$\begin{cases} x_k = 1, if\ p_k \geqslant GreedyThreshold \\ x_k = 0,\ if\ p_k < GreedyThreshold \end{cases}$$

Training Stop Conditions. Training is stopped when one of the following conditions is met:

(1) The maximum number of training epochs is reached, or
(2) The lifting rate of the evaluation index in a certain number of epochs is continuously less than the given threshold.

3.2.3. *Optimization module for dynamic constraints*

The graph pre-solver for static constraints filters some constraints and offers a better search starting point for the optimization module. The GA, as one of the most utilized metaheuristic algorithms, is used for optimization with dynamic constraints (storage and energy constraints) in this study. It includes coding, crossover and mutation, selection mechanism, and fitness function design.

Coding. Binary coding is performed based on the task decision variables, as shown in Fig. 4**(a)**, with each chromosome bit representing an observational task. Its value is either 0 or 1, indicating whether the task is selected to be scheduled for execution or not. The length of the chromosome is the number of observation tasks performed by all satellites for all targets.

Crossover and Mutation. The crossover operation mimics the reproductive phenomenon of biological evolution by exchanging the gene fragments of two chromosomes to produce new individuals. In this study, single-point crossover is adopted, as shown in Fig. 4**(b)**. A random multipoint mutation strategy is adopted, as shown in Fig. 4**(c)**, to ensure the genetic diversity of the population and prevent the evolutionary process from falling into a local optimum prematurely. For each individual in the mating pool, its chromosome decides whether to mutate based on a small mutation probability. When a chromosome is selected

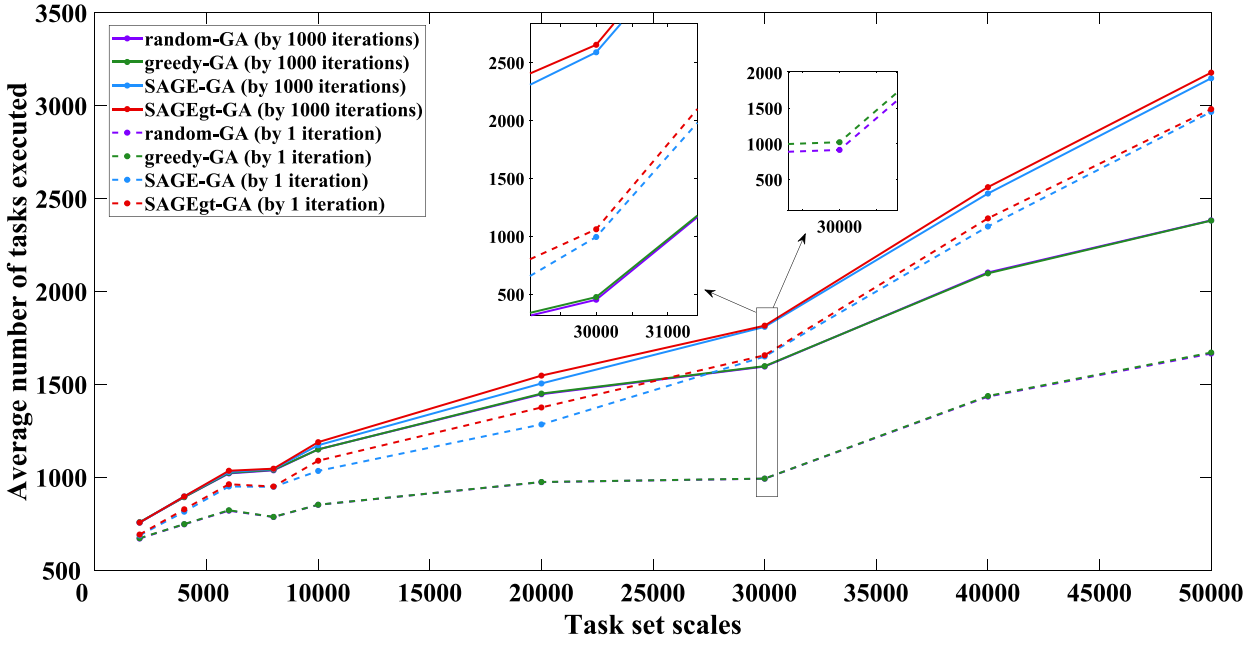


Fig. 9. Average number of tasks executed by the final scheduling program.

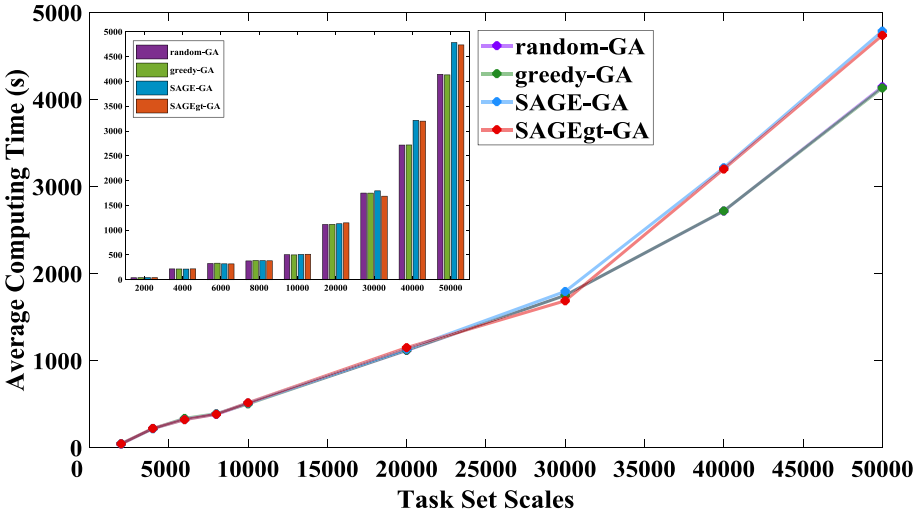


Fig. 10. Average calculation time.

for mutation, some chromosome bits are randomly selected to change their values.

Selection. Owing to the randomness of crossover and mutation, offspring may not inherit the superior genes of their parents in the genetic process. Consequently, we designed the following selection mechanism: for the newly generated population, an elite retention strategy is adopted. The individuals with the highest fitness values in the population are directly selected into the mating pool, and the remaining individuals are selected based on the roulette mechanism. Similar to the evolutionary "survival of the fittest" concept in nature, individuals with high fitness values have higher probabilities of entering the mating pool, whereas individuals with low fitness values have lower probabilities of passing on their gene fragments.

Fitness Function. The fitness value is the direction of evolution in the GA, which is the bridge connecting the algorithm and optimization objective, directly affecting the performance and efficiency of the algorithm in solving practical problems. The fitness function is typically established based on the optimization goal of the problem, where individuals are selected by evaluating their fitness in the population. In this study, the objective function described in Section 2.4 can be regarded as the fitness function.

4. Results: Numerical experiments

To verify the performance of the SAGE-gt-GA in solving large-scale multi-satellite task scheduling problems, we conducted extensive experiments and compared the results with those of several baselines. Section 4.1 describes the dataset; Section 4.2 summarizes the baselines and experimental settings, and the results and analysis are presented in Section 4.3.

4.1. Data generation

Due to the difficulty in obtaining the real data on a large scale and the lack of the benchmark dataset for satellite scheduling problem, we set reasonable value intervals for each parameters based on laws and

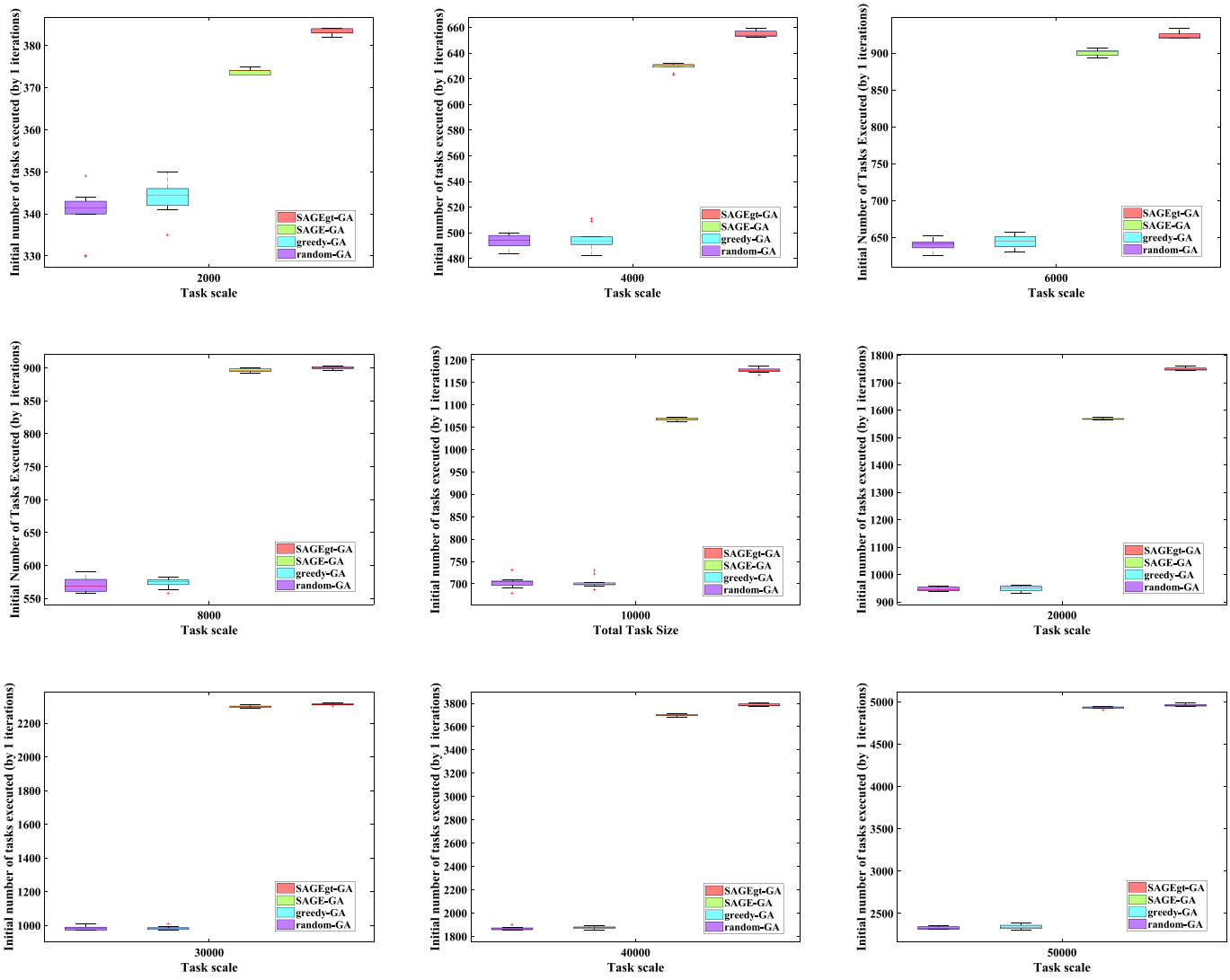


Fig. 11. Initial solution provided by the graph pre-solver module to the optimization module at different task-quantity scales. The vertical coordinate represents the number of tasks executed in the initial solution. Each chart shows the results for a separate task scale.

experience of actual EOSSPs to construct data sets, like the similar method used by Chen et al. [34], Ma et al. [35] and Wei et al. [62]. The simulation data value settings are listed in Table 2, and the dataset settings are presented in Table 3. We randomly generated satellite task datasets of different sizes (Table 3) within the range of values listed in Table 2 and constructed undirected graphs of the satellite tasks, where the corresponding numbers of nodes, edges, and node average degrees are provided.

4.2. Baselines and experimental settings

4.2.1. Baselines and evaluations

To verify the effectiveness of the proposed SAGE-gt pre-solver module, we compared the proposed SAGE-gt with the following baselines.

(1) **GCN:** Traditional GCN model.
(2) **GCN-gt:** GCN model with greedy threshold.
(3) **SAGE:** SAGE network model (the network architecture is the same as SAGE-gt, but without the greedy threshold).

The model performance evaluation indices include the following.

(1) **Comparison of training results for task data at different scales**: The number of MIS nodes, training time, and loss-function convergence are compared for the purpose of our study. The number of MIS nodes indicates the task selection results based on a static constraint graph, which represents the number of tasks without static conflicts. The greater the number of MIS nodes, the better the performance. The training time and loss function reflect the training process. The shorter the training time, the faster the convergence rate of the loss function, and the better the training effect.
(2) **Model generalization performance test**: The model performance was tested for different task scales, with certain differences from the training data distribution. The performance evaluations were conducted primarily based on the number of MIS nodes and computational time.

To verify the effectiveness of the methodological framework for solving large-scale multi-satellite scheduling problems, we compared the proposed SAGE-gt-GA with the following baselines.

(1) **random-GA:** A GA based on a stochastic strategy, that is, the initial population is generated using a stochastic strategy, and the

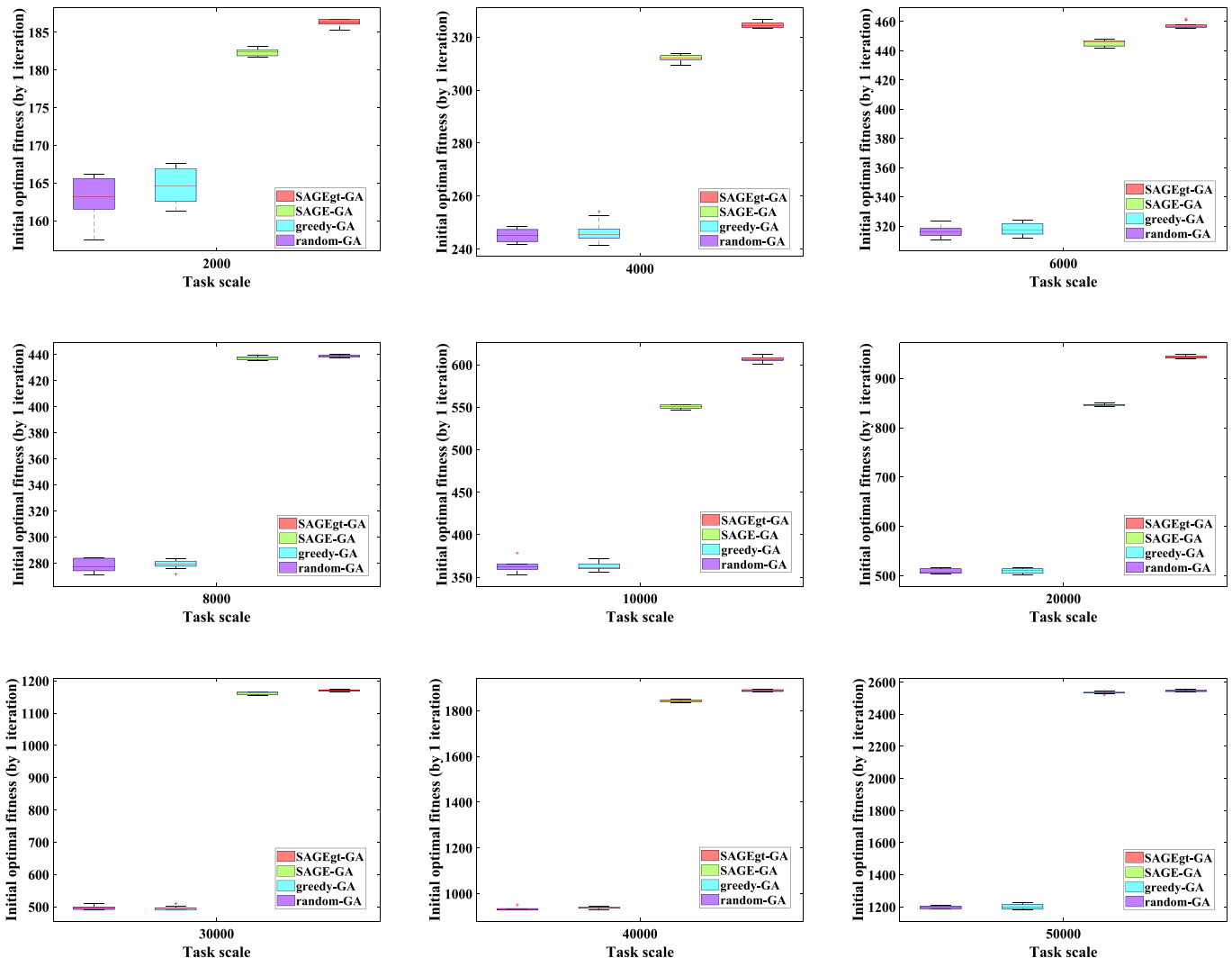


Fig. 12. Fitness values of the initial solutions provided by the graph pre-solver module to the optimization module. Each chart shows the results for a separate task scale.

gene values of all individuals in the initial population are random.

(2) **greedy-GA:** A GA based on a greedy strategy, that is, the initial population is generated using a greedy strategy, and the initial population contains individuals whose gene values are all 1.

(3) **SAGE-GA:** A GA based on the graph pre-solver module, that is, the initial population contains the results of the static graph network pre-solver, and the graph network model adopts the SAGE network model with a non-greedy threshold.

The performance evaluation indices include the following.

(1) The optimal fitness value of the initial population and corresponding number of scheduled execution tasks. A higher optimal fitness value implies that more tasks can be planned to be executed, indicating an initial reference point of better quality.

(2) The optimal fitness value of the final population (when the iteration stops) and the corresponding number of scheduled execution tasks. A higher optimal fitness value implies that more tasks can be planned to be executed, indicating better quality solution.

(3) Computing time.

4.2.2. Experimental settings

The simulation computer environment comprised an Intel Core i7-8700 CPU, running at 3.20 GHz with 32.0 GB RAM. The main network parameter settings included $learning\_rate = 10^{-4}$, $\beta = 2$, $\alpha = -1$, $\gamma = -0.3$, and $max\_epoch = 10^5$. To save training time, we set an early stop mechanism, that is when the loss variation was less than 10^{-4} for *10,000* epochs, the loss function was considered to have reached the convergence state, and training was stopped early.

4.3. Results and discussion

4.3.1. The performance of graph pre-solver module for static constraints

To verify the ability of the graph pre-solver module, we trained it using satellite tasks of different quantity scales and tested the generalization performance of the model using satellite task data with certain distribution differences. The results and analyses are as follows.

(1) **Analysis of Training Results:** The training results of the proposed method and baselines based on satellite task sets of different quantities and scales are listed in Table 4, and the loss function curve is shown in Fig. 5. Table 4 reveals that the proposed SAGE-gt algorithm displays significant advantages with respect to handling static conflicts between satellite tasks, especially for large-scale tasks.

The MIS set output by the graph pre-solver module represents the

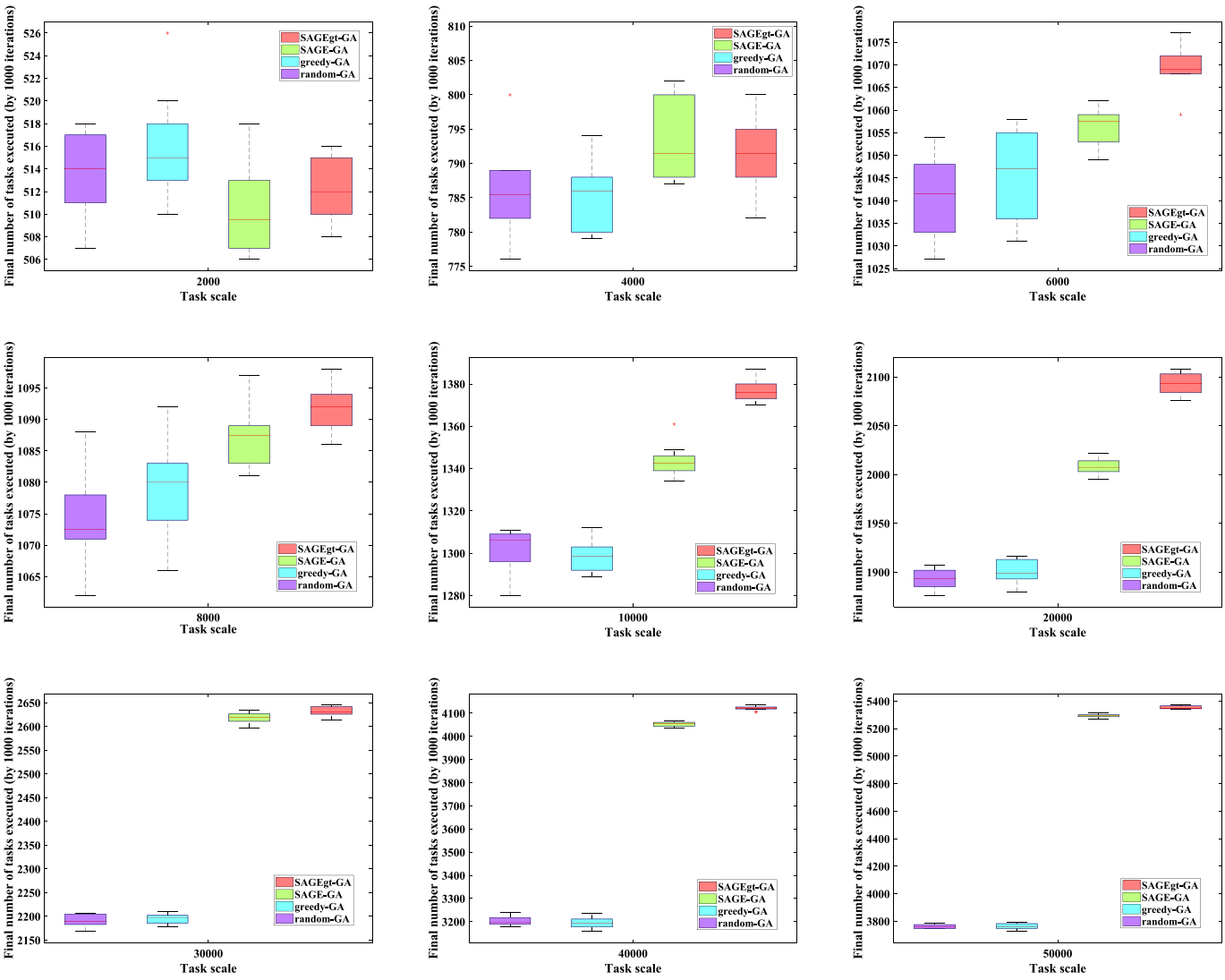


Fig. 13. Number of tasks executed in the final schedule solutions output by the optimization module after 1000 iterations. Each chart shows the results for a separate task scale.

task selection result based on the static constraint graph, and the number of MIS nodes denotes the number of tasks without static conflicts. As is evident from the statistics, the SAGE network has clear advantages over the traditional GCN with respect to managing the MIS problem of satellite task conflicts, and its advantages manifest more clearly as the task scale increases. This may be because the over-smoothing phenomenon occurs in the traditional GCN when dealing with the static conflict graph of satellite tasks. The differences between task nodes are ignored during the feature aggregating and expressing process, owing to the complex conflict edges between task nodes. In other words, tasks with more or fewer conflicts may not be effectively expressed, resulting in a low number of output MISs.

By contrast, the SAGE network adopts the feature *concatenate* between the central node and neighboring nodes in the aggregation process and uses different aggregation functions in the layer. It avoids the over-smoothing phenomenon and allows task nodes with different conflict degrees to obtain a sufficiently differentiated representation. Additionally, the greedy threshold mechanism enhances the handling of larger-scale task-conflict graphs.

The performance of each method at different task node scales is demonstrated in Fig. 6. As illustrated in Fig. 6**(b)**, the training time increases as the satellite task scale increases, although the training time differs slightly between the methods. However, under the condition of almost equal time costs, as depicted in Fig. 6**(a)**, the SAGE network structure adopted in this study has notable advantages over the GCN when dealing with the static conflicts of large-scale satellite tasks.

(2) **Analysis of Generalization Test Results:** Based on a set of test data (with certain distribution differences from the training data), we tested the generalization performance of the graph model. The performance curves of each method under different task quantities are as shown in Fig. 7.

The above results demonstrate that the proposed SAGE-gt pre-solver module exhibits excellent and stable performance in dealing with different task set scales. Each method achieves rapid processing of task-conflict graphs of different sizes within 2s time overheads. However, in terms of the MIS results, the SAGE-gt model outperforms the other models.

4.3.2. *The performance of the optimization module for dynamic constraints*

In this section, the effectiveness of the proposed optimization module and solution framework for satellite scheduling is further verified.

The performance curves and bars of each method as the task scale increases are presented in Figs. 8–10. Considering the randomness of metaheuristic search algorithms such as the GA, all data in the figures

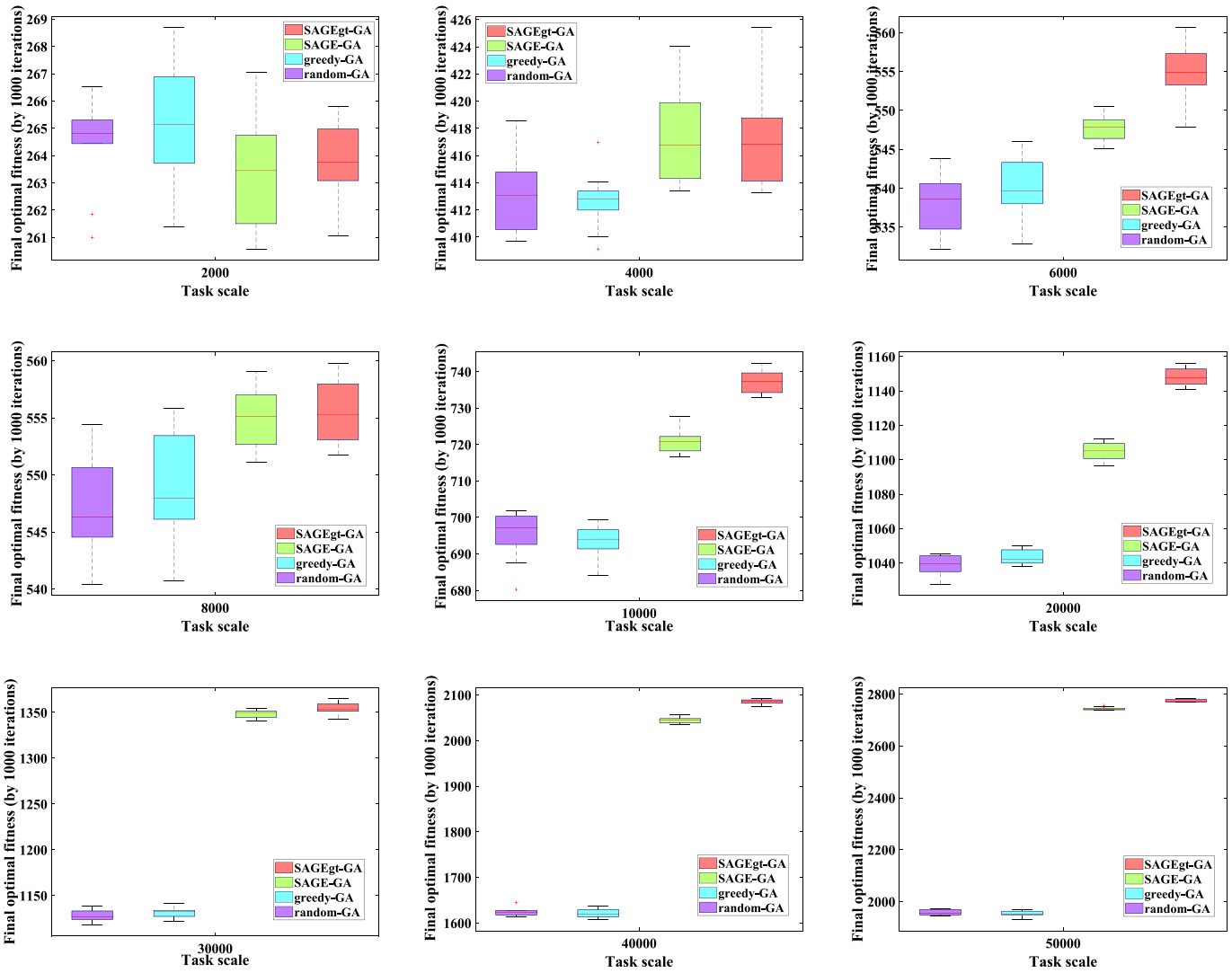


Fig. 14. Fitness values of the final schedule solutions output by the optimization module after 1000 iterations. Each chart shows the results for a separate task scale.

are the averages of the results of ten runs. Fig. 8 compares the initial and final solution fitness values under different methods when solving multi-satellite scheduling problems at different task scales, Fig. 9 shows the statistics of the number of tasks executed corresponding to the initial and final solutions, and Fig. 10 depicts the computational overhead of each method.

Clearly, under the same iteration conditions, the SAGE-gt-GA can optimize the solution based on higher-quality reference solutions because of the better initial solutions, which is even higher than the final solution of the baselines in the case of larger-scale tasks. Thus, better schedule solutions are yielded at almost similar computational costs.

To analyze the performance of each method more comprehensively, box-line plots of the proposed method and baselines based on the data of ten runs are shown in Figs. 11–14. These figures demonstrate the distribution of the data from multiple groups of experimental results at different task scales, including the upper and lower bounds of the data, the median, the outliers, and a box consisting of the first quartile and the third quartile. The median can reflect the central tendency and average of the data, which is effective to compare the average performance of each algorithm under the evaluation index. The size of the box reflects the fluctuation of the data, smaller box corresponding to less data fluctuation and higher algorithm stability. Apparently, the SAGE-gt-GA method exhibits excellent, stable performance in each evaluation index, and its advantages become more obvious as the task scale increases.

5. Conclusions

In this study, a multi-satellite cooperative scheduling method for large-scale tasks based on a hybrid GNN and metaheuristic algorithm is proposed. First, the graph structure relationship between the static and dynamic constraints of satellite tasks was analyzed, and a multi-satellite task-conflict graph model was constructed. Second, a generalized multi-satellite cooperative scheduling framework using a hybrid GNN and metaheuristic optimization algorithm was proposed. The framework consists of a graph pre-solver module for static constraint conflicts and a metaheuristic optimization module for dynamic constraint conflicts of satellite missions. In the graph pre-solver module, we adopted the graphSAGE-gt network to avoid the over-smoothing problem for task node feature representation and added a greedy threshold to improve the task completion rate. Additionally, we designed a QUBO loss function with multiple influencing factors, such as task gain, conflict penalties, and cooperativity requirements, to convert the discrete optimization into the optimization with a continuous function. Finally, the experimental results showed that the proposed method could efficiently solve the multi-satellite scheduling problem with the order of tens of thousands of tasks. And it has more excellent and stable performances in various scenarios, compared with the common multi-satellite scheduling algorithms (GA and greedy-GA).

This study may provide a novel insight into the large-scale mission

planning problems. It tries to learn and extract the common knowledge in the static constraint conflicts in different scenarios and greatly improves the computational efficiency of scheduling with intensive tasks. In addition, the method framework proposed in this study can combine with other metaheuristic algorithms, and may be able to expand to other similar fields rather than being limited to the satellite scheduling.

In future studies, we attempt to make more and further work to the actual EOSSPs. First, we aim to conduct in-depth research on the dynamic constraint processing problem in large-scale multi-satellite scheduling, and explore the conflict and correlation mechanisms of the dynamic constraints among satellite tasks. In addition, the powerful potential of DRL and GNNs for feature representation and sequential decision making will be leveraged to establish a rapid solution model for end-to-end large-scale multi-satellite scheduling.

CRediT authorship contribution statement

Xiaoen Feng: Methodology, Software, Formal analysis, Validation, Investigation, Data curation, Writing – original draft, Visualization. **Yuqing Li:** Conceptualization, Formal analysis, Funding acquisition, Methodology, Project administration, Resources, Writing – review & editing, Data curation. **Minqiang Xu:** Conceptualization, Formal analysis, Funding acquisition, Methodology, Resources, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported in part by the National Key R&D Program of China [grant number 2021YFA1003501], the National Natural Science Foundation of China [grant number 52075117], the Science Research Project [grant number JSZL2020203B004] and the Provincial Key R&D Program of Heilongjiang [grant number 2022ZX01A20].

References

[1] G. Wu, W. Pedrycz, H. Li, M. Ma, J. Liu, Coordinated planning of heterogeneous Earth observation resources, IEEE Trans. Syst. Man Cybern. Syst. 46 (2015) 109–125, https://doi.org/10.1109/TSMC.2015.2431643.
[2] H. Kim, Y.K. Chang, Optimal mission scheduling for hybrid synthetic aperture radar satellite constellation based on weighting factors, Aerosp. Sci. Technol. 107 (2020) 106287, https://doi.org/10.1016/j.ast.2020.106287.
[3] Z. Li, X. Li, Current status and prospect of imaging satellite task dynamic scheduling methods. 8th international conference on intelligent human-machine systems and cybernetics; 2016 September 11–12; Hangzhou, China. Piscataway, IEEE Press, 2016, pp. 436–439.
[4] J. Qi, J. Guo, M. Wang, C. Wu, A cooperative autonomous scheduling approach for multiple earth observation satellites with intensive missions, IEEE Access. 9 (2021) 61646–61661, https://doi.org/10.1109/ACCESS.2021.3075059.
[5] P. Wang, G. Reinelt, P. Gao, Y.J. Tan, A model, a heuristic and a decision support system to solve the scheduling problem of an earth observing satellite constellation, Comput. Ind. Eng. 61 (2011) 322–335, https://doi.org/10.1016/j.cie.2011.02.015.
[6] S. Wang, L. Zhao, J. Cheng, J. Zhou, Y. Wang, Task scheduling and attitude planning for agile earth observation satellite with intensive tasks, Aerosp. Sci. Technol. 90 (2019) 23–33, https://doi.org/10.1016/j.ast.2019.04.007.
[7] P. Monmousseau, Scheduling of a constellation of satellites: Creating a mixed-integer linear model, J. Optim. Theor. Appl. 191 (2021) 846–873, https://doi.org/10.1007/s10957-021-01875-2.
[8] X. Wang, G. Wu, L. Xing, W. Pedrycz, Agile earth observation satellite scheduling over 20 years: formulations, methods, and future directions, IEEE Publications. 15 (3) (2021) 3881–3892.
[9] V. Gabrel, A. Moulet, C. Murat, V.T. Paschos, A new single model and derived algorithms for the satellite shot planning problem using graph theory concepts, Ann. Oper. Res. 69 (1997) 115–134, https://doi.org/10.1023/A:1018920709696.
[10] C.G. Valicka, D. Garcia, A. Staid, J.-P. Watson, G. Hackebeil, S. Rathinam, L. Ntaimo, Mixed-integer programming models for optimal constellation scheduling given cloud cover uncertainty, Eur. J. Oper. Res. 275 (2) (2019) 431–445.
[11] X. Chen, G. Reinelt, G. Dai, A. Spitz, A mixed integer linear programming model for multi-satellite scheduling, Eur. J. Oper. Res. 275 (2019) 694–707, https://doi.org/10.1016/j.ejor.2018.11.058.
[12] Y. He, Y. Chen, J. Lu, C. Chen, G. Wu, Scheduling multiple agile earth observation satellites with an edge computing framework and a constructive heuristic algorithm, J. Syst. Archit. 95 (2019) 55–66, https://doi.org/10.1016/j.sysarc.2019.03.005.
[13] J.-S. Pan, P. Hu, V. Snášel, S.-C. Chu, A survey on binary metaheuristic algorithms and their engineering applications, Artif. Intell. Rev. 56 (7) (2023) 6101–6167.
[14] O.E. Turgut, M.S. Turgut, E. Kırtepe, A systematic review of the emerging metaheuristic algorithms on solving complex optimization problems, Neural Comput. Appl. 35 (2023) 14275–14378, https://doi.org/10.1007/s00521-023-08481-5.
[15] Y.Q. Li, R.X. Wang, M.Q. Xu, An improved genetic algorithm for a class of multi-resource range scheduling problem, J. Astronaut. 33 (2012) 85–90, https://doi.org/10.3873/j.issn.1000-1328.2012.01.012.
[16] Z. Zixuan, G. Jian, G. Eberhard, Swarm satellite mission scheduling & planning using hybrid dynamic mutation genetic algorithm, Acta Astronaut. 137 (2017) 243–253, https://doi.org/10.1016/j.actaastro.2017.04.027.
[17] M.Y. Cheng, Q. Qian, Z.W. Ni, X.H. Zhu, Co-evolutionary particle swarm optimization for multitasking, Pattern Recogn. Artif. Intell. 31 (2018) 322–334, https://doi.org/10.16451/j.cnki.issn1003-6059.201804004.
[18] H.P. Hsu, H.H. Tai, C.N. Wang, C.C. Chou, Scheduling of collaborative operations of yard cranes and yard trucks for export containers using hybrid approaches, Adv. Eng. Inform. 48 (2021) 1–14, https://doi.org/10.1016/j.aei.2021.101292.
[19] A. Sarkheyli, A. Bagheri, B. Ghorbani-Vaghei, R. Askari-Moghadam, Using an effective tabu search in interactive resources scheduling problem for LEO satellites missions, Aerosp. Sci. Technol. 29 (1) (2013) 287–295, https://doi.org/10.1016/j.ast.2013.04.001.
[20] X. Long, S. Wu, X. Wu, Y. Huang, Z. Mu, A GA-SA hybrid planning algorithm combined with improved clustering for LEO observation satellite missions, Algorithms. 12 (2019) 231, https://doi.org/10.3390/a12110231.
[21] W.K. Wong, C.I. Ming, A review on metaheuristic algorithms: recent trends, benchmarking and applications, in: International Conference on Smart Computing & Communications (ICSCC), 2019, pp. 330–334, https://doi.org/10.1109/icscc.2019.8843624.
[22] Y. Li, X. Feng, G. Wang, D. Yan, P. Liu, C. Zhang, A real-coding population-based incremental learning evolutionary algorithm for multi-satellite scheduling, Electronics 11 (7) (2022) 1147, https://doi.org/10.3390/electronics11071147.
[23] J. Pasha, A.L. Nwodu, A.M. Fathollahi-Fard, G. Tian, Z. Li, H. Wang, M. A. Dulebenets, Exact and metaheuristic algorithms for the vehicle routing problem with a factory-in-a-box in multi-objective settings, Adv. Eng. Inform. 52 (2022) 101623, https://doi.org/10.1016/j.aei.2022.101623.
[24] H. Kim, Y.K. Chang, Mission scheduling optimization of SAR satellite constellation for minimizing system response time, Aerosp. Sci. Technol. 40 (2015) 17–32, https://doi.org/10.1016/j.ast.2014.10.006.
[25] X. Niu, H. Tang, L. Wu, Satellite scheduling of large areal tasks for rapid response to natural disaster using a multi-objective genetic algorithm, Int. J. Disaster Risk Reduc. 28 (2018) 813–825, https://doi.org/10.1016/j.ijdrr.2018.02.013.
[26] H. Mao, M. Alizadeh, I. Menache, S. Kandula, Resource management with deep reinforcement learning, in: Proceedings of the 15th ACM Workshop on Hot Topics in Networks, Association for Computing Machinery, Atlanta, Georgia, 2016, pp. 50–56. https://doi.org/10.1145/3005745.3005750.
[27] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Netw. 61 (2015) 85–117, https://doi.org/10.1016/j.neunet.2014.09.003.
[28] H.V. Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double Q-learning, Comput. Ence (2015) https://doi.org/10.48550/arXiv.1509.06461.
[29] I. Bello, H. Pham, Q.V. Le, et al., Neural combinatorial optimization with reinforcement learning, ICLR 2017 (2017), https://doi.org/10.48550/arXiv.1611.09940.
[30] Y.Q. Li, R.X. Wang, M.Q. Xu, Rescheduling of observing spacecraft using fuzzy neural network and ant colony algorithm, Chin. J. Aeronaut. 27 (2014) 678–687, https://doi.org/10.1016/j.cja.2014.04.027.
[31] J.i. Lu, Y. Chen, R. He, A learning-based approach for agile satellite onboard scheduling, IEEE Access 8 (2020) 16941–16952, https://doi.org/10.1109/ACCESS.2020.2968051.
[32] Y. Huang, Z. Mu, S. Wu, B. Cui, Y. Duan, Revising the observation satellite scheduling problem based on deep reinforcement learning, Remote Sens. 13 (2021) 2377, https://doi.org/10.3390/rs13122377.
[33] Y. He, L. Xing, Y. Chen, W. Pedrycz, L. Wang, G. Wu, A generic Markov decision process model and reinforcement learning method for scheduling agile earth observation satellites, IEEE Trans. Syst., Man, Cybernet.: Syst. 52 (3) (2022) 1463–1474, https://doi.org/10.1109/TSMC.2020.3020732.
[34] M. Chen, Y.N. Chen, Y.W. Chen, et al., Deep reinforcement learning for agile satellite scheduling problem, in: 2019 IEEE Symposium Series on Computational Intelligence (SSCI), 2019, pp. 126–132. https://doi.org/10.1109/SSCI44817.2019.9002957.

[35] Y.F. Ma, F.Y. Zhao, X. Wang, et al., Satellite earth observation task planning method based on improved pointer networks, J. ZheJiang Univ. (Eng. Sci.) 55 (2) (2021) 395–401, https://doi.org/10.3785/j.issn.1008-973X.2021.02.020.
[36] Z. Wen, L.u. Li, J. Song, S. Zhang, H. Hu, Scheduling single-satellite observation and transmission tasks by using hybrid Actor-Critic reinforcement learning, Adv. Space Res. 71 (9) (2023) 3883–3896.
[37] Y. Huang, Z. Mu, S. Wu, B. Cui, Y. Duan, Revising the observation satellite scheduling problem based on deep reinforcement learning, Rem Sens 13 (12) (2021) 2377.
[38] Y. Song, J. Ou, P.N. Suganthan, W. Pedrycz, Q. Yang, L. Xing, Learning adaptive genetic algorithm for earth electromagnetic satellite scheduling, IEEE Trans. Aerosp. Electron. Syst. 59 (6) (2023) 9010–9025, https://doi.org/10.1109/TAES.2023.3312626.
[39] Y. Chen, X. Shen, G. Zhang, Z. Lu, Multi-objective multi-satellite imaging mission planning algorithm for regional mapping based on deep reinforcement learning, Rem. Sens. 15 (2023) 3932. https://doi.org/10.3390/rs15163932.
[40] Y. Song, L. Wei, Q. Yang, J. Wu, L. Xing, Y. Chen, RL-GA: a reinforcement learning-based genetic algorithm for electromagnetic detection satellite scheduling problem, Swarm Evolution. Comput. 77 (2023) 101236, https://doi.org/10.1016/j.swevo.2023.101236.
[41] A. Herrmann, H. Schaub, Reinforcement learning for the agile earth-observing satellite scheduling problem, IEEE Trans. Aerosp. Electron. Syst. 59 (5) (2023) 5235–5247, https://doi.org/10.1109/TAES.2023.3251307.
[42] L. Ren, J. Li, X. Ning, Hierarchical reinforcement-learning for real-time scheduling of agile satellites, IEEE Access 99 (2020) 220523–220532, https://doi.org/10.1109/ACCESS.2020.3040748.
[43] X.L. Bao, S.M. Zhang, X.Y. Zhang, An effective method for satellite mission scheduling based on reinforcement learning, in: 2020 Chinese Automation Congress (CAC 2020), 2020, pp. 4037–4042, https://doi.org/10.1109/CAC51589.2020.9327581.
[44] W. Haijiao, Y. Zhen, Z. Wugen, L. Dalin, Online scheduling of image satellites based on neural networks and deep reinforcement learning, Chin. J. Aeronaut. 32 (4) (2019) 1011–1019, https://doi.org/10.1016/j.cja.2018.12.018.
[45] J.T. Lam, F. Rivest, J. Berger, Deep reinforcement learning for multi-satellite collection scheduling, in: 8th International Conference on the Theory and Practice of Natural Computing (TPNC), 2019, https://doi.org/10.1007/978-3-030-34500-6_13.
[46] J. Ou, L. Xing, F. Yao, M. Li, J. Lv, Y. He, Y. Song, J. Wu, G. Zhang, Deep reinforcement learning method for satellite range scheduling problem, Swarm Evolution. Comput. 77 (2023) 101233, https://doi.org/10.1016/j.swevo.2023.101233.
[47] L. Ren, X. Ning, Z. Wang, A competitive Markov decision process model and a recursive reinforcement-learning algorithm for fairness scheduling of agile satellites, Comp. Indust. Eng. 169 (2022) 108242, https://doi.org/10.1016/j.cie.2022.108242.
[48] L.i. Dalin, W. Haijiao, Y. Zhen, G.u. Yanfeng, S. Shi, An online distributed satellite cooperative observation scheduling algorithm based on multiagent deep reinforcement learning, IEEE Geosci. Sens. Lett. 18 (11) (2021) 1901–1905, https://doi.org/10.1109/LGRS.2020.3009823.
[49] X. Zhao, Z. Wang, G. Zheng, Two-phase neural combinatorial optimization with reinforcement learning for agile satellite scheduling, J. Aerosp. Inf. Syst. 17 (7) (2020) 346–357, https://doi.org/10.2514/1.I010754.
[50] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, P.S. Yu, A comprehensive survey on graph neural networks, IEEE Trans. Neural Netw. Learn. Syst. 32 (1) (2021) 4–24, https://doi.org/10.1109/TNNLS.2020.2978386.
[51] J. Bruna, W. Zaremba, A. Szlam, et al., Spectral networks and locally connected networks on graphs, Comput. Sci. (2013) https://arxiv.org/abs/1312.6203.
[52] T.N. Kipf, M. Welling, Variational graph auto-encoders, 2016, https://doi.org/10.48550/arXiv.1611.07308.
[53] P. Velikovi, G. Cucurull, A. Casanova, et al., Graph attention networks, in: ICLR 2018, 2018, https://doi.org/10.48550/arXiv.1710.10903.
[54] B. Ybc, B. Ala, B. Apa, Machine learning for combinatorial optimization: a methodological tour d'horizon, Eur. J. Oper. Res., ScienceDirect 290 (2020) 405–421, https://doi.org/10.1016/j.ejor.2020.07.063.
[55] N.A. Asif, Y. Sarker, R.K. Chakrabortty, M.J. Ryan, M.H. Ahamed, D.K. Saha, F. R. Badal, S.K. Das, M.F. Ali, S.I. Moyeen, M.R. Islam, Z. Tasneem, Graph Neural Network: a comprehensive review on non-euclidean space, IEEE Access 9 (2021) 60588–60606, https://doi.org/10.1109/ACCESS.2021.3071274.
[56] T. Gaudelet, B. Day, et al., Utilising graph machine learning within drug discovery and development, Brief. Bioinf. (2021), https://doi.org/10.48550/arXiv.2012.05716.
[57] I. Titov, M. Welling, M. Schlichtkrull, et al., Modeling relational data with graph convolutional networks, in: 5th Conference on Semantic Web Challenges (SemWebEval Challenge) at 15th Extended Semantic Web Conference (ESWC), 2018, https://doi.org/10.48550/arXiv.1703.06103.
[58] W.L. Hamilton, R. Ying, J. Leskovec, Inductive representation learning on large graphs, NIPS 2017 (2017), https://doi.org/10.48550/arXiv.1706.02216.
[59] P. Veličković, G. Cucurull, A. Casanova, et al., Graph attention networks, ICLR 2018 (2018), https://doi.org/10.48550/arXiv.1710.10903.
[60] B. Toaza, D. Esztergár-Kiss, A review of metaheuristic algorithms for solving TSP-based scheduling optimization problems, Appl. Soft Comput. 148 (2023) 110908, https://doi.org/10.1016/j.asoc.2023.110908.
[61] F. Glover, G. Kochenberger, Y. Du, Quantum bridge analytics I: a tutorial on formulating and using QUBO models, 4OR-Q, J Oper Res. 17 (2019) 335–371, https://doi.org/10.1007/s10288-019-00424-y.
[62] L. Wei, Y. Chen, M. Chen, Y. Chen, Deep reinforcement learning and parameter transfer based approach for the multi-objective agile earth observation satellite scheduling problem, Appl. Soft Comput. 110 (2021) 107607, https://doi.org/10.1016/j.asoc.2021.107607.