Front. Eng. Manag. 2025, 12(4): 1157–1174
https://doi.org/10.1007/s42524-025-4098-y

RESEARCH ARTICLE

Cheng CHEN, Lei LI, Yonghao DU, Feng YAO, Lining XING

A hybrid learning-assisted multi-parallel algorithm for a large-scale satellite-ground networking optimization problem

Abstract The rapid expansion of satellite Internet deployments, driven by the rise of Space-Ground Integration Network (SGIN) construction, has led to a significant increase in satellite numbers. To address the challenge of efficient networking between large-scale satellites and limited ground station resources, this paper presents a hybrid learning-assisted multi-parallel algorithm (HLMP). The HLMP features a multi-parallel solving and deconflicting framework, a learning-assisted metaheuristic (LM) algorithm combining reinforcement learning (RL) and Tabu simulated annealing (TSA), and a linear programming (LP) exact-solving algorithm. The framework first divides the problem into parallel sub-problems based on the time domain, then applies LM and LP to solve each sub-problem in parallel. LM uses LP-generated scheduling results to improve its own accuracy. The deconflicting strategy integrates and refines the planning results from all sub-problems, ensuring an optimized outcome. HLMP advances beyond traditional task-driven satellite scheduling methods by offering a novel approach for optimizing large-scale satellite-ground networks under the new macro paradigm of "maximizing linkage to the greatest extent feasible." Experimental cases involving up to 1,000 satellites and 100 ground stations highlight HLMP's efficiency. Comparative experiments with other metaheuristic algorithms and the CPLEX solver further demonstrate HLMP's ability to generate high-quality solutions more quickly.

Keywords satellite–ground networking, multi-parallel framework, metaheuristics, reinforcement learning, linear programming

Received Jun. 5, 2024; revised Aug. 22, 2024; accepted Sep. 23, 2024

Cheng CHEN, Lei LI, Yonghao DU (✉), Feng YAO
College of Systems Engineering, National University of Defense Technology, Changsha 410073, China
E-mail: duyonghao15@163.com

Lining XING
School of Electronic Engineering, Xidian University, Xi'an 710075, China

This research was supported by the National Natural Science Foundation of China (Grant Nos. 72201272 and U23B2039), the National Fundamental Research Project, China (Grant No. 2023-JCJQ-QT-042), the Science Foundation of National University of Defense Technology, China (Grant No. ZK22-48).

1 Introduction

Satellite Internet has seen tremendous development growth-over-the-recent years (Kodheli et al., 2021; He et al., 2023; Lin et al., 2024a). With low latency, cost-effectiveness, wide coverage, very high commercial potential, military applications, and resilience to natural calamities, satellite Internet has made itself increasingly important to such sectors as network communication, disaster emergency response, environmental monitoring, telemedicine, and online education (Wang et al., 2023; Lin et al., 2024b). As a result, it has become an attractive strategic initiative for various countries across the world (Del Portillo et al., 2019). However, the rapid growth of satellite Internet has brought about a tremendous increase in the number of satellites, compounding the pressure on ground station resources, which are already somewhat limited for the speedy reception and transmission of vast data that satellite systems generate around the globe. Thus, the need for efficient satellite-ground networking has become even more urgent, for this will be among the core strategic support facilities within the Space-Ground Integration Network (SGIN) in the foreseeable future. The satellite-ground networking optimization problem (SGNOP) is essentially a large-scale combinatorial optimization problem. The SGNOP essentially aims to connect satellite networks with ground stations for their Internet link while maximizing the operational efficiency or functionalities of ground stations via adequacy of timing windows. Very few works, indeed, touch upon this SGNOP subject as yet. It needs to be, however, quite aggressively noted that, since the nature of SGNOP resembles some characteristics of the satellite range

scheduling problem (SRSP), which works on scheduling the execution for tasks based on the visibility between satellites and ground stations (Luo et al., 2017), this provides a mindset and rationale for comparing SGNOP with SRSP. Hence, some insights into SGNOP could be got in comparisons with SRSP. Figure 1 shows the similarities and differences between SGNOP and SRSP.

Both SGNOP and SRSP use planning based on time windows. Whereas concentrated on specific tasks in SRSP, emphasis is placed on this task allocation and execution; SGNOP addresses constellations with a focus on constellation-ground station connections. This satellite-ground architecture serves as the basis for the SRSP to fulfill its function to some extent. Second, SGNOP has several key characteristics: 1) Large in scale: The SRSP is task-oriented; as per daily task requirements, a satellite has to complete several measurements and controls in a designated time window. In contrast, SGNOP views each time window as a separate task with the maximization of the windows to effectively connect the constellation with the ground stations. Thus, the solution space for SGNOP is greatly wider than that of SRSP. 2) Time-decomposable: The characteristics of SGNOP give the opportunity to divide the scheduling periods into many time segments by carrying on with identical high-level goals throughout all time segments. This renders it natural for the problem to be decomposed into naturally interacting sub-problems ; such interaction is minimal among them. The SRSP problem, on the other hand, cannot be very well decomposed into time segments due to the constraints imposed by some specific tasks' requirements. 3) The window is portable: In SGNOP, the feasibility of handover of link time windows through feeder antennas is possible but not taken into consideration in classic SRSP problems. The existence of this characteristic complicates the already large-scale SGNOP even further.

Given the aforementioned qualities in terms of size and decomposability of SGNOP, we discuss the multi-parallel solving three-dimension feasibility: 1) Problem parallelization: The large-scale SGNOP problem can be divided into independent parallel sub-problems by splitting the planning period into several sub-periods with an assigned duration, which will decrease the complexity and computational difficulty of SGNOP. 2) Algorithm parallelization: We can think of running more than one algorithm to solve the sub-problems in parallel based on the current computing power and advanced algorithm techniques (Yao et al., 2023). Effective solution processing can therefore be ensured by this tackling of two dimensions, which can lift up the stability and accuracy of the solution approach.

Current research on the SGNOP is independently in its infancy. So far, no mature algorithms are resuscitated to tackle SGNOP. Therefore, one might suggest that due to intrinsic similarities between SGNOP and SRSP algorithms that have surfaced on SRSP may serve as useful reference to SGNOP.

These algorithms can be generally divided into two categories: 1) exact algorithms and 2) metaheuristic algorithms. SGNOP is characterized by having robust linear constraints; therefore, this may offer an option of solving them by exact algorithms (Liu et al., 2019). The existing studies (Marinelli et al., 2011; Yu et al., 2017; Wang and Lu, 2019; Su et al., 2023) indicate that the application of exact algorithms in mathematical programming models can identify optimal solutions for small-scale problems with great efficiency (Zhou et al., 2020). However, with an increase in the problem size, the computational costs increase rapidly; thus, a solution cannot be found in a reasonable time span (Wang et al., 2021a; Du et al., 2022). Metaheuristic algorithms, on the other hand, were used in large-scale SRSP problems with great success.

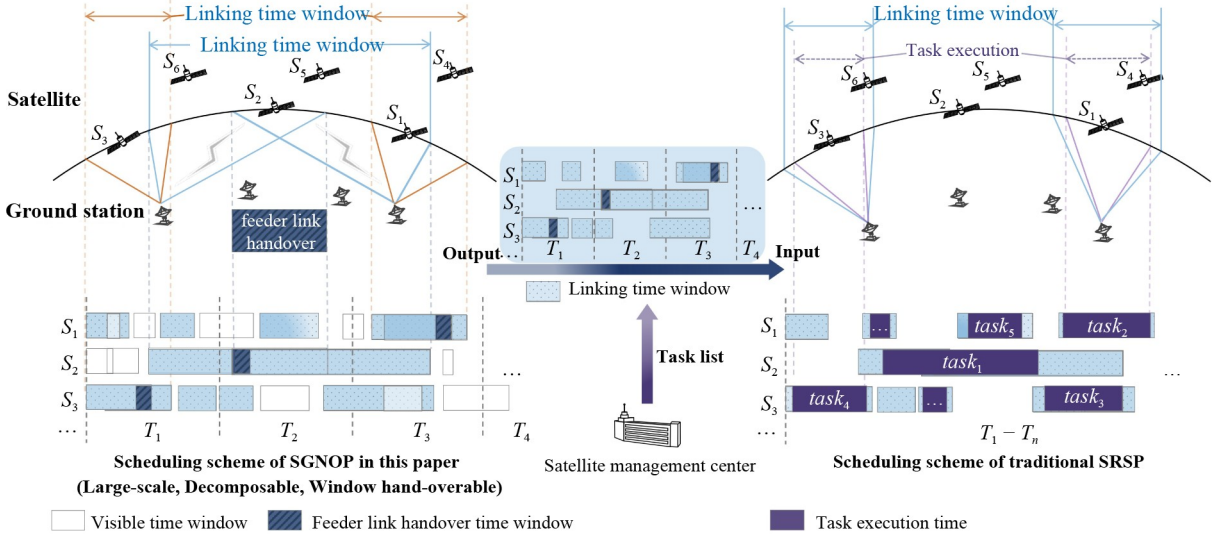

Fig. 1 The similarities and differences between SGNOP and SRSP.

For example, Du et al. (2019) employed local search metaheuristics to solve conflicts and to achieve balanced loads, greatly improving the optimization of multi-objective SRSP. Likewise, Song et al. (2019) developed a hybrid algorithm that combines an improved genetic algorithm with a local search method to rapidly enhance the quality of SRSP schemes. Liu et al. (2022) utilized a simulated annealing algorithm with a Tabu list to solve the downlink scheduling problem for multiple satellites and ground stations. Metaheuristic algorithms guarantee that the solutions of relatively reasonable quality can be obtained within a comparatively shorter time for large-scale combinatorial optimization problems. Although they achieve a much greater solution efficiency compared to exact algorithms, their solution quality evaluation remains a problematic issue due to their nature along with the fact that they fall back into local optima (Zhu et al., 2023). In light of these two approaches, on one side precise algorithms are inefficient with larger problem domains and on the other side is often halted due to premature convergence, Puchinger and Raidl (2005) conducted a systematic investigation into the integration of exact solution techniques with metaheuristic algorithms. Additionally, Hooker (2015) also explored the similarities between exact and heuristic algorithms and made recommendations on transitioning from exact to heuristic modes as more complex problem scenarios unfold. Thus, we describe a joint method based on integrating the reliable and quick solutions of SGNOP that will use exact linear programming (LP) algorithms and metaheuristic algorithms.

In recent times, increasing numbers of workers have taken up artificial intelligence (AI) to provide an enhancement for algorithms' efficiencies (Wang et al., 2020; 2021b; Li et al., 2024). AI methods have recently proved efficient in solving some scheduling problems related to satellites (Chen et al., 2024). For establishing a probability prediction model that assigns to satellites tasks that are most likely to be scheduled, Du et al. (2020; 2021) trained neural networks with historical data. Song et al. (2023a) developed a cluster-based genetic algorithm for solving the SRSP that uses the k-means clustering method to support the population for the evolutionary process. Ren et al. (2022) proposed a recursive approach incorporated with reinforcement learning algorithms with block coding to solve the problem of fairness scheduling in satellites. Wu et al. (2022; 2023) performed pattern mining to extract modest frequent patterns from an elite set to form a new solution. Furthermore, Song et al. (2023b) combined reinforcement learning (RL) with genetic algorithms to discover effective evolutionary operators, guiding the population search process. Li et al. (2023) modeled reinforcement learning into single-objective multitask optimization problems to dynamically change the assigned parameters of mating probabilities with regard to random mating across tasks, thereby facilitating the autonomous transfer of adaptive learning. By merging the robust generalization capabilities of AI methods with the domain-specific knowledge inherent in metaheuristics (Shiue et al., 2018; Zhao et al., 2023), the dual capability of "autonomy and scalability" can be achieved.

The present paper, by building on the merits of the myriad aforementioned methodologies and in the spirit of their interrelation, has come forth with the initiation of a learning-assisted hybrid multi-parallel algorithm (HLMP) with effective framework to solve the SGNOP. The distinguished contributions of this paper can be furthered enumerated as:

1) A 0–1 integer linear programming (ILP) model is started with to appropriately set forth the SGNOP. Constructing links are represented as variables with values corresponding to 0–1 in order to determine if links would be made between the respective ground station and satellite. This helps solve the problem in an easier way and further optimize the SGNOP.

2) A multi-parallel algorithm integrated for the assignment and resolution of the SGNOP efficiently develops a decomposition whereby each subproblem is solved by using the LP exact-solving algorithm and learning-assisted metaheuristics (LM) in parallel. A conflict resolution strategy is applied to include all those in other states derived from all subproblems and to resolve any conflicts that may arise. The RL-TSA uses a reinforcement-based neighborhood selection to perform rapid optimization by employing reinforcement learning to further aid the effective selection of those neighborhoods.

Section 2 describes the other features of the SGNOP and provides the description of the ILP model. Multi-Parallel solving and deconflicting framework development are indicated in Section 3 along with their little but significant features in the LM and LP approaches. Experiments were conducted with detailed reporting of results in Section 4. The discussion and conclusion will follow in Sections 5 and 6, respectively.

2 Problem description

In this section, the SGNOP is illustrated and explained using an example with one satellite and two ground stations to make it easy to understand. Then, the variables and the full ILP model are provided.

2.1 Preliminaries

Before modeling the SGNOP, several preliminary concepts must be established. These preliminaries are outlined below:

1) Ground resources and satellites have a defined visibility period, limited by the curvature of the Earth and the linear propagation of radio waves. We define this

visibility period as the “visible time window” or visible arc, which includes the time from the initial tracking to its conclusion. During this interval, connections between satellites and ground stations can be established, as illustrated in Fig. 1. Excluding the preparation time required before linking the satellite to the ground station and the buffer time following the establishment of the link, the resulting duration is termed the “link time window” or link arc.

2) In this study, each satellite is outfitted with two feeder antennas. Three different scenarios arise when a satellite sequentially passes over two ground stations. As depicted in Fig. 2(a), if two link arcs overlap and the duration of their intersection exceeds a specified threshold, the feeder antennas will become operational. By alternating between the feeder antennas, continuous connections between the satellite and both ground stations can be maintained, facilitating a smooth

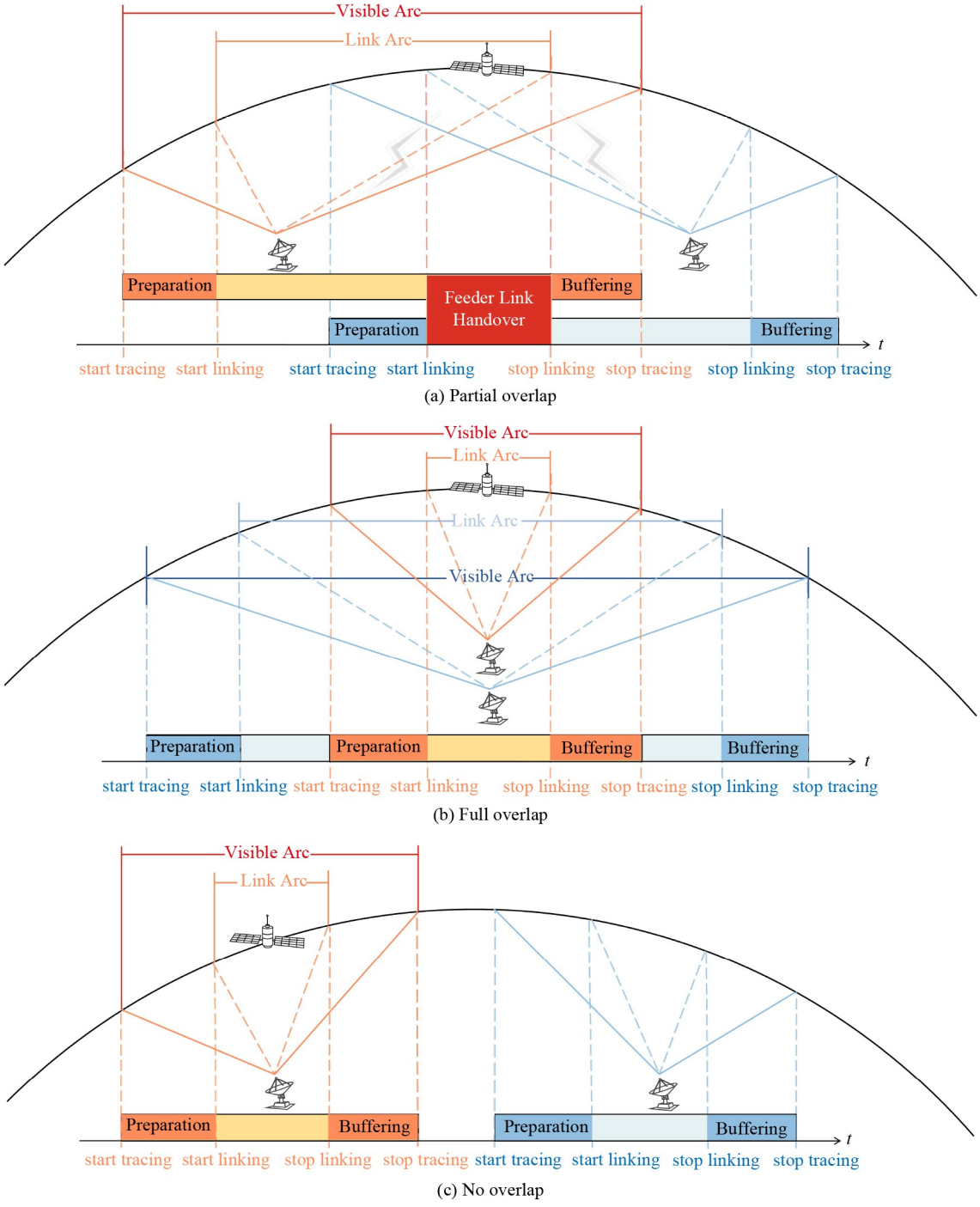

Fig. 2 Visible arcs, link arcs, and feeder link handover schematic diagram.

handover between the two arcs. Figure 2(b) illustrates an inclusion relationship between the two link arcs, while Fig. 2(c) shows a scenario where no overlap exists between them.

To better illustrate the complexity of the problem and facilitate further modeling, we have defined the boundaries of the SGNOP. The SGNOP is defined as follows: Given a set of satellites $S=\{s_1,s_2,s_3,\dots\}$, a set of ground tracking stations $G=\{g_1,g_2,g_3,\dots\}$, a set of satellite-to-ground link ranges $R=\{r_1,r_2,r_3,\dots\}$, and their corresponding time window set $T^{Win}=\{t_1^{Win},t_2^{Win},t_3^{Win},\dots\}$. Each range r_i includes the source satellite s_i, the ground tracking station g_i, the time window t_i^{Win}, the tracking duration T_i^{tDur}, and the linking duration T_i^{lDur}. Each satellite is equipped with two feeder antennas, and the constraint on minimum feeder handover time α must be satisfied if the situation depicted in Fig. 2(a) occurs. The optimization objective is to maximize the total link duration (or maximize the number of total weighted scheduled ranges).

We establish that the SGNOP is NP-hard by referencing a well-known NP-complete problem known as the Multiple Resource Range Scheduling (MuRRS) problem (Barbulescu et al., 2004), which is defined as follows: A problem instance consists of n task requests where the objective is to minimize the number of unscheduled tasks. A task request T_i, $0<i<n$, specifies both a required processing duration T_i^{Dur} and a time window T_i^{Win} within which the duration must be allocated. Each task request T_i additionally specifies a resource $R_i\in[1\dots m]$, where m is the total number of resources available. Further, T_i may optionally specify $j\geqslant 0$ additional (R_i, T_i^{Win}) pairs, each identifying a particular alternative resource and time window for the task. Concurrency and preemptions are not allowed. For each task request T_i, we can discretize T_i^{Win} into l windows of duration $T_i^{Dur},[t_{i_1}^{Win},t_{i_2}^{Win},\cdots,t_{i_k}^{Win},\dots,t_{i_l}^{Win}],0<k\leqslant l$, thus disassembling T_i into l subtasks $[t_{i_1},t_{i_2},\dots,t_{i_k},\dots,t_{i_l}]$, and each subtask is in conflict with one another.

We consider a special case (subset) of SGNOP with the following characteristics:

1) **No Feeder Antenna Handover:** A satellite carries only one common antenna. It does not consider the condition of the feeder antenna handover.

2) **No Preparation Time or Buffering Time:** $T_i^{tDur}=T_i^{lDur}$.

3) **Equal Range Weights**: All ranges are assumed to have equal weight, that is, all arcs have the same unit duration.

We refer to this problem as SGNOP with equal range weights, no feeder antenna handover, and no preparation or buffering time (SGNOP-FPE). The objective of the optimization is to maximize the total link duration, which is equivalent to minimizing the number of unscheduled ranges.

Proof: The SGNOP is NP-hard. We assume that the total amount of scheduled time and the number of tasks or ranges to be scheduled are unbounded. The MuRRS is NP-complete and can be reduced to a SGNOP-FPE as follows: The subtask in the MuRRS is equivalent to the range in the SGNOP-FPE: $t_{i_k}=r_i$. Both have a duration: $T_i^{Dur}=T_i^{tDur}$. The time window of subtask t_{i_k} in the MuRRS is equivalent to the time window of range r_i in the SGNOP-FPE: $t_{i_k}^{Win}=t_i^{Win}$. Both problems count the number of unscheduled tasks or ranges. This completes the reduction. The SGNOP-FPE is NP-hard. Because the set of SGNOP-FPE is a subset of SGNOP, the general SGNOP problem is NP-hard.

2.2 Assumptions and symbols

A few reasonable assumptions are necessary to support the preliminary framework and to establish a robust model for the large-scale SGNOP examined in this paper:

1) The static scheduling environment remains constant throughout the entire scheduling period, without accounting for the emergence and influence of dynamic or uncertain factors.

2) The communication link between the ground station and the satellite is uninterrupted and remains established from initiation to conclusion.

3) The effects of resource equipment failures are not considered.

4) Satellite energy levels are adequate to ensure the successful completion of the link.

For clarity, the definitions of the relevant symbols used in the model are provided in Table 1.

2.3 Integer linear programming modeling

The core of the large-scale SGNOP involves addressing the challenges of resource allocation and conflict resolution in the context of feeder antennas. Specifically, this addresses scenarios where multiple ground station antennas can simultaneously connect with the same satellite or when a single ground station antenna has visibility over multiple satellites. The objective is to allocate ground resources effectively across different satellites, thereby maximizing the utilization of these resources and extending the duration of satellite-ground connections. By leveraging information from the visible arcs, this optimization model prioritizes these visible arcs over the satellites and ground station resources, streamlining the model and reducing the complexity of parameters involved.

2.3.1 Decision variables

We use the result of link-building—whether the arc is activated—to directly and clearly infer whether a satellite and ground station antenna have established a connection. The decision variable is designed as follows:

Table 1 Description of variables

Notation	Description
R	Set of visible arcs needed to be scheduled within a given time horizon, $R = \{r_1, r_2, ..., r_i, ..., r_n\}$
s_i	Link-building satellite of visible arc r_i
a_i	Link-building ground station antenna of visible arc r_i
b_i^T	Trace beginning time of visible arc r_i
e_i^T	Trace ending time of visible arc r_i
b_i^L	Link beginning time of visible arc r_i
e_i^L	Link ending time of visible arc r_i
τ	Attitude conversion time of ground stations
α	Feeder link handover time between two satellite antennas
β	Attitude conversion time of satellites
M	A sufficiently large positive integer
x_i	Equal 1 if r_i is activated, and 0 otherwise.
$y_{i,j}$	Equal 1 if b_i^T is not later than b_j^T, and 0 otherwise.

Let x_i determine whether visible arc r_i is activated between s_i and a_i or not by 1 or 0, respectively.

Let $y_{i,j}$ be 1 if b_i^T is not later than b_j^T, and otherwise 0. The decision variables can be expressed as

$$x_i \in \{0,1\},\ \forall r_i \in R, \tag{1}$$

$$y_{i,j} \in \{0,1\},\ \forall r_i, r_j \in R, \tag{2}$$

2.3.2 Constraints

Based on the preceding explanations and variables, the constraints governing the large-scale SGNOP presented in this paper are formulated as follows:

Ground station antenna conversion constraint. Each ground station antenna is permitted to track only one satellite at any given moment, while also adhering to the attitude conversion time constraint required for tracking two consecutive satellites:

$$\begin{aligned} e_i^T + \tau \leqslant b_j^T + (2 - x_i - x_j)M + (1 - y_{i,j})M, \\ \forall r_i, r_j \in R,\ a_i = a_j, \end{aligned} \tag{3}$$

where M is a big integer. The M-method is introduced to indicate that the above constraints only work when both arcs are activated to establish links, and the condition that the starting time of the visible arc r_i is not later than that of the visible arc r_j is satisfied. As Eq. (3) shows, e_i^T represents the trace ending time of the visible arc r_i, b_j^T represents the trace beginning time of the visible arc r_j, and τ is the attitude conversion time of ground stations, and the condition that the interval between e_i^T and b_j^T is not less than τ is satisfied.

Satellite feeder antenna conversion constraints. There are three visible scenarios generated during satellite transit, as depicted in Figs. 1(a)–(c).

1) If there is a partial overlap between the two link arcs—that is, $b_i^L < b_j^L < e_i^L < e_j^L$—then the coverage time of the link arcs corresponding to the two antennas of the same satellite should not be less than the time required for feeder link handover.

$$\begin{aligned} b_j^L + \alpha \leqslant e_i^L + (2 - x_i - x_j)M + (1 - y_{i,j})M, \\ \forall r_i, r_j \in R,\ s_i = s_j,\ b_j^L < e_i^L < e_j^L, \end{aligned} \tag{4}$$

2) If the two link arcs belong to the inclusion relation—that is, $[b_j^L, e_j^L] \subseteq [b_i^L, e_i^L]$—then the arcs with a shorter time period are discarded.

$$\begin{aligned} x_j \leqslant (2 - x_i - x_j)M + (1 - y_{i,j})M, \\ \forall r_i,\ r_j \in R,\ s_i = s_j, e_i^L \geqslant e_j^L. \end{aligned} \tag{5}$$

3) If there is no overlap between the two link arcs—that is, $e_i^L \leqslant b_j^L$—then the feeder link handover is not performed and the satellite's attitude conversion time between the two adjacent visible arcs must be checked.

$$\begin{aligned} e_i^L + \beta \leqslant b_j^L + (2 - x_i - x_j)M + (1 - y_{i,j})M, \\ \forall r_i, r_j \in R,\ s_i = s_j,\ e_i^L \leqslant b_j^L \end{aligned} \tag{6}$$

$$y_{i,j} + y_{j,i} \leqslant 1,\ \forall r_i,\ r_j \in R. \tag{7}$$

2.3.3 Objective function

Maximizing the link duration time of satellite-ground networking is determined as the objective in this paper, as shown in Eq. (8):

$$\text{Max} \sum_{i=1}^{n} x_i (e_i^L - b_i^L). \tag{8}$$

Although multi-objective optimization, which yields Pareto solutions, is a viable methodology to address various objectives, it is not employed here for two key reasons: First, this single objective aligns closely with the management agency's principle of "maximizing linkage to the greatest extent feasible." Secondly, multi-objective optimization could lead to extended computation times for Pareto solutions, which may not adequately support the rapid-response large-scale SGNOP solving emphasized in this study.

3 A hybrid learning-assisted multi-parallel algorithm

This section introduces the HLMP, which comprises a multi-parallel solving and deconflicting framework, a fast-solving algorithm utilizing LM with Tabu simulated annealing (TSA), and a LP exact-solving algorithm. The central premise of HLMP is to decompose the problem into several parallel sub-problems from a time-domain perspective, subsequently employing LP and LM in parallel for the resolution of each sub-problem. The exact solutions provided by LP serve to validate the quality of the LM solutions. The deconflicting strategy integrates the planning results derived from all sub-problems, addressing any conflicts to ensure an optimized and coherent outcome. The forthcoming sections will outline the framework and the essential details required for implementing this approach.

3.1 Framework of HLMP

The framework of HLMP designed for the efficient resolution of SGNOP is illustrated in Fig. 3. It is divided into the following three steps:

Step 1: Problem Parallelization. The large-scale SGNOP is characterized by an extended planning period and the relative independence of various sub-problems. To address this, we divide the SGNOP into parallel, independent sub-problems by segmenting the planning period into several sub-periods of specified duration, which can be determined based on the total length of the planning period or the specific needs of users.

Step 2: Algorithm Parallelization. Both LP and LM algorithms are employed concurrently to tackle all sub-problems. If the solution time for LP is shorter than that of LM, or if LP's solution time exceeds that of LM but remains within an acceptable threshold T for users, the LP solution is incorporated into the overall solution set. Conversely, if these conditions are not met, the LM solution is included. We can assess the performance of LM while utilizing LP as a supportive mechanism by incorporating LP into the HLMP. This setup enables both algorithms to operate simultaneously within a reasonable time frame, achieving a level of performance that exceeds the capabilities of either algorithm when used independently. Given the characteristics of the large-scale SGNOP, the LM algorithm combines RL with TSA, while the LP algorithm employs the CPLEX engine for problem-solving.

Step 3: Sub-Problem Deconfliction. Upon completion of Step 2, a deconflicting strategy is applied to the boundaries of the solutions from all adjacent sub-problems within the obtained solution set. Following this, the individual sub-problems are merged to generate an efficient and stable planning scheme.

In particular, we integrated LP and LM in **Step 2** due to their unique advantages:

Linear Programming (LP). First, LP is adept at

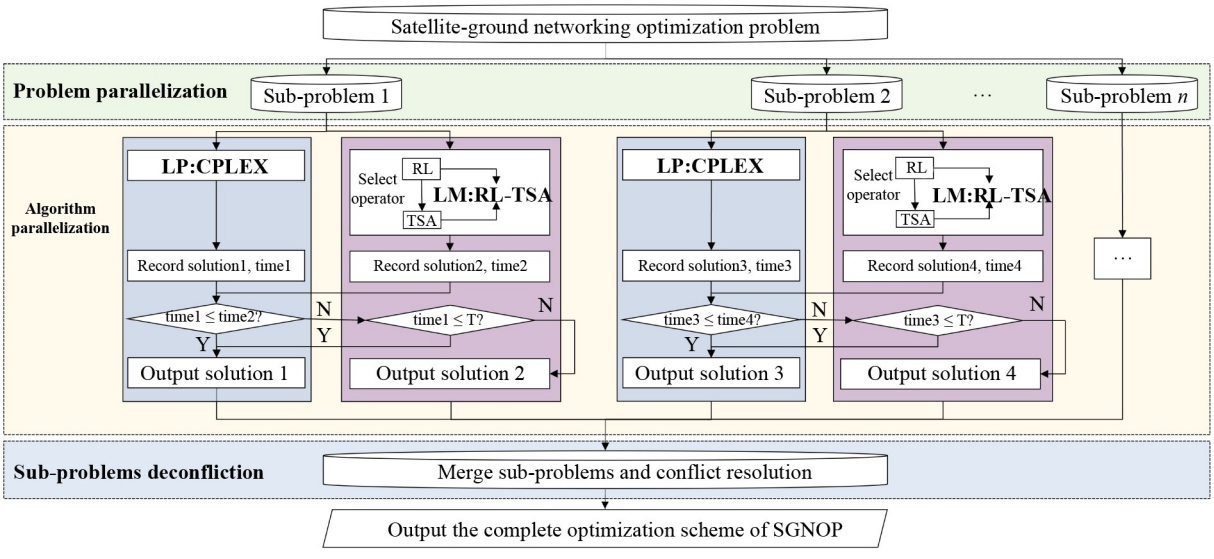

Fig. 3 HLMP core framework.

identifying optimal solutions for small-scale problems, making it particularly effective for resolving small-scale sub-problems post-segmentation, as well as for deconflicting sub-problems in Step 3 of the framework. Additionally, the precise solutions generated by LP serve as benchmarks for evaluating the performance of the LM that we developed. However, LP becomes less viable as the scale of the problem increases or when users impose stringent time constraints on solution delivery.

Metaheuristics. For large-scale combinatorial optimization challenges, metaheuristics can quickly yield relatively optimal solutions. They generally offer greater solving efficiency compared to LP; however, the quality of the solutions can often be challenging to assess effectively. Moreover, metaheuristics can engage in extensive invalid iterative searches, which significantly depletes computational resources.

Reinforcement Learning (RL) serves a crucial role in enhancing metaheuristic algorithms by efficiently directing the search process through the selection of neighborhood operators. Unlike other methods, which may rely on random selection or equal probability—thereby failing to effectively evaluate the quality of various neighborhood operations—RL offers a more nuanced approach. Some researchers have employed adaptive large neighborhood search (Liu et al., 2017), which selects operators probabilistically based on their cumulative scores. However, this method does not account for changes in the environment, presenting a significant limitation. In contrast, RL can adeptly select the most appropriate operator based on the current state of the solution, thereby minimizing unnecessary iterative searches and significantly improving search efficiency.

3.2 LM/RL-TSA

In metaheuristic algorithms, techniques such as genetic algorithms, ant colony algorithms, and other swarm search methodologies often exhibit a high dependency on parameters and stringent implementation requirements, despite their success across various fields. Additionally, processes like the encoding and decoding in evolutionary algorithms—such as genetic algorithms—demand substantial computational resources, particularly as problem size increases, which can inhibit overall algorithm performance and stability in achieving satisfactory solutions. Local search algorithms, like simulated annealing (SA) and tabu search (TS), stand out for their ease of configuration, straightforward implementation, moderate CPU time requirements, and effectiveness in addressing large-scale problems.

Simulated annealing, in particular, demonstrates robust local search capabilities and a notable ability to escape local optima. Consequently, this paper employs TSA as the foundational metaheuristic for addressing large-scale SGNOP. However, in TSA, neighborhood operation selection is typically conducted with equal probability, which does not adequately differentiate the quality of diverse neighborhood operations. Here, RL presents a compelling solution to this limitation. By integrating RL into the TSA framework, we can leverage the strengths of both approaches to enhance search efficiency. In the RL-TSA model, RL initially conducts the selection of neighborhood actions during the SA process, while TS manages the control of action objects.

3.2.1 TSA

TSA represents the principal structure of the LM, as outlined in the pseudocode of Algorithm 1, with the fifth line specifically utilizing RL for the selection of neighborhood operators.

Initial solution construction. The choice of the initial solution significantly impacts both the convergence speed and the overall effectiveness of the SA algorithm. A more suitable initial solution tends to yield better convergence effect of the iterative optimization process. In this paper, we propose a greedy approach tailored to the objective function of the large-scale SGNOP. Arc segments are selected based on visible arc duration to establish connections, resulting in an initial feasible solution that meets the imposed constraints.

Tabu search strategy. The SA method has inherent limitations in memory for solution transformations during the search process, which may lead to local loops. To address this issue, a tabu list can be constructed, integrating TS with SA. This combination effectively compensates for the algorithm's lack of memory capacity in the iterative search process. The tabu list minimizes resource wastage by preventing short-term revisits to already explored and inferior solution arcs, thereby guiding the algorithm toward discovering the global optimal solution. The length of the tabu list, denoted as $|T|$, is managed using the first-in, first-out principle.

Termination condition setting. The outer loop of the TSA incorporates two termination conditions: a preset acceptable running time and a maximum prescribed number of consecutive unimproved iterations. The termination condition for the inner loop is defined by the maximum allowable number of iterations.

3.2.2 RL

RL-TSA incorporates Q-learning into the TSA framework, where the agent's decisions govern the improvement in each iteration. Q-learning steers the search process by selecting pairs of neighborhood operators. The results of neighborhood operations are utilized to calculate rewards, which subsequently update the Q-values to inform future decisions. The pseudocode for Q-learning is presented in Algorithm 2.

Algorithm 1: Reinforcement learning assisted Tabu simulated annealing (RL-TSA) algorithm
Inputs:initial solution s_0, objective function $f(s)$, operator pair set O, initial temperature T_0 anneal coefficient ΔT, empty Tabu set D^T of cancellation arc, termination condition for outer loop, and stability conditions for internal loop.
Outputs:current solution s.

$s \leftarrow s_0$
$T \leftarrow T_0$
while the termination condition is not met do
while the stability conditions are not reached at the current temperature T do
select one operator pair o $(o \in O)$ according to Q-learning
select one cancellation arc $\notin D^T$ and several recovery arcs according to o // The cancellation arc is not tabued
obtain a neighboring solution $s' \leftarrow o(s)$
if $f(s') \geq f(s)$ then
$s \leftarrow s'$
else if random (0, 1) < $exp[\ 10 * (f(s') - f(s) - 1) / T\ ]$ then
$s \leftarrow s'$ // Accept this worse solution
else
store cancellation arc into Tabu set D^T by FIFO
end if
end while
$T \leftarrow \Delta T * T$ // Perform annealing
end while
return s

Algorithm 2: Reinforcement learning
Inputs:initial solution s_0, state S, objective function f(s), operator pair set O, ε, learning rate α, discount factor γ, termination condition.
Outputs: Q-table.

$s \leftarrow s_0$
initialize Q-table
initialize S according to s
while the termination condition is not met do
choose a $(a \in O)$ using ε-greedy strategy
take action a
obtain a neighboring solution $s' \leftarrow o(s)$
$R \leftarrow f(s') - f(s)$
$S' \leftarrow$ update state according to s'
$Q(S,A) \leftarrow Q(S,A) + \alpha(R + \gamma * \max_a Q(S',a) - Q(S,A))$
$S \leftarrow S'$
$s \leftarrow s'$
end while
return Q-table

The Q-learning search is contingent solely on the current state, thereby fulfilling the criteria for constructing a Markov decision process (MDP) (Doltsinis et al., 2014). An MDP comprises four essential components: state, action, reward, and value function. In this paper, Q-learning correlates the agent's state with the fitness value, represented by two dimensions: the fitness function value and the status of improvement (improved, unchanged, or decreased). The state will be updated based on changes in the fitness value. The actions in Q-learning are defined as pairs of neighborhood operators.

To enhance the diversity of neighborhood solutions during the search process, we developed eight neighborhood operation operators based on greedy criteria and

random search strategies for the large-scale SGNOP problem. This set comprises six cancellation operators and two recovery operators. Notably, one cancellation operator is paired with one recovery operator, resulting in 12 different neighborhood structures within the operator pair set O, which expands the algorithm's effective search range. Below are the specific functions of each neighborhood operator: (1) Random Cancellation Operator: This operator randomly selects one visible arc from the feasible solution to cancel the link building. The arcs in the feasible solution are prioritized in ascending order of their link starting time, focusing on operations involving adjacent visible arcs, as illustrated in operations (2) – (5). (2) Random Cancellation Operator for Two Adjacent Arcs: This operator randomly identifies two adjacent visible arcs to cancel the link building. (3) Random Cancellation Operator for Three Adjacent Arcs: This operator randomly identifies three adjacent visible arcs to cancel the link building. (4) Shortest Arc Duration Cancellation Operator for Two Adjacent Arcs: This operator seeks out two adjacent visible arcs with the minimum combined duration for the purpose of canceling the link building. (5) Shortest Arc Duration Cancellation Operator for Three Adjacent Arcs: This operator identifies three adjacent visible arcs with the minimum combined duration to cancel the link building. (6) Longest Arc Duration Conflict Set Size Cancellation Operator: Recognizing that longer visible arcs with more conflicting arcs present greater opportunities for solution improvement, this operator sorts the arcs in the feasible solution in descending order of arc duration conflict set size and sequentially cancels one visible arc from this sorted list. (7) Conflict Set Duration Traversal Recovery Operator: Upon canceling a visible arc, this operator traverses the conflict set and selects visible arcs to establish links based on descending order of duration, ensuring no conflicts occur. (8) Conflict Set Time Series Traversal Recovery Operator: After canceling a visible arc, this operator traverses the conflict set and selects visible arcs to form links based on the order of their tracking start time, again ensuring no conflicts arise.

Q-learning is employed to select the action corresponding to the maximum Q-value in each state, replacing the conventional operator selection method found in metaheuristic algorithms. The reward for each action is calculated based on the difference in the fitness value before and after the action's execution, with the interaction between the agent and the environment evaluated through the reward function. The formula for calculating rewards is provided below:

$$R_t = f_t(S_t, A_t) - f_{t-1}(S_{t-1}, A_{t-1}), \tag{9}$$

where f_t and f_{t-1} are values of fitness at the time t and $t-1$, respectively.

The update of the Q-value is fundamental to Q-learning, a dynamic programming method grounded in the Bellman equation (Moon, 2021). This equation provides the optimal decision value based on the current state, which equates to the expected value of the optimal decision value in the subsequent state, enhanced by the immediate reward obtained in the current state. The updated formula for the Q-value is illustrated below.

$$\begin{aligned} Q_{t+1}(S_t, A_t) = Q_t(S_t, A_t) + \\ \alpha[R_t + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)], \end{aligned} \tag{10}$$

where α is the learning rate and γ is the discount factor.

Q-learning requires a balance between exploration and exploitation within the algorithm. We utilize a parameter to adjust this process, thereby preventing the algorithm from becoming trapped in a local optimum from which it is challenging to escape. When a randomly generated probability value is less than a predefined threshold, a random action is selected to produce a new solution.

The Q-learning process necessitates the completion of a search procedure, culminating in the optimal result as the final execution plan after a specified number of iterations. The Q-value may fluctuate when the agent selects an action, reflecting a neighborhood operation. We employ the number of consecutive unimproved Q-value instances as the termination evaluation criterion for the algorithm. When the count of constant, unimproved instances reaches a predetermined maximum, the search process of the algorithm is concluded.

3.3 LP and deconflicting strategy

Recent advancements in exact solving technology have significantly enhanced the performance of CPLEX in solving LP, mixed-integer programming, and related problems. During the algorithm's parallelization phase, the ILP model leverages both the RL-TSA and CPLEX engines to resolve small-scale SGNOP problems swiftly and accurately. This methodology allows the two algorithmic types to compete efficiently within a reasonably constrained timeframe.

In the sub-problems deconflicting stage, it is essential to address the conflicts at the boundaries of adjacent sub-problems. Given the minimal correlation between different sub-problems, only a limited number of conflicts typically arise at these boundaries, which can be resolved effectively using CPLEX. We define the conflict interval [*a, b*] as the fixed period surrounding the boundary of adjacent sub-problems. An activated visible arc is classified as a potential conflict arc if its tracking start or end time falls within this interval. The set of potential conflict arcs is represented by $R_c = \{r_{ck} | k = 1, 2, ..., K\}$, as depicted in Fig. 4. CPLEX is used to deconflict sub-problems by solving the set R_c.

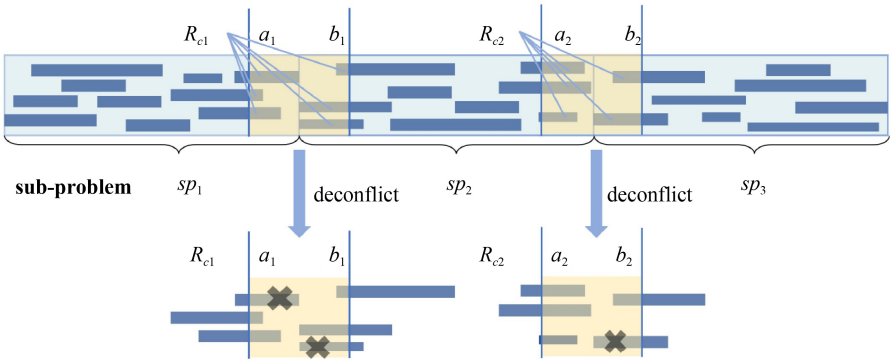

Fig. 4 Deconfliction of sub-problems at the boundaries.

4 Experimental study

4.1 Experiment settings

4.1.1 Experiment environment

All algorithms utilized in the experiment were executed using Java 17 on an Intel (R) Core (TM) i7-12700 CPU operating at 2.10 GHz, under the Windows 10 operating system, with 128 GB of RAM and 20 threads.

Experimental instances used in this study were generated using STK version 11.6.0. The input data required for scheduling was obtained through preprocessing completed with MATLAB 2018b. We developed 10 different simulation instances of constellations and ground stations that vary in scale, as shown in Table 2. As an example, the constellation comprising 1,000 satellites was modeled after the Starlink Stage I constellation, which utilizes the Walker inclined orbit configuration; the specific parameters for this configuration are outlined in Table 3. We derived instances of varying scales by adjusting the numbers of satellites and ground station antennas. The scheduling period for all instances is set at three days.

Table 2 Basic information of different instances

Instance	Number of satellites	Number of ground station antennas	Number of visible arcs
1	100	10	20,938
2	200	20	84,464
3	300	30	185,344
4	400	40	333,148
5	500	50	520,914
6	600	60	759,228
7	700	70	1,035,782
8	800	80	1,340,778
9	900	90	1,699,998
10	1000	100	2,098,264

Table 3 Constellation parameter settings of Instance 10

Parameter	Value
Orbital altitude	550
Number of satellites per orbit	40
Number of orbital planes	25
Phase factor	11
Orbit inclination	53°

4.1.2 Comparative algorithms

As the SGNOP represents a relatively novel challenge in satellite operations, there currently exists no established benchmark algorithm for direct comparison. Therefore, we selected several competitive algorithms that exhibit strong performance in combinatorial optimization and satellite scheduling, including (TS) (Glover, 1986), SA (Kirkpatrick et al., 1983), adaptive large neighborhood search (ALNS) (Liu et al., 2017), genetic algorithm (GA) (Holland, 1992), and ant colony algorithm (ACO) (Dorigo et al., 2006) as benchmark methods. We implemented tailored adaptations of these algorithms to better suit the specific needs of our problem. To ensure the credibility of our comparisons, we extensively optimized the parameters of each benchmark algorithm prior to conducting the experiments, thereby maximizing their performance. Additionally, we employed ILOG CPLEX version 12.6 to validate the effectiveness of our proposed algorithm.

The parameters of HLMP include the sub-problem period (6 h), initial temperature T_0 (1000), anneal coefficient ΔT(0.01), random probability ε(0.2), learning rate α(0.1), and discount factor γ(0.9). The length of the tabu list $|T|$ is set to half the number of arcs activated in the initial solution. The maximum number of consecutive unimproved iterations allowed for the outer loop is 100, while the inner loop is permitted a maximum of 500 iterations. The parameter settings for TS and SA in the comparison algorithms are consistent with those in RL-TSA. In ALNS, the parameters include an initial score of 2000, an initial temperature of 1000, and an

annealing coefficient of 0.9999, with score increments in ALNS being related to changes in income value. The parameters for GA are as follows: population size of 20, crossover probability of 0.9, and mutation probability of 0.1. In ACO, the number of ants is set to 20, with the relative importance of pheromone set to 1 and that of the heuristic factor set to 2; the pheromone evaporation ratio is 0.8.

The maximum runtime for each algorithm is limited to 60 min. This time constraint is essential, as the algorithm must identify a solution promptly in practical scenarios. To minimize the impact of randomness, each search algorithm is executed 10 times across all instances. The optimization performance of our algorithm is assessed through three measures: best profit (referred to as Max), average profit (designated as Avg), and CPU time (measured in seconds).

4.2 Experimental results

4.2.1 Evaluation of scheduling performance

All 10 instances are used to evaluate the scheduling performance of the algorithms, with the results of the three-day SGNOP scheduling presented in Table 4. The HLMP algorithm exhibited superior performance in all three metrics across all ten instances, indicating that, under the same time constraints, HLMP outperforms other commonly used algorithms for tackling the large-scale SGNOP problem.

In the previous eight examples, the proposed HLMP outperformed its competitors by 1.25% to 36.73% in terms of average results. The SA algorithm also demonstrated notable improvements over the other four comparison algorithms; however, it still lagged behind HLMP. The average results and CPU time for the GA and ACO algorithms across all instances displayed significant disparities when compared to the other algorithms. For Instances 9 and 10, all comparative algorithms failed to generate a solution within the specified time frame. This limitation arises because it is necessary to precompute the conflicting arcs for each visible arc prior to generating the initial solution, which facilitates rapid iterations in the subsequent algorithms. Due to the excessively high number of visible arcs in Instances 9 and 10, the comparative algorithms could not complete the calculation of conflicting arcs within the allocated time.

Furthermore, we divided the 3-day period into 12 sub-problems, each including 6 h. Our algorithm can report the frequency with which LP and LM prevail in competitions conducted within each sub-problem. The profits for all sub-threads in Instances 1−4 were computed using LP. However, as the scale of SGNOP increases, the complexity of the problem presents significant challenges for LP. In Instance 5, CPLEX was unable to solve all sub-threads within the designated time. Detailed competition results for Instance 5 are summarized in Table 5. While CPLEX can determine the theoretical optimal value, it does not outperform LM. Table 6 presents two different outcomes for Instance 5.

The results from the LP and LM analysis indicate that all sub-threads of the algorithm await the LP solution until the maximum time limit is reached, at which point the final output is recorded. When comparing this result with the intermediate output derived from the LM results across all sub-threads, we observe that while the computation time increases by more than 8-fold, the profit only improves by approximately 1%. Nonetheless, this approach remains more competitive than other comparative algorithms.

As the problem scale expands, the LM algorithm proves increasingly suitable for addressing large-scale SGNOP issues. Therefore, we focus more on the solution time and benefits of the LM algorithm in the subsequent Instances 6−10. In practical engineering applications, this mechanism may help identify the scale threshold at which the LM algorithm becomes more advantageous for solving large-scale challenges.

Additionally, we utilize a line chart for a more intuitive comparison of the CPU time for each algorithm across all examples; the results are illustrated in Fig. 5. The overall trend indicates that HLMP achieves higher profits more rapidly than other comparative algorithms in larger-scale instances. When the size of the satellite and the ground station increases to 500 and 50, respectively, HLMP begins to exhibit two different forms of solutions. The black line in the figure, labeled HLMP', represents the situation where, after applying LM to solve all sub-threads and output results, HLMP continues to execute all sub-threads while waiting for the LP results until the maximum time limit is reached.

As the instance scale continues to grow, the CPU time for the TS algorithm is lower than that of other comparative algorithms. However, the profit achieved is relatively lower, suggesting that TS is prone to becoming trapped in a local optimum within a shorter timeframe. It is evident that the GA and ACO algorithms require significant computation time, which is closely related to the complex division of labor within the population and the interaction of ants' search information. This indicates that such algorithms are not well-suited for solving large-scale combinatorial optimization problems.

These experimental results demonstrate that the proposed algorithm converges more rapidly than the comparative algorithms while maintaining strong optimization performance. It is well-equipped for task scheduling, considering that the environment of practical problems is often highly complex and involves large-scale scenarios.

Furthermore, we utilize the results of CPLEX computations on smaller scales to evaluate the effective-

Table 4 Scheduling results of 10 instances

Instance	HLMP				TS		
	Max	Avg	CPU time (s)	Winner	Max	Avg	CPU time (s)
1	**1,571,301**	**1,571,301**	**5**	LP	1,463,741	1,460,204	43
2	**3,327,721**	**3,327,721**	**32**	LP	3,003,512	2,998,520	117
3	**5,074,221**	**5,074,221**	**118**	LP	4,622,503	4,619,061	363
4	**6,855,221**	**6,855,221**	**305**	LP	6,245,560	6,235,465	1,083
5	**8,431,670**	**8,422,546**	**378**	LM	8,024,874	8,014,799	1,562
6	**10,127,884**	**10,118,078**	**663**	LM	9,515,024	9,505,990	1,464
7	**11,862,972**	**11,853,473**	**931**	LM	11,156,980	11,139,919	1,923
8	**13,615,608**	**13,603,610**	**1,114**	LM	12,808,828	12,790,346	3,600
9	**15,333,038**	**15,323,615**	**1,506**	LM	No feasible solution		
10	**17,064,724**	**17,054,095**	**2,045**	LM			

Instance	SA			ALNS		
	Max	Avg	CPU time (s)	Max	Avg	CPU time (s)
1	1,525,899	1,524,398	15	1,495,143	1,465,851	7
2	3,224,569	3,218,676	74	3,154,169	3,086,414	103
3	4,902,538	4,898,454	253	4,814,096	4,714,521	402
4	6,617,387	6,608,945	564	6,422,270	6,348,925	433
5	8,324,883	8,318,800	1,012	8,201,290	8,168,744	1,466
6	10,002,943	9,987,627	1,915	9,826,106	9,753,116	2,586
7	11,707,846	11,697,799	3,131	11,528,633	11,450,532	3,502
8	13,412,978	13,381,530	3,600	13,118,329	12,937,664	3,600
9	No feasible solution					
10						

Instance	GA			ACO		
	Max	Avg	CPU time (s)	Max	Avg	CPU time (s)
1	1,438,415	1,431,299	143	1,373,798	1,349,531	1,932
2	2,952,913	2,943,214	2,846	2,468,704	2,460,221	2,541
3	4,487,128	4,479,093	3,600	3,719,577	3,711,116	3600
4	5,981,141	5,973,045	3,600	5,037,608	5,026,465	3600
5	7,607,728	7,593,474	3,600	6,293,758	6,287,244	3600
6	8,886,497	8,874,106	3,600	7,596,905	7,580,147	3600
7	10,410,384	10,395,347	3,600	8,919,284	8,908,844	3600
8	11,976,964	11,970,112	3,600	10,225,257	10,212,980	3600
9	No feasible solution					
10						

Table 5 The number of victories of sub-thread competition between LP and LM in Instance 5

	1	2	3	4	5	7	8	9	10
LP	6	5	6	6	5	6	5	6	5
LM	6	7	6	6	7	6	7	6	7

Table 6 Comparison of two different results of Instance 5

	Max	Avg	CPU time (s)
LM	8,431,670	8,422,546	378
LP&LM	8,526,046	8,518,394	3,600

ness of our algorithm. As shown in Table 7, our algorithm achieves a profit close to that of CPLEX on Instance 1, with a gap of no more than 0.85%. However, CPLEX took over one hour to solve, which is impractical for real-world applications. This further demonstrates that

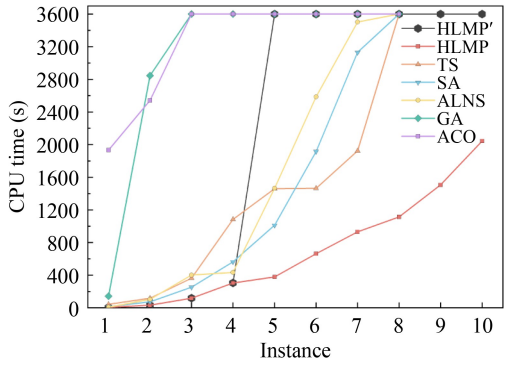

Fig. 5 Optimization time of six algorithms in different instances.

Table 7 The performance comparison between CPLEX and HLMP in Instance 1

	Avg	CPU time (s)
HLMP	1,571,301	5
CPLEX	1,584,851	3,624

the large-scale SGNOP is challenging to solve using a standard solver.

4.2.2 Algorithm stability analysis

Large-scale instances provide a clearer indication of algorithm stability. We selected Instances 1–8 for the stability analysis. The box plots are presented in Fig. 6. It is evident that the HLMP algorithm consistently demonstrates strong average performance across multiple runs, exhibiting minimal volatility. Conversely, the ALNS algorithm performed the worst, showing significant deviation in its results. The stability of the TS and SA algorithms is superior compared to the other three algorithms under review

4.3 Analysis of algorithm strategies

4.3.1 Problem parallelization

To assess the effectiveness of the problem parallelization strategy, we evaluated the benefits and time variations brought about by this strategy based on Instances 1–10. The results are depicted in Fig. 7, where the bar chart represents profit and the line chart illustrates CPU time. It is clear that the problem parallelization strategy signifi-

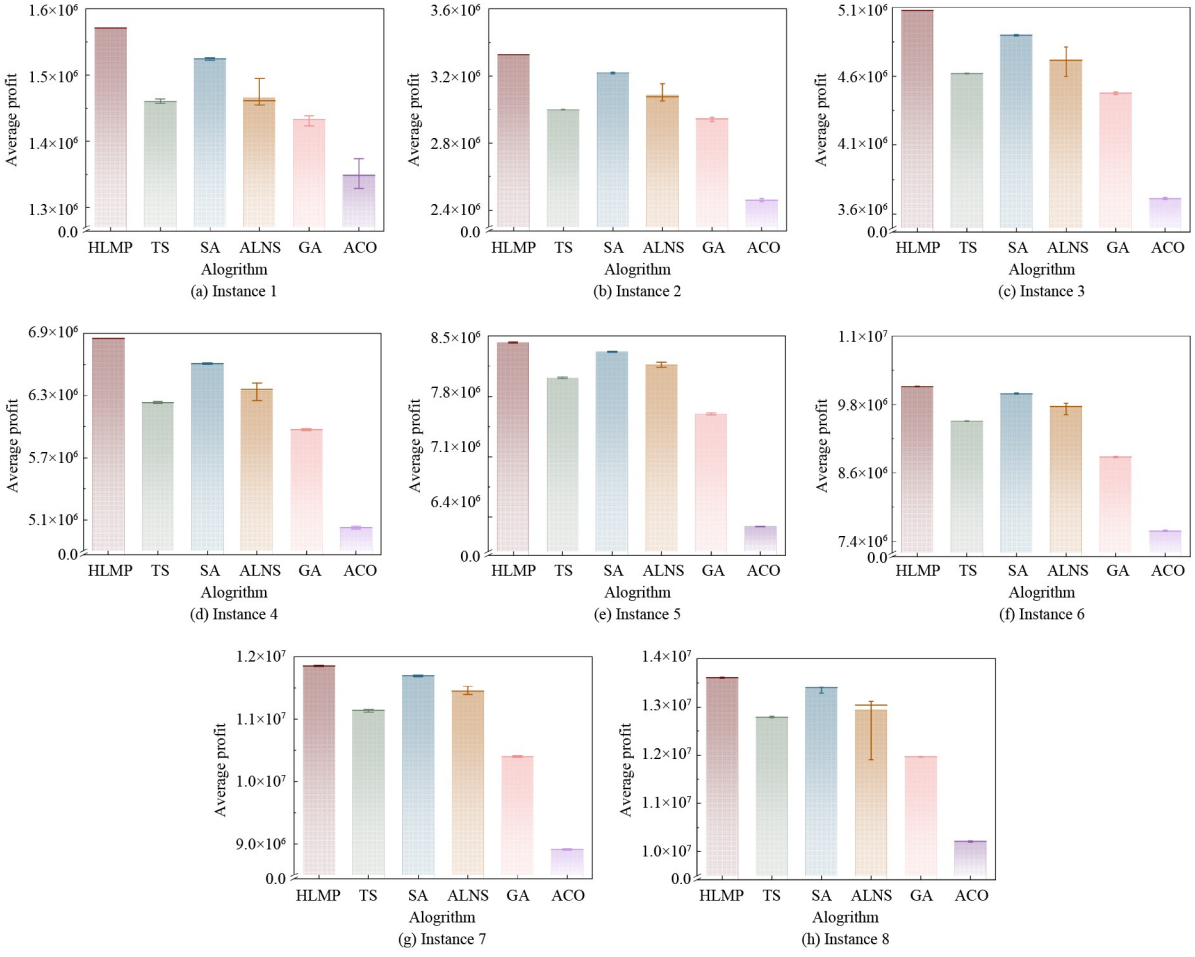

Fig. 6 Box plots of Instances 1–8.

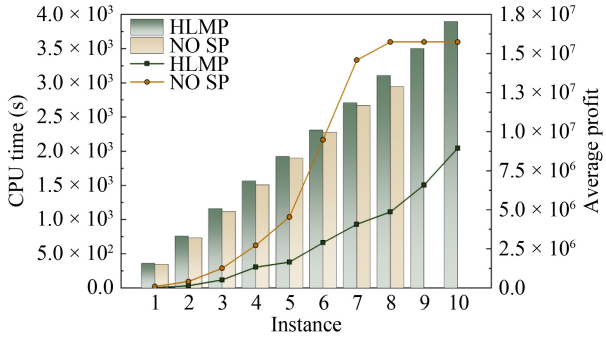

Fig. 7 Effectiveness analysis of the problem parallelization strategy.

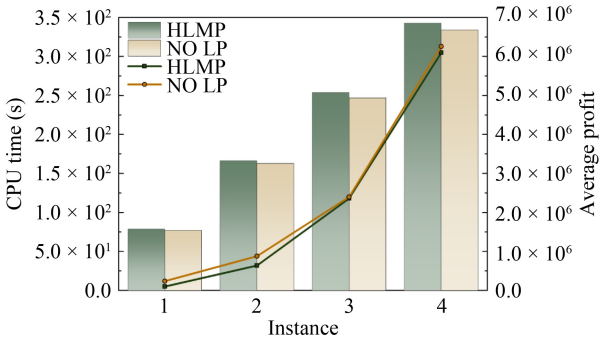

Fig. 8 Effectiveness analysis of LP/CPLEX in parallel competition.

cantly improves CPU time as the scale increases, resulting in reductions of 51.10% to 77.58%, along with profit increases ranging from 1.45% to 5.56%. Furthermore, in the absence of this strategy, solving ultra-large-scale problems such as Instances 9 and 10 would be impossible.

In the initial phase of our algorithm, problem parallelization incorporates a parameter—the duration of each sub-problem. We investigated the feasibility of various duration settings and their impact on the algorithm's performance, as presented in Table 8. We determined that six hours is the optimal duration for the sub-problem, as selecting a period longer than this would hinder problem solving as the scale increases. Conversely, choosing a shorter duration could result in a loss of potential profits.

4.3.2 LP participates in parallel competition

This section contrasts the performance of HLMP with HLMP without LP. We calculated the differences in profit and CPU time attributed to this strategy based on Instances 1, 2, 3, and 4, as CPLEX performs better with small-scale examples.

It can be observed from Fig. 8 that for Instances 1–4, all decomposed sub-problems are solved using LP, which demonstrates a significant advantage in addressing small-scale problems. The integration of LP has resulted in profit increases of 1.98%, 2.16%, 2.72%, and 2.65%, respectively, while also reducing CPU time by 59.35%, 27.44%, 2.24%, and 2.56%, respectively. It is clear that the differences in profit are minimal. This is attributed to the prior enhancement of the LM algorithm's performance through the results of LP, enabling LM to sustain effective solving capabilities. However, as the problem scale increases, the improvement in CPU time achieved through CPLEX diminishes, consistent with the observation that CPLEX becomes less suitable for solving large-scale problems.

4.3.3 Reinforcement learning

To assess the validity of the RL strategy, we evaluated the differences in profit and CPU time based on Instances 6–10. As illustrated in Fig. 9, the integration of RL into the HLMP algorithm considerably optimizes CPU time. When employing RL, the HLMP algorithm converges with only approximately 50% to 71% of the CPU time compared to its version without RL. This indicates that the application of RL to large-scale problems enhances solution efficiency. However, the data also suggests that RL does not offer significant advantages in terms of profit improvement.

5 Discussion

This section discusses the key features and practical applications of the proposed HLMP algorithm in light of the experimental results presented above. First, HLMP can be effectively utilized in real-world satellite scheduling scenarios. Specifically, in the context of satellite Internet engineering applications, HLMP aims to establish a

Table 8 Scheduling results in different durations of the sub-problem

Instance	12H		8H		6H	
	Avg	CPU time (s)	Avg	CPU time (s)	Avg	CPU time (s)
1	**1,578,754**	12	1,574,626	12	1,571,301	**5**
2	**3,346,586**	228	3,336,636	69	3,327,721	**32**
3	**5,106,997**	158	5,088,973	135	5,074,221	**118**
4	**6,899,625**	537	6,882,057	291	6,855,221	**279**
5	No feasible solution		No feasible solution		**8,422,546**	**378**

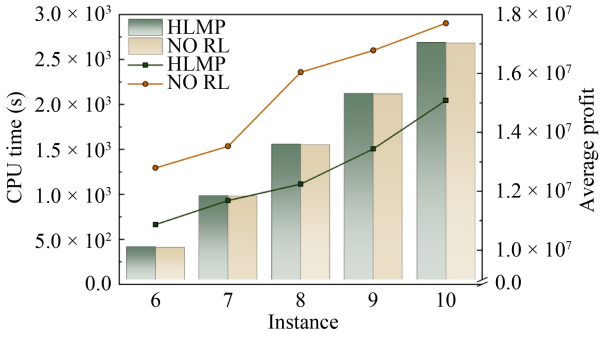

Fig. 9 Effectiveness analysis of reinforcement learning.

stable, efficient, and sustainable communication network between large-scale satellite constellations (such as Internet communication satellites) and ground stations, thereby providing foundational support for Earth observation satellite scheduling and satellite range scheduling. Additionally, HLMP is designed with a multi-parallel solving and deconflicting framework to accommodate the ever-expanding scale of satellite Internet and its extended scheduling periods. This not only satisfies user demands for solution speed, but also caters for their interest in optimizing the quality of multiple solutions.

The second highlight of the HLMP is its significant applicability to the large-scale SGNOP scheduling problem addressed in this paper. Comparative experiments clearly demonstrate that HLMP delivers rapid results while maintaining strong competitiveness, outperforming other comparison algorithms in terms of profit, stability, and convergence speed. The algorithm's exceptional performance on ultra-large-scale instances, in contrast to the inability of comparison algorithms to tackle such instances, further highlights the HLMP's capability to address problems across a range of scales.

Additionally, the proposed HLMP is accessible and straightforward to implement, as it does not involve complicated algorithms or strategies. Emphasizing problem parallelization and algorithm parallelization proves essential for enhancing the efficiency and quality of the solutions. However, a limitation of the algorithm is its inability to adaptively adjust the period of the sub-problems according to the scale of the problem. Furthermore, the design of the state space, based on fitness and its enhancements, ensures that the reinforcement learning model remains highly generalizable and adaptable to various problem scenarios.

6 Conclusions

The paper examined the satellite-ground networking optimization problem to maximize the link establishment time for satellite-ground networking in an integer LP model with a LP and learning-based metaheurics hybrid learning-assisted multi-parallel algorithm. To work with satellite-ground networking instances at different scales up to 1000 satellites and million-level time window scales, the developed algorithm was compared with five metaheuristics and the mathematical programming solver CPLEX in a fairly extensive set of experiments. The results of the comparison and discussion indicate that the proposed model and algorithm provide a more efficient solution for this problem and can stably and efficiently be used in cases of any scale to fulfill the competing requirements for an accurate solution for small-scale situations and a fast and effective solution for fairly large-scale situations. In the future, we can study how to adaptively adjust the period of sub-problems depending on the problem's scale and consider the inter-satellite link. At the same time, we can also think about how to extend this research to other fields.

Competing Interests The authors declare that they have no competing interests.

References

Barbulescu L, Watson J P, Whitley L D, Howe A E (2004). Scheduling space–ground communications for the air force satellite control network. Journal of Scheduling, 7(1): 7–34

Chen M, Du Y, Tang K, Xing L, Chen Y, Chen Y (2024). Learning to construct a solution for the agile satellite scheduling problem with time-dependent transition times. IEEE Transactions on Systems, Man, and Cybernetics. Systems, 54(10): 5949–5963

del Portillo I, Cameron B G, Crawley E F (2019). A technical comparison of three low earth orbit satellite constellation systems to provide global broadband. Acta Astronautica, 159: 123–135

Doltsinis S, Ferreira P, Lohse N (2014). An MDP model-based reinforcement learning approach for production station ramp-up optimization: Q-learning analysis. IEEE Transactions on Systems, Man, and Cybernetics. Systems, 44(9): 1125–1138

Dorigo M, Birattari M, Stutzle T (2006). Ant colony optimization. IEEE Computational Intelligence Magazine, 1(4): 28–39

Du Y, Wang L, Xing L, Yan J, Cai M (2021). Data-driven heuristic assisted memetic algorithm for efficient inter-satellite link scheduling in the Beidou navigation satellite system. IEEE/CAA Journal of Automatica Sinica, 8(11): 1800–1816.

Du Y, Wang T, Xin B, Wang L, Chen Y, Xing L (2020). A data-driven parallel scheduling approach for multiple agile earth observation satellites. IEEE Transactions on Evolutionary Computation, 24(4): 679–693

Du Y, Xing L, Chen Y (2022). Satellite scheduling engine: The intelligent solver for future multi-satellite management. Frontiers of Engineering Management, 9(4): 683–688

Du Y, Xing L, Zhang J, Chen Y, He Y (2019). MOEA based memetic algorithms for multi-objective satellite range scheduling problem. Swarm and Evolutionary Computation, 50: 100576

Glover F (1986). Future paths for integer programming and links to

artificial intelligence. Computers & Operations Research, 13(5): 533–549

He H, Zhou D, Sheng M, Li J (2023). Hierarchical cross-domain satellite resource management: An intelligent collaboration perspective. IEEE Transactions on Communications, 71(4): 2201–2215

Holland J H (1992). Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT press

Hooker J N (2015). Toward unification of exact and heuristic optimization methods. International Transactions in Operational Research, 22(1): 19–48

Kirkpatrick S, Gelatt Jr C D, Vecchi M P (1983). Optimization by simulated annealing. Science, 220(4598): 671–680

Kodheli O, Lagunas E, Maturo N, Sharma S K, Shankar B, Montoya J F M, Duncan J C M, Spano D, Chatzinotas S, Kisseleff S, Querol J, Lei L, Vu T X, Goussetis G (2021). Satellite communications in the new space era: A survey and future challenges. IEEE Communications Surveys and Tutorials, 23(1): 70–109

Li R, Gong W, Lu C, Wang L (2023). A learning-based memetic algorithm for energy-efficient flexible job-shop scheduling with type-2 fuzzy processing time. IEEE Transactions on Evolutionary Computation, 27(3): 610–620

Li S, Gong W, Wang L, Gu Q (2024). Evolutionary multitasking via reinforcement learning. IEEE Transactions on Emerging Topics in Computational Intelligence, 8(1): 762–775

Lin X, Chen Y, Xue J, Zhang B, Chen Y, Chen C (2024a). Parallel machine scheduling with job family, release time, and mold availability constraints: model and two solution approaches. Memetic Computing, 16(3): 355–371

Lin X, Chen Y, Xue J, Zhang B, He L, Chen Y (2024b). Large-volume LEO satellite imaging data networked transmission scheduling problem: Model and algorithm. Expert Systems with Applications, 249: 123649

Liu X, Laporte G, Chen Y, He R (2017). An adaptive large neighborhood search metaheuristic for agile satellite scheduling with time-dependent transition time. Computers & Operations Research, 86: 41–53

Liu Y, Zhang S, Hu H (2022). A simulated annealing algorithm with tabu list for the multi-satellite downlink schedule problem considering waiting time. Aerospace, 9(5): 235

Liu Z, Feng Z, Ren Z (2019). Route-reduction-based dynamic programming for large-scale satellite range scheduling problem. Engineering Optimization, 51(11): 1944–1964

Luo K, Wang H, Li Y, Li Q (2017). High-performance technique for satellite range scheduling. Computers & Operations Research, 85: 12–21

Marinelli F, Nocella S, Rossi F, Smriglio S (2011). A Lagrangian heuristic for satellite range scheduling with resource constraints. Computers & Operations Research, 38(11): 1572–1583

Moon J (2021). Generalized risk-sensitive optimal control and Hamilton–Jacobi–Bellman equation. IEEE Transactions on Automatic Control, 66(5): 2319–2325

Puchinger J, Raidl G R (2005). Combining metaheuristics and exact algorithms in combinatorial optimization: A survey and classification. In: International Work-Conference on the Interplay between Natural and Artificial Computation. Berlin, Heidelberg: Springer Berlin Heidelberg, 41–53

Ren L, Ning X, Wang Z (2022). A competitive Markov decision process model and a recursive reinforcement-learning algorithm for fairness scheduling of agile satellites. Computers & Industrial Engineering, 169: 108242

Shiue Y R, Lee K C, Su C T (2018). Real-time scheduling for a smart factory using a reinforcement learning approach. Computers & Industrial Engineering, 125: 604–614

Song Y, Ou J, Wu J, Wu Y, Xing L, Chen Y (2023a). A cluster-based genetic optimization method for satellite range scheduling system. Swarm and Evolutionary Computation, 79: 101316

Song Y, Wei L, Yang Q, Wu J, Xing L, Chen Y (2023b). RL-GA: A reinforcement learning-based genetic algorithm for electromagnetic detection satellite scheduling problem. Swarm and Evolutionary Computation, 77: 101236

Song Y J, Zhang Z S, Song B Y, Chen Y W (2019). Improved genetic algorithm with local search for satellite range scheduling system and its application in environmental monitoring. Sustainable Computing: Informatics and Systems, 21: 19–27

Su J, Fu Y, Gao K, Dong H, Mou J (2023). Integrated scheduling problems of open shop and vehicle routing using an ensemble of group teaching optimization and simulated annealing. Swarm and Evolutionary Computation, 83: 101373

Wang J, Song G, Liang Z, Demeulemeester E, Hu X, Liu J (2023). Unrelated parallel machine scheduling with multiple time windows: An application to earth observation satellite scheduling. Computers & Operations Research, 149: 106010

Wang L, Li K, Ma Q, Lu Y (2020). Hybrid dynamic learning mechanism for multivariate time series segmentation. Statistical Analysis and Data Mining, 13(2): 165–177

Wang L, Lu J (2019). A memetic algorithm with competition for the capacitated green vehicle routing problem. IEEE/CAA Journal of Automatica Sinica, 6(2): 516–526

Wang L, Pan Z, Wang J (2021b). A review of reinforcement learning based intelligent optimization for manufacturing scheduling. Complex System Modeling and Simulation, 1(4): 257–270

Wang L, Wang J, Jiang E (2021a). Decomposition based multiobjective evolutionary algorithm with adaptive resource allocation for energy-aware welding shop scheduling problem. Computers & Industrial Engineering, 162: 107778

Wu J, Song B, Zhang G, Ou J, Chen Y, Yao F, He L, Xing L (2022). A data-driven improved genetic algorithm for agile earth observation satellite scheduling with time-dependent transition time. Computers & Industrial Engineering, 174: 108823

Wu J, Yao F, Song Y, He L, Lu F, Du Y, Yan J, Chen Y, Xing L, Ou J (2023). Frequent pattern-based parallel search approach for time-dependent agile earth observation satellite scheduling. Information Sciences, 636: 118924

Yao F, Du Y, Li L, Xing L, Chen Y (2023). General modeling and optimization technique for real-world earth observation satellite scheduling. Frontiers of Engineering Management, 10(4): 695–709

Yu Y, Sun W, Tang J, Wang J (2017). Line-hybrid seru system conversion: Models, complexities, properties, solutions and insights.

Computers & Industrial Engineering, 103: 282–299

Zhao F, Jiang T, Wang L (2023). A reinforcement learning driven cooperative meta-heuristic algorithm for energy-efficient distributed no-wait flow-shop scheduling with sequence-dependent setup time. IEEE Transactions on Industrial Informatics, 19(7): 8427–8440

Zhou L, Liang Z, Chou C A, Chaovalitwongse W A (2020). Airline planning and scheduling: Models and solution methodologies. Frontiers of Engineering Management, 7(1): 1–26

Zhu W, Ao Z, Baldacci R, Qin H, Zhang Z (2023). Enhanced solution representations for vehicle routing problems with split deliveries. Frontiers of Engineering Management, 10(3): 483–498