Swarm and Evolutionary Computation 79 (2023) 101316

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

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

A cluster-based genetic optimization method for satellite range scheduling system

Yanjie Song [a,1,*], Junwei Ou [a,1], Jian Wu [a,*], Yutong Wu [b], Lining Xing [c], Yingwu Chen [a]

[a] *College of Systems Engineering, National University of Defense Technology, Changsha, Hunan, 410073, China*
[b] *Newcastle University Business School, NE1 4SE, UK*
[c] *School of Electronic Engineering, Xidian University, Xi'an, 710126, China*

ARTICLE INFO

Keywords:
Satellite range scheduling
Clustering
Meta-heuristic algorithm
Genetic algorithm
MIP
Task arrangement algorithm
Machine learning

ABSTRACT

With the rapid development of the satellite industry, how to effectively manage satellites has become an essential issue for ground operation management. By using the k-means clustering method, a cluster-based genetic algorithm (C-BGA) is proposed for the satellite ranging scheduling problem (SRSP). In the C-BGA, a heuristic-based population initialization strategy and a cluster-based evolution strategy are designed for searching for an ideal solution. Four heuristic rules were used in the initial population generation process. Population evolution process is accomplished by cluster-based crossover and mutation. These strategies also improve the algorithm's adaptability to cope with different scenarios. To increase the possibility of the task being successfully scheduled, a task arrangement algorithm (TAA) is used to generate task execution plans. Experiments are carried out to prove that the proposed algorithm can effectively solve the SRSP problem.

1. Introduction

With the rapid development of space technology in recent years, the satellite is increasingly indispensable and has been implemented in many fields [1]. Many of the satellites are used to complete multiple tasks such as observation, navigation, and communication [2]. Regardless of the type of tasks need to execute, satellite ground stations deliver instructions through communication links to satellites directly or indirectly (transmit to the relay satellite first). This process is called satellite TT&C (telemetry, track, and command), which uploads action instructions to satellites and obtains satellites' operating status. The satellite range scheduling system carries out effective management of satellite and ground station resources by using high-efficiency scheduling algorithms [3]. Therefore, effective task scheduling for the satellite range scheduling problem (SRSP) is vital when numerous satellites need to be managed.

Satellite range scheduling describes the process of arranging TT&C tasks when satellites and satellite ground stations are visible to each other. The mutually visible time range is called the visible time window (VTW). In other words, satellite range scheduling is simply the process of determining the optimal combination of TT&C tasks for a series of satellites and satellite ground stations within VTWs [4]. Our optimization goal is to find a reasonable satellite-ground station time window scheme that maximizes the total profit.

The primary factors that hinder the SRSP problem are sequence dependence and over-subscription [5]. Firstly, sequence dependency, means the scheduling result of one task will affect other subsequent tasks, making the plan lack regularity. Additionally, over-subscription, means that even if the entire VTWs are used, a part of tasks still cannot be successfully executed. The coexistence of these two characteristics makes it almost impossible to find the optimal solution. Besides, the complexity of SRSP has been proven to be NP-hard [6].

Evolutionary algorithms such as genetic algorithm, ant colony algorithm, and particle swarm algorithm [7–9] have been widely used in the satellite scheduling problem. Among them, the genetic algorithm, as a classic evolutionary algorithm, has successfully solved many task scheduling problems in practical applications [10]. Some accurate algorithms have also successfully solved some satellite scheduling problems. However, due to the difficulty of solving the problem, accurate algorithms can only find optimal solutions for small-scale problems, and the solution quality of large-scale problems is not ideal and stable [11]. Compared with accurate solving algorithms, the genetic algorithm is more effective to solve scheduling problems in large-scale scenarios, and it is not easy to occur exponential explosion [12].

* Corresponding author at: College of Systems Engineering, National University of Defense Technology, Changsha, Hunan, 410073, China.
E-mail addresses: songyj_2017@163.com (Y. Song), junweiou@163.com (J. Ou), jianwwwu@126.com (J. Wu), wuyutong119@gmail.com (Y. Wu), xinglining@gmail.com (L. Xing), ywchen@nudt.edu.cn (Y. Chen).
[1] These authors contributed equally to this article.

https://doi.org/10.1016/j.swevo.2023.101316
Received 16 February 2022; Received in revised form 3 March 2023; Accepted 13 April 2023
Available online 19 April 2023

Some traditional evolutionary algorithms also tried to improve algorithm performance from the perspective of data features, but these are often based on assumptions or subjective inferences. These methods are also effective for some cases with low data dimensions. As a method that uses data features to complete classification or prediction, machine learning methods are capable of learning data and processing high-dimensional data. Therefore, combining machine learning methods with evolutionary computation is indeed an effective strategy for solving combinatorial optimization problems. In this paper, a cluster-based genetic algorithm (C-BGA) will be proposed to solve the SRSP problem. Moreover, the K-means clustering method and genetic algorithm are combined to improve search efficiency. Classification results obtained by the K-means clustering method will assist the crossover and mutation operations. This process allows information obtained through machine learning to feedback on population evolution, and the information transfer mechanism promotes algorithms to find a better solution. It is meaningful to drive the algorithm optimization process according to task characteristics.

The main contributions of this study are:

(1) A clustering classification strategy based on the characteristics of SRSP is proposed. The clustering method divides task data into multiple categories according to the tasks' characteristics. The classification results will guide the optimization process of the algorithm search.

(2) A cluster-based genetic algorithm is proposed to solve the SRSP problem. The C-BGA combines a K-means clustering method and a genetic algorithm. Heuristic population initialization strategy, cluster-based crossover, cluster-based mutation, and other improved strategies are used in the proposed algorithm.

(3) A task arrangement algorithm is proposed to generate plans. This algorithm can quickly identify whether a task can be scheduled within the visible time window. Tasks can be arranged in a heuristic way, and time window resources will be updated accordingly.

The structure of this study is as follows. The second section introduces the related work that has been carried out. The third section introduces the SRSP mixed integer model. The fourth section introduces the clustering-based genetic algorithm and a task arrangement algorithm. The fifth part will design several experiments and compare them with other competitive algorithms. The last part will introduce research conclusions and future research directions.

2. Related work

Satellite range scheduling problems, as an important category of satellite scheduling problems, along with satellite data download scheduling problems and relay satellite scheduling problems, constitute satellite support scheduling problems. The earliest representative research is related to the AFIT network. Through comparison of multiple algorithms, the genetic algorithm has achieved good performance in solving SRSP problem [13]. Luo (2017) used a combination of relaxation and heuristics to obtain high-performance solutions [14]. Zufferey (2008) combined the graph coloring approach and heuristics to find good solutions by constructing feasible solutions [15]. Marinelli (2011) introduced a time-indexed 0,1-linear programming formulation and developed a Lagrangian version of the fix-and-relax MIP heuristic [16]. Zhang (2014) designed a two-stage ant colony algorithm, and several heuristic strategies are used to guide the search process of ant colony [17]. The solution effect was studied, and the genetic algorithm, iterative repair algorithm, and max–min ant system algorithm were compared. Zhang (2018) tried to adopt a new model-building method to reduce the workload of dispatchers [18]. Optimization of the model makes a simple ant colony algorithm perform better than other novel ACO algorithms. Song (2019) used a multi-objective optimization method to solve the SRSP problem. Ensemble ideas are used in the NSGA-II algorithm [19]. Deep reinforcement learning methods also provide a new way of solving satellite scheduling problems. Ou et al. (2023) proposed a deep reinforcement learning method for task assignment [20]. This method has good performance in small-scale instance scenarios.

Algorithms are the key to solving combinatorial optimization problems. Among the many types of algorithms, the genetic algorithm has been widely used in combinatorial optimization problems and solves satellite scheduling problems well. Parish (1994) tried to use genetic algorithms and simple rules to dispatch satellite ground station systems automatically [21]. Sun (2011) designed a genetic algorithm implemented in Matlab and verified the effectiveness of solving the SRSP problem through random instances [22]. Xhafa (2012) analyzed the improvement of optimization effect when heuristic rules were implemented in genetic algorithm [23]. This algorithm is combined with an STK toolbox to improve the scheduling quality of the satellite range scheduling system. A steady-state genetic algorithm is proposed by Xhafa(2013) [24]. Replacement of some individuals in the population is also considered. This algorithm cannot achieve the same optimization performance for the multi-objective SRSP problem. Kim (2015) used a genetic algorithm to obtain SAR satellite constellation operation plan [25]. System response time can be reduced by this method. Zheng (2017) proposed a genetic algorithm with a dynamic mutation strategy [26]. This algorithm is tested by multi-satellite and multi-task scheduling problem. Results show its outstanding speed and reliability. Song (2018) combined a genetic algorithm and neighborhood search algorithm to find feasible solutions for satellite data download scheduling problem [27]. Berger (2018) proposed a graph-based genetic algorithm, which is a low-cost task scheduling heuristic that was integrated into a genetic algorithm [28]. Barkaoui (2020) combined good rules for solving vehicle route planning with the time windows (VRPTW) problem with genetic algorithms to find plans for satellite constellations [29]. The satellite range scheduling problem can be regarded as a parallel machine scheduling problem, and the genetic algorithm has been well applied in parallel machine scheduling problems [30,31].

Other evolutionary algorithms such as the particle swarm algorithm and firework algorithm were also used to solve the satellite scheduling problem. Khojah et al. (2022) used a particle swarm algorithm to solve the observation satellite scheduling problem and obtained high-quality solutions by task prioritization [32]. The proposed algorithm is mainly for problems with multiple optimization objectives, and the fast convergence of the algorithm becomes critical when it is only a single-objective optimization problem. Song et al. (2021) proposed a dynamic population of fireworks algorithm to generate execution plans for relay satellites [33]. The proposed algorithm is effective for solving specific problem scenarios.

Clustering methods are mainly divided into partitional clustering and hierarchical clustering, based on the properties of the generated clusters [34,35]. Using clustering methods or other intelligent methods to solve planning problems is showing a trend of development in recent years. Several researchers have tried to use clustering methods with heuristic algorithms, memetic algorithms, and hyper-heuristic algorithms to solve several types of vehicle routing problems [36–38]. In these algorithms, the clustering method measures the proximity relationship between transportation destinations, which can quickly classify transportation tasks to improve the quality of the solution.

To our best knowledge, the combination of the clustering method and the evolutionary algorithm has not been used to solve the satellite scheduling problem. Compared with using evolutionary algorithms alone, clustering methods can effectively use data characteristics required for task scheduling and guide the optimization process. The clustering method and genetic algorithm are combined in this study. Some other strategies are adopted in the genetic algorithm to improve search performance.

3. Scheduling model

In this section, a description of the problem will be given and a task scheduling model for the SRSP problem will be proposed. Constraints about tasks and resources are considered in the model.

Table 1
Symbols and variables.

Notation	Description
T	Task set, $T = \{task_1, task_2, task_3, \ldots, task_{\|T\|}\}$
Ant	Antenna set, $Ant = \{ant_1, ant_2, ant_3, \ldots, ant_{\|A\|}\}$
A_i	Available antenna set of task i
TW_{ij}	Visible time window set of task i on antenna j
tw_{ij}^k	kth visible time window of task i on antenna j
$\left[tws_{ij}^k, twe_{ij}^k\right]$	Start time and end time of visible time window
$\left[est_i, let_i\right]$	Earliest allowable start time and latest allowable end time of task i
$\left[st_{ij}^k, et_{ij}^k\right]$	Actual start time and end time of task i
p_i	Profit of task i
γ	Conversion time between two tasks
d_i	Required duration of task i
act_{ij}^k	Actual duration of task i
M	A big integer
x_{ij}^k	If task i is scheduled in the kth time window on the antenna j, $x_{ij}^k = 1$; otherwise, $x_{ij}^k = 0$

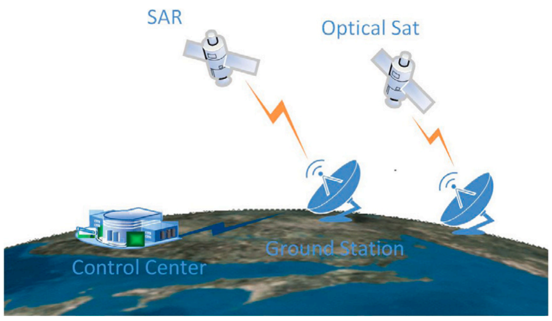


Fig. 1. Scenario diagram.

3.1. Symbols and variables

Variables and symbols involved in the model are given in Table 1.

3.2. Problem description

When man-made earth satellites orbit the earth, the ground satellite control center needs to monitor the state of satellites and give commands to satellites [39]. The scene of satellite control is shown in Fig. 1. Satellite-ground station links are established between the two satellites and the ground stations, while the other one cannot execute TT&C tasks due to the lack of resources.

In the SRSP problem, there are multiple satellites, ground stations, and tasks that need to be scheduled. A task is defined by multiple attributes $\{est_t, let_t, p_t, d_t\}$. Time-related characteristics of tasks are shown in Fig. 2. These attributes limit time intervals within which tasks can be executed [40]. A task needs to be executed in a VTW of the ground station while satisfying its time attribute constraints. For example, there are four TT&C tasks and four antennas. For task No. 1, antenna No. 1 and antenna No. 2 have VTWs. For task No. 2 and task No. 3, only antenna No. 4 and antenna No. 2 have VTW, respectively. For task No. 4, all antennas have VTWs. The planning results are shown in Fig. 3, where tasks No. 1–4 are assigned to antennas No. 1, No. 4, No. 2, and No. 3, respectively. Tasks are executed within the time range and VTWs required by the tasks.

It is obvious that obtaining a suitable solution requires a reasonable mathematical model and an efficient solution algorithm, thus a mixed-integer programming model is proposed.

3.3. Mathematical model of SRSP

The mathematical model is based on the following assumptions.
Assumptions:

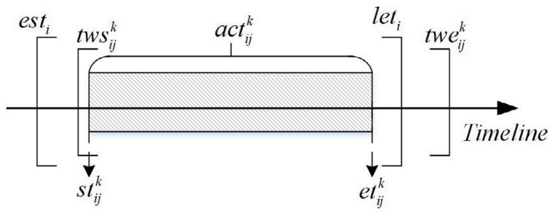


Fig. 2. Time-related characteristics of task.

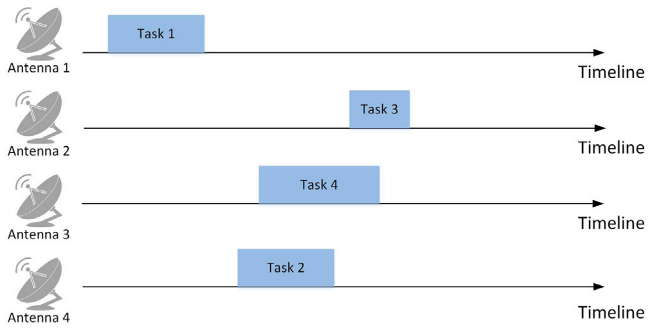


Fig. 3. Scheduling results for four tasks and four antennas.

1. An antenna can only serve one satellite at any moment;
2. A task can only be executed once at most;
3. No interruption will occur during the execution of tasks;
4. Task attributes are known in advance, and no temporary tasks will appear;
5. Failure of execution due to electromagnetic, geographic, and other factors are not considered;
6. Energy on satellites is sufficient to ensure the completion of tasks;

The aim of scheduling is to select the appropriate antenna and start time for TT&C tasks. Therefore, a decision variable x_{ij}^k is introduced into the model. When task i is scheduled in kth time window on antenna j, $x_{ij}^k = 1$; otherwise, $x_{ij}^k = 0$. The scheduling goal of the SRSP problem is to obtain the highest profit. The objective function is represented as follows:

Objective function:

$$\max f = \sum_{i \in T} \sum_{j \in A_i} \sum_{k \in TW_{ij}} p_i \cdot x_{ij}^k \tag{1}$$

where p_i denotes profit of task i, x_{ij}^k denotes whether task i is scheduled in the kth time window on the antenna j.

Constraints of the SRSP problem mainly come from two aspects, one is the task requirement constraint, and the other is the resource requirement constraint [41]. The task requirement constraint restricts the execution of the task in terms of the number of tasks and their durations. Resource requirement constraint restricts the time attributes of a task from the perspective of the resource's ability to execute tasks.

Constraints:

$$\left(act_{ij}^k - d_t\right) \times x_{ij}^k = 0, \forall i \in T, j \in A_i, tw_{ij}^k \in TW_{ij} \tag{2}$$

$$st_{ij}^k + act_{ij}^k = et_{ij}^k, \forall i \in T, j \in A_i, tw_{ij}^k \in TW_{ij} \tag{3}$$

$$est_{ij}^k \cdot x_{ij}^k \leq st_{ij}^k, \forall i \in T, j \in A_i \tag{4}$$

$$et_{ij}^k \cdot x_{ij}^k \leq let_{ij}^k, \forall i \in T, j \in A_i \tag{5}$$

$$tws_{ij}^k \cdot x_{ij}^k \leq st_{ij}^k, \forall i \in T, j \in A_i, tw_{ij}^k \in TW_{ij} \tag{6}$$

$$et_{ij}^k \cdot x_{ij}^k \leq twe_{ij}^k, \forall i \in T, j \in A_i, tw_{ij}^k \in TW_{ij} \tag{7}$$

$$\sum_{j \in A_i} x_{ij}^k \leq 1, \forall i \in T, j \in A_i, tw_{ij}^k \in TW_{ij} \tag{8}$$

$$\sum_{k \in TW_{ij}} x_{ij}^k \leq 1, \forall i \in T, j \in A_i \tag{9}$$

$$\sum_{k \in TW_{ij}} \sum_{j \in A_i} x_{ij}^k \leq 1, \forall i \in T \tag{10}$$

$$et_{ij} \cdot x_{ij}^k + \gamma \leq st_{i'j} + M \cdot \left(1 - x_{i'j}^{k'}\right), \forall i, i' \in T, j \in A_i \tag{11}$$

$$x_{ij}^k \in \{0, 1\}, \forall i \in T, j \in A_i, tw_{ij}^k \in TW_{ij} \tag{12}$$

Constraint (2) indicates that the actual duration of a task should be the same as the required time. Constraint (3) indicates the relationship between task start time and end time. Constraint (4) indicates that the task needs to start after the earliest allowable start time of the task. Constraint (5) indicates that the task needs to be completed before the latest allowable time of the task. Constraint (6) indicates that the task needs to start after the start time of the visible time window. Constraint (7) indicates that the task needs to be completed before the end time of the visible time window. Constraint (8) indicates that one task can be executed at most once. Constraint (9) indicates that each task only can be executed by one antenna. Constraint (10) indicates that each task only can be executed in one visible time window. Constraint (11) indicates that two tasks must meet task conversion time interval requirements. Constraint (12) indicates the value range of the decision variable.

4. Solving method

To solve the SRSP problem, the C-BGA is proposed by combining a genetic algorithm and a clustering method. The clustering method guides the crossover and mutation operation of population search. A task arrangement algorithm (TAA) is adopted to generate a task execution plan for individuals. This section will introduce the TAA and C-BGA below.

4.1. Task arrangement algorithm

The TAA determines the specific execution plan of each task according to a series of individuals obtained by the C-BGA. Tasks are inserted into time windows following a priority order. The pseudo-code of the TAA is shown in Algorithm 1.

As shown in Algorithm 1, a task determines whether a time window can execute it in order (Line 2). If the length of a time window exceeds the time required by the task, the algorithm will try to schedule it (Line 5). Based on this, A quick identification method is designed to determine whether the task is to be scheduled. Two new variables are introduced before scheduling, eat_i and lat_i, which denote the earliest actual available time and the latest actual available time, respectively. The calculation method of these two variables is as shown in Eqs. (13) and (14). If the range of actual available time exceeds the required time for a task, it can be successfully scheduled, and the task is scheduled to start at the earliest actual available time (Line 6). After the task is successfully scheduled, available time window resources need to be updated. If the earliest actual available time of the task is equal to the earliest allowable start time of the task, the original time window is clipped to a new time window (Line 9); otherwise, the original time window is clipped to two new time windows (Line 11). After finishing the clipping process, the time window set will be updated (Line 12). The next task will be considered to schedule.

Algorithm 1: Task Arrangement Algorithm (TAA)

```
Input: Task Set $T$, Time Window $TW$, Population $P$
Output: Fitness of population $R$
1  foreach $indi_o$ in $P$ do
2      foreach $task_i$ in the order in $indi_o$ do
3          foreach $tw_{ij}^k$ in $TW$ do
4              $eat_i \leftarrow \max\left\{tws_{ij}^k, est_i\right\}$ ;
5              $lat_i \leftarrow \min\left\{twe_{ij}^k, let_i\right\}$ ;
6              if $(twe_{ij}^k - tws_{ij}^k) \geq d_i$ and $(lat_i - eat_i) \geq d_i$ then
7                  $st_i \leftarrow$ Arrange task start time at $eat_i$;
8                  $et_i \leftarrow$ Arrange task end time at $(st_i + d_i)$;
9                  Omit $tws_{ij}^k$ from TW ;
10                 if $eat_i == tws_{ij}^k$ then
11                     Generate a new time window $tw_{ij}^{k'}$ with the
                       attributes $\left[et_i, twe_{ij}^k\right]$ ;
12                 else
13                     Generate two new time windows $tw_{ij}^{k'}$ and
                       $tw_{ij}^{k''}$ with the attributes $\left[tws_{ij}^k, st_i\right]$ and
                       $\left[et_i, twe_{ij}^k\right]$ ;
14                 $TW \cup \{tw', tw''\} \leftarrow$ Update $TW$ with new time
                   windows ;
15                 Try to arrange the next task $task_{i+1}$;
16             else
17                 Turn to next time window $tw_{ij}^{k+1}$;
18     Save as task execution plan ;
19 $R \leftarrow$Calculate fitness for arranged tasks in $P$;
```

$$eat_i \leftarrow \max\left\{tws_{ij}^k, est_i\right\}, tws \in TW, est \in T \tag{13}$$

$$lat_i \leftarrow \min\left\{twe_{ij}^k, let_i\right\}, twe \in TW, let \in T \tag{14}$$

TAA arranges all tasks T one by one according to time windows TW. Therefore, the time complexity of the task arrangement algorithm is $O(|T| \times |TW|)$, where $|T|$ represents the number of tasks, $|TW|$ denotes the number of time windows.

4.2. Cluster-based genetic algorithm

To make full use of data features in the problem, we incorporate the clustering method that introduces data features into the genetic algorithm framework. The clustering method updates the feature matrix in time according to arrangement results. New categories will generate and guide the subsequent optimization process according to the updated information.

The traditional genetic algorithm has a strong global search ability but performs poorly in local search. Moreover, genetic operators (including crossover, mutation, etc.) in genetic algorithms need to be carefully designed according to the characteristics of the specific problem. This algorithm design method might easily lead to an algorithm that can only perform well for specific problems or specific scenarios. To change this situation, features and the information obtained are used through the optimization process in the algorithm framework. The k-means clustering method gives the algorithm a strong capacity for generalization.

Algorithm 2: Cluster–based Genetic Algorithm

```
   Input: Task Set $T$, Time Window Set $TW$, $Gen$, $Thre_1$, $Thre_2$,
          $Thre_3$, $per$, $n$, $K$
   Output: $Solution$
1  Initialization: $gen = 1$, $count_1 = 0$, $count_2 = 0$. $count_3 = 0$, $l\_best$,
     $g\_best$, $last\_best$ ;
2  $P_0 \leftarrow$ Heuristic Initial Population Generation($T$,$|P|$) ;
3  $C \leftarrow$ K-means Method($T$,$n$,$K$) ;
4  while $gen$ not equals to $Gen$ do
5      $R \leftarrow$ Generate plan by Task Arrangement
         Algorithm($T$,$TW$,$P$) ;
6      if $l\_best > g\_best$ then
7          $g\_best \leftarrow l\_best$ ;
8          $count_1 = count_1 + 1$ ;
9      if $l\_best < last\_best$ then
10         $count_2 = count_2 + 1$ ;
11     else
12         if $l\_best < last\_best * per$ then
13             $count_3 = count_3 + 1$ ;
14     Select individual and category by Roulette Wheel Method ;
15     $P' \leftarrow$ Use Cluster-based Crossover Method($P$) ;
16     $P' \leftarrow$ Use Cluster-based Mutation Method($P$) ;
17     if $count_1$ equals to $Thre_1$ then
18         $K' \leftarrow$ Update $K$ ;
19         $C' \leftarrow$ Update Feature Matrix by K-means
             ($C \cup R$,$n+1$,$K'$) ;
20         Reset $count_1$ to 0 ;
21     if $count_2$ equals to $Thre_2$ then
22         $P' \leftarrow$ Update worst individual in $P$ with $g\_best$ ;
23         Reset $count_2$ to 0 ;
24     if $count_3$ equals to $Thre_3$ then
25         $P' \leftarrow$ Update worst individual by $l\_best$ ;
26         $P' \leftarrow$ Generate a new individual and select an
             individual to replace randomly ;
27         Reset $count_3$ to 0 ;
28     $last\_best \leftarrow$ Update $l\_best$;
29     $P \leftarrow$ Update $P'$ ;
30     $gen = gen + 1$ ;
```

The pseudo-code of the C-BGA is shown in Algorithm 2. As shown in Algorithm 2, a variety of strategies are adopted in the C-BGA. A heuristic population initialization strategy is used to generate the initial population (Line 2). K-means clustering method obtains classification results according to data and the initial value of K (Line 3). TAA is used to schedule tasks and calculate fitness value according to tasks' profit (Line 5). According to the fitness value, a cluster-based crossover (Line 15) and cluster-based mutation (Line 16) are used to generate offspring. Two new individual generation strategy is also used to improve the search performance of the C-BGA (Line 17–27).

5	6	1	4	2	3

Fig. 4. An example of individual coding.

4.2.1. *Coding*

The encoding and decoding method are crucial for genetic algorithm and directly affects the generation of solutions. C-BGA uses an integer number encoding method. Each gene represents a task to be executed. The advantage of such encoding is that the genes within the C-BGA individual are unique. Therefore, the decoding does not require the use of any additional repair methods to ensure the correctness of the solution. The TAA is used in the C-BGA to accomplish the decoding process.

Fig. 4 shows a C-BGA individual encoding example of one solution to the SRSP problem containing 6 tasks, where “5”, “6”, “1” “4”, “2”, and “3” correspond to the tasks in the task set, respectively. When using TAA for decoding, it will try to arrange tasks in the order of the figure and obtain the plan.

4.2.2. *Heuristic population initialization method*

Algorithm 3: Heuristic Initial population generation

```
   Input: Task Set $T$, Population Size $|P|$
   Output: Initial Population $P_0$
1  Initialization: $i = 1$ ;
2  while $i$ not equals to $|P|$ do
3      if $i \leq \frac{1}{5}|P|$ then
4          $P_0 \leftarrow$ Use EST Initialization Method($T, l_1, l_2$) ;
5      if $i > \frac{1}{5}|P|$ and $i \leq \frac{2}{5}|P|$ then
6          $P_0 \leftarrow$ Use LET Initialization Method($T, l_1, l_2$) ;
7      if $i > \frac{2}{5}|P|$ and $i \leq \frac{3}{5}|P|$ then
8          $P_0 \leftarrow$ Use TP Initialization Method($T, l_1, l_2$) ;
9      if $i > \frac{3}{5}|P|$ and $i \leq \frac{4}{5}|P|$ then
10         $P_0 \leftarrow$ Use TD Initialization Method($T, l_1, l_2$) ;
11     if $i > \frac{4}{5}|P|$ then
12         Use Random Initialization Method($T$) ;
13     $l_1, l_2 \leftarrow$Choose two locations randomly ;
14     $i = i + 1$ ;
```

Four heuristic initialization methods and a random initialization population method are used in the C-BGA to generate a high-quality population. These four heuristic population initialization methods are called the initialization method based on the earliest allowable start time (**EST**), the initialization method based on the latest allowable end time (**LET**), the initialization method based on task profit sorting (**TP**), and the initialization method based on task duration sorting (**TD**) respectively.

Heuristic Rule 1: Initialization method based on the earliest allowable start time sorting (**EST**). According to the earliest allowable start time of tasks, a sequence is sorted in order from front to back, and individuals are generated according to the sorted order.

Heuristic Rule 2: Initialization method based on the latest allowable end time sorting (**LET**). According to the latest allowable end time of tasks, the sequence is sorted in order from front to back, and individuals are generated according to the sorted order.

Heuristic Rule 3: Initialization method based on task profit sorting (**TP**): According to the profit of tasks, tasks are sorted in descending order, and individuals are generated according to the sorted order.

Heuristic Rule 4: Initialization method based on task duration sorting (**TD**). According to the profit of tasks, tasks are sorted from

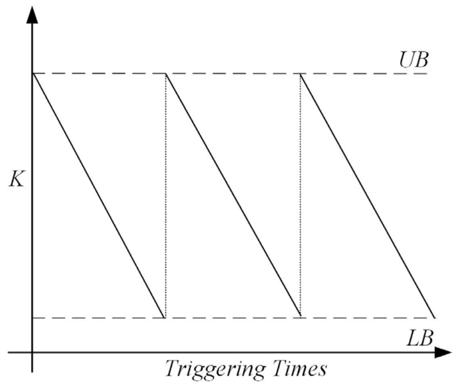


Fig. 5. Changing process of K value.

small to large, and individuals are generated according to the sorted order.

After adopting each heuristic rule to generate a population of individuals, it is necessary to adjust partial sequences to make individuals more diverse. Combining heuristics and random methods can make it easier for each individual to find a good search starting position. These four heuristic methods and random methods respectively occupy 20% individuals in the population. The pseudo-code of the heuristic population initialization method is shown as Algorithm 3.

4.2.3. Clustering method

The goal of the clustering method is to divide the sample set into multiple sub-sets based on data characteristics. Using this method has two advantages. On the one hand, the clustering method can make full use of satellite task data and guide the optimization process. On the other hand, this approach can cope with the lack of regularity in the field of satellite scheduling. Among these methods, the K-means clustering method is a typical type. The K-means clustering method divides the data into multiple clusters of discrete or hierarchical structures based on the similarity of data features. The final result of classification is to be as close as possible between samples in one cluster, and samples in different clusters are significantly different.

Feature information used by the clustering method is updated in the optimization process. An important feature that needs to be added is whether tasks are successfully scheduled. The feature matrix needs to be updated during the population search. Before clustering, normalization should be used. Through normalization, data will not be affected by units. The formula of normalization is as follows:

$$y' = \frac{y - y_{\min}}{y_{\max} - y_{\min}} \tag{15}$$

where, y is the feature value, $y_{\max}$ is the maximum value of feature y, and $y_{\min}$ is the minimum value of feature y.

A fixed value of K is likely to cause invalid searches when the optimization process reaches a certain stage. To improve the search efficiency, we choose to make the number of categories K change continuously in the range of [lower_bound, upper_bound]. When the condition is satisfied, the K value will decrease by 1. The process of K value change is shown in Fig. 5.

The pseudo-code of the K-means method is shown in Algorithm 4.

```
Algorithm 4: K-means Method
  Input: Set $D$, Feature Number $n$, Category $k$
  Output: $C$
1 Initialization: Random choose $k$ in $D$ to generate $\mu$ ;
2 while No update occurs in $\mu$ do
3     Let $C = \emptyset$ ;
4     for $i = 1, 2, ..., |D|$ do
5         $d_{ij} \leftarrow$ Calculate Euclidean distance $\left\| x_i, \mu_j \right\|_2$ ;
6         $\lambda_j \leftarrow \arg\min_{j \in (1,2,...,K)} d_{ij}$ ;
7         $C_{\lambda_i} \leftarrow C_{\lambda_i} \cup \{x_i\}$ ;
8     for $j = 1, 2, ..., |k|$ do
9         $\mu_j{}' \leftarrow \frac{1}{|C_j|} \sum_{x \in C_j} x$ ;
10        if $\mu_j{}'$ not equals to $\mu_j$ then
11            $\mu_j \leftarrow$ Update $\mu_j{}'$ ;
```

4.2.4. Fitness function

Fitness function is the basis for genetic evolution, selection, cluster-based crossover, and mutation. The performance of individuals in the population is also evaluated by the fitness function. The objective function is used as our fitness function, and the fitness value of an individual is calculated by Eq. (1).

4.2.5. Selection

The selection operation contains two aspects, one is to select an individual, and the other is to select a category to which a task belongs. Individuals and categories selected will be used in the crossover or mutation process. The selection operation needs to reflect performance differences between individuals in the population. The **Roulette Wheel method** is used according to the fitness function, which makes the individual with better performance and categories to be selected easier. The equation for selecting individuals of **Roulette Wheel method** is as follows:

$$\bar{p}_l = \frac{f_l}{\sum_{l \in P} f_l} \tag{16}$$

where $\bar{p}_l$ is the probability of individual l, f_l is the fitness of individual l.

After selecting individuals for follow-up evolution operation, it is necessary to select categories based on the results of classification. The equation for calculating fitness value based on categories is shown below.

$$\hat{p}_m = \frac{\sum_{l \in P} \sum_{t_m \in T_m} f_{t_m j}}{\sum_{l \in P} \sum_{i \in T} f_{il}}, \forall m \in M \tag{17}$$

where, $\hat{p}_m$ is the probability of category m, f_m is the fitness of category m.

After using **Roulette Wheel** to select individual and task categories respectively, cluster-based crossover and mutation operations can be performed.

4.2.6. Cluster-based crossover

A new crossover method based on the clustering method is proposed in C-BGA, called cluster-based crossover. Cluster-based crossover introduces a new concept, the neighborhood of cluster, that is, several categories that are close to one category.

After selecting an individual and one of the categories for crossover, it is necessary to use the neighborhood of the cluster to determine another task category. The other task category used for crossover requires both to be within the previously selected neighborhood and outside the task category that has been selected.

After obtaining the two categories for crossover, equal-length gene fragments are selected belonging to the two categories respectively. Then, the two gene segments are swapped to complete the crossover operation.

4.2.7. Cluster-based mutation

There are two types of mutation based on clustering, one is called mutation within the same category, and the other is called mutation within different categories. Using these two mutation methods not only can exchange execution positions between tasks that have similar data characteristics, but also exchange between large characteristic-differences tasks. Using two mutation methods can increase the diversity of mutations and let the algorithm find a solution.

Mutation within the same category: Mutation within the same category refers to exchange positions between two tasks in the same category obtained by the clustering method. In this way, a new individual is obtained and added to the population.

Mutation between different categories: Select two tasks for position exchange from different categories, and two tasks in different categories are exchanged. In this way, a new individual is obtained and added to the population.

These two mutation methods have the same possibility of being selected. When the mutation operation occurs, one of the mutation methods is randomly selected to generate a new individual.

4.2.8. Termination condition

After the optimization process to a certain stage, the algorithm stops and outputs the optimal solution, which is the final task execution plan. We set the termination condition of the C-BGA algorithm to be when iterations reach the maximum optimization generation.

4.2.9. Complexity analysis

The solution method is mainly composed of a task arrangement algorithm, genetic algorithm, and K-means method. The task arrangement algorithm and K-means method are included in the main process of the genetic algorithm, which should be determined according to the overall complexity of the C-BGA algorithm. The time complexity of C-BGA algorithm is $O(Gen \times |T| \times |TW| \times |P|)$, where Gen denotes the max generation $|T|$ denotes the number of tasks, $|TW|$ denotes the number of time windows, $|P|$ denotes the population size.

5. Experimental results and discussions

5.1. Experiment settings

5.1.1. Experiment environment

All algorithms in the experiment are run by Matlab2020a under the configuration environment of Core I7-7700 3.6 GHz CPU, 8 GB memory, and Windows 10 operating system.

5.1.2. Comparative algorithms

Some algorithms solving satellite scheduling problems and other scheduling problems are selected as comparative algorithms. We choose a knowledge-based genetic algorithm, improved adaptive large neighborhood search algorithm, tabu-based adaptive large neighborhood search algorithm, and firework algorithm as comparative algorithms. In addition, we also use ILOG CPLEX 12.6 Version as one of the comparative algorithms. The maximum runtime of the Solver is set to 30 min. This time setting is due to the fact that in a real scenario, the algorithm needs to find the solution in a short time. Then, the satellites and ground stations can be fully used. Knowledge-based genetic algorithm (KBGA) combines the knowledge information of planning into a genetic algorithm, and an adaptive mechanism was used to let this select algorithm knowledge optimize [42]. The tabu-based adaptive large neighborhood search algorithm (ALNS/TPF) use tabu search strategy is used in an adaptive large neighborhood search framework to reduce the possibility of repeated searches [43]. The firework algorithm (FWA) generates sparks from explosions of fireworks and uses the fireworks population to search [44].

Each random search algorithm will run 30 times and record results. The optimization performance of the algorithm will be evaluated through multiple dimensions. To comprehensively evaluate the effect of the algorithm on the SRSP problem, we set up a variety of evaluation indicators. We will evaluate the random search algorithm from three aspects: best performance (denoted as Max), average performance (denoted as Avg), and worst performance (denoted as Min).

5.1.3. Instance settings

Experimental instances used in this section are generated by STK 11.2.0. The time range of tasks is within one day. There are three categories based on the number of instances, small-scale (S), medium-scale (M), and large-scale (L). For each scale, two types of scenes with low density (L) and high density (H) are set. We set 10 scenarios for each scale, 5 low-density scenarios, and 5 high-density scenarios, respectively. Small-scale instances are set between 100 and 300, medium-scale instances are set between 400 and 600, and large-scale instances are set between 800 and 1000. In each scale, there is an interval of 50 tasks between instances. All low-density tasks are distributed within 24 h. Small-scale with high-density instances are distributed within 9 h, medium-scale with high-density instances are distributed within 15 h, and large-scale with high-density instances are distributed within 21 h. For convenience, the “A-B-C” format is used to represent an instance. “A” represents task size, “B” represents density, and “C” represents the internal number in a group.

5.1.4. Parameters settings

In the C-BGA, the population size $|P|$ is set to 10, the max Generation Gen is set to 500, $Threhold_1$ is set to 10, $Threhold_2$, $Threhold_3$ are set to 20, the probability of crossover is set to 0.9 and the probability of mutation is set to 0.05. The parameter settings of the comparative algorithm remain consistent with those in the literature.

5.2. Result analysis

5.2.1. Planning results under different task scales

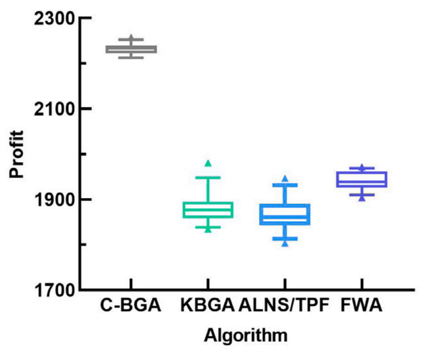
All 30 instances with different densities are used to verify the scheduling performance of algorithms and results are shown in Table 2. The increase in instance scale is accompanied by growing in profit. When the instance scale increases to medium, the growth rate of profit begins to slow down. This is due to the limited resources of satellites and ground stations. It is extremely difficult to find a better execution plan among numerous tasks. C-BGA has good performance in all 30 instances. In small-scale and medium-scale instances, scheduling results from the best to the worst are followed by C-BGA, FWA, ALNS/TPF, and KBGA. In large-scale scenarios, the optimization performances of KBGA and ALNS/TPF are between C-BGA and FWA. It can be seen from an overall trend that the C-BGA is easier to obtain higher profit than other comparative algorithms in larger-scale instances. ILOG CPLEX can find the theoretical optimal value. However, since the SRSP problem is NP-hard and the maximum runtime of the algorithm is limited, the results reflect the solution performance of ILOG CPLEX is relatively poor. The Solver search performance begins to decline after the task scale reached 200, and it is difficult to find a good solution when instance density or scale increases.

5.2.2. Algorithm stability analysis

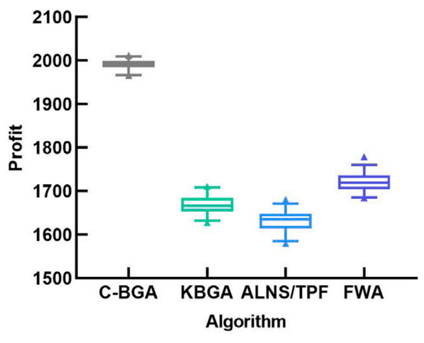
Large-scale instances are easier to reflect the stability of algorithms. Box plots of L-L-5 and L-H-5 instances are shown in Fig. 6(a) and (b). The C-BGA has the best stability, maintaining small volatility when the task scale is 1000. ALNS/TPF performs the worst, which is the easiest to obtain results with a large deviation. The stability results of FWA and KBGA are between the scheduling results of C-BGA and ALNS/TPF.

5.2.3. Significance test

Statistical analysis is also meaningful for evaluating random search algorithms. Wilcoxon rank sum test is taken on C-BGA with KBGA, ALNS/TPF, and FWA in pairs. Results are shown in Table 3. These results clearly show that the proposed algorithm is better than comparative algorithms except for a few instances. In terms of statistical results, there is a significant difference between results obtained by the C-BGA and comparative algorithms.



(a) L-L-5



(b) L-H-5

Fig. 6. Box plots of L-L-5 and L-H-5 instances.

Table 2
Scheduling results of 30 instances.

Instance	C-BGA			KBGA			ALNS/TPF			FWA			ILOG
	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	
S-L-1	**522**	**522**	**522**	**522**	516.6	509	**522**	518.6	515	**522**	521.03	519	519
S-L-2	**777**	**772.4**	**767**	763	748.37	739	762	752.7	738	769	762.2	758	708
S-L-3	**1016**	**1009.5**	**1003**	986	947.13	929	980	961.6	944	998	979.37	967	992
S-L-4	**1212**	**1187.03**	**1169**	1171	1135.2	1118	1159	1147.5	1134	1178	1166.07	1154	1061
S-L-5	**1293**	**1265**	**1246**	1222	1184.6	1155	1227	1197.77	1180	1240	1218.53	1206	399
S-H-1	**353**	**348.13**	**342**	336	319.9	302	341	324.8	316	344	335.80	329	345
S-H-2	**504**	**489.23**	**472**	471	438	415	458	438.77	420	475	461.50	450	220
S-H-3	**632**	**619.87**	**608**	577	544.77	524	582	552.27	538	588	572.53	561	586
S-H-4	**637**	**613.7**	**599**	570	541.03	515	572	552.17	536	590	574.87	562	577
S-H-5	**661**	**642.4**	**626**	597	572.7	553	606	575.17	547	627	605.17	593	437
Ave	**760.7**	**746.93**	**735.4**	721.5	694.83	675.9	720.9	702.13	686.8	733.1	719.71	709.9	584.4
M-L-1	**1576**	**1553.73**	**1527**	1447	1416.27	1377	1476	1437.9	1394	1490	1472.1	1460	1447
M-L-2	**1547**	**1517.3**	**1478**	1447	1395	1351	1431	1391.43	1369	1461	1438.1	1420	806
M-L-3	**1612**	**1592.07**	**1555**	1479	1445.47	1414	1462	1438.90	1408	1525	1485.73	1462	705
M-L-4	**1772**	**1743.7**	**1725**	1629	1564.7	1532	1614	1565.7	1523	1649	1613.93	1591	765
M-L-5	**1836**	**1816.47**	**1796**	1667	1612.97	1580	1671	1627.77	1566	1704	1672.67	1651	799
M-H-1	**1300**	**1274.67**	**1252**	1183	1153.2	1128	1192	1159.97	1135	1223	1203.07	1186	1191
M-H-2	**1358**	**1332.07**	**1319**	1237	1192.8	1171	1247	1205.97	1164	1276	1252.60	1231	537
M-H-3	**1344**	**1331.97**	**1311**	1233	1186.07	1134	1224	1188.57	1150	1267	1237.93	1220	627
M-H-4	**1434**	**1405.17**	**1378**	1305	1233.4	1196	1276	1225.27	1179	1311	1282.93	1259	629
M-H-5	**1556**	**1538.47**	**1522**	1323	1281.97	1246	1314	1277.1	1223	1383	1341	1315	762
Ave	**1533.5**	**1510.56**	**1486.3**	1395	1348.18	1312.9	1390.7	1351.86	1311.1	1428.9	1400.01	1379.5	826.8
L-L-1	**2118**	**2083.2**	**2065**	1825	1770.53	1725	1811	1757.4	1688	1842	1820.57	1796	962
L-L-2	**2046**	**2013.47**	**1990**	1769	1725.1	1685	1750	1709.57	1662	1819	1784.4	1756	3
L-L-3	**2210**	**2177.37**	**2160**	1872	1776.23	1721	1825	1756.93	1678	1858	1829.93	1807	1064
L-L-4	**2165**	**2141.53**	**2118**	1838	1810.8	1773	1881	1810.37	1730	1929	1893.47	1865	3
L-L-5	**2257**	**2231.8**	**2212**	1980	1878.13	1834	1946	1866.5	1803	1971	1940.87	1903	871
L-H-1	**1910**	**1878.57**	**1861**	1702	1606.73	1568	1637	1580.87	1524	1683	1658.47	1638	780
L-H-2	**1929**	**1908.33**	**1888**	1614	1562.67	1525	1636	1566.33	1486	1662	1625.47	1595	901
L-H-3	**1847**	**1797.3**	**1769**	1561	1517.2	1483	1574	1513.83	1436	1613	1587.23	1566	721
L-H-4	**1839**	**1812.93**	**1784**	1590	1534.47	1495	1626	1535.13	1440	1635	1608.57	1591	597
L-H-5	**2011**	**1990.9**	**1965**	1710	1668.47	1627	1680	1631.13	1579	1778	1719.93	1684	549
Ave	**2033.2**	**2003.54**	**1981.2**	1746.1	1685.03	1643.6	1736.6	1672.81	1602.6	1779	1746.89	1720.1	645.1

5.2.4. *Optimization time analysis*

We compared the time required for the C-BGA to reach the same average performance as KBGA, and the results are shown in Figs. 7(a) and 7(b). The 100% bar in the figure represents the time taken by the KBGA for each instance, and the black bar represents the percentage of comparative algorithm time taken by the C-BGA. It can be seen from the results that the C-BGA only needs about one-third of KBGA's time to achieve the same optimization effect. In other words, a good-quality solution can be found within a short time after a population is initialized.

5.2.5. *Coverage analysis*

Due to the length of the article, we chose two instances, L-L-5, and L-H-5, to verify the convergence speed of the algorithm. Results in Figs. 8(a) and 8(b) clearly show that the convergence speeds of algorithms are different. FWA has a faster coverage in the initial stage of optimization, but it is easy to fall into a local optimum. KBGA's coverage speed is the slowest among several algorithms. C-BGA has a fast convergence speed and ensures that this algorithm is not easy to fall into local optimum through the update of the clustering method. These experimental results show that the proposed algorithm can obtain better optimization performance than comparative algorithms while maintaining good convergence speed.

5.2.6. *Comparison with the deep reinforcement learning method*

Deep reinforcement learning is also a method with good solution performance. We use the Deep Reinforcement Learning (DRL) proposed in [20] to compare the solution performance with the C-BGA. The

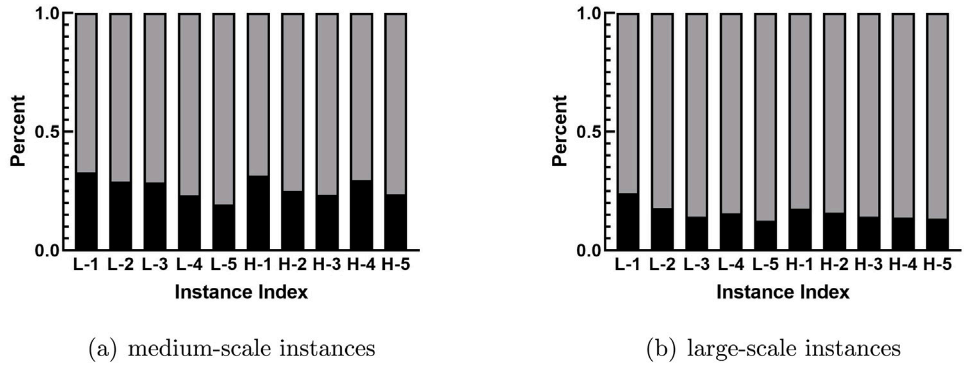


Fig. 7. Proportion of optimization time using different scheduling algorithms.

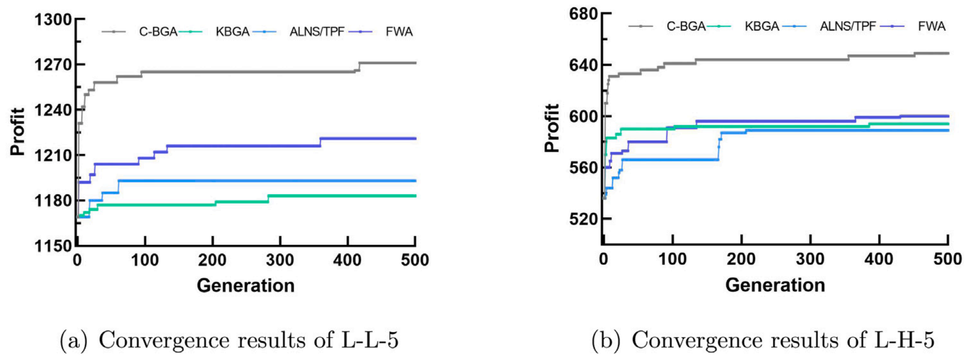


Fig. 8. Convergence results.

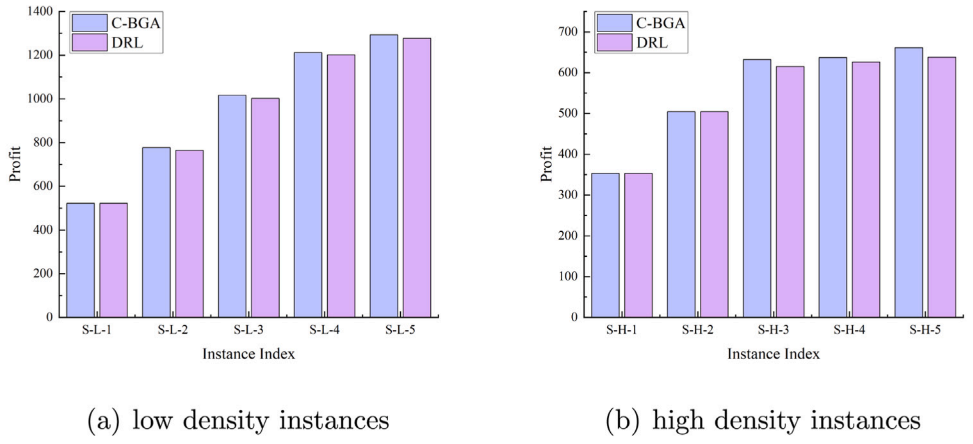
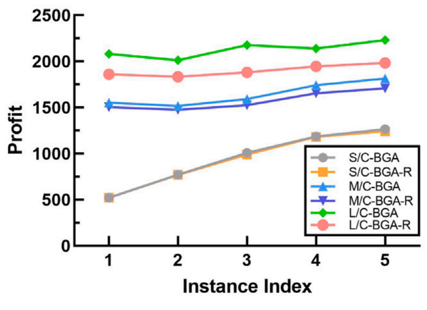


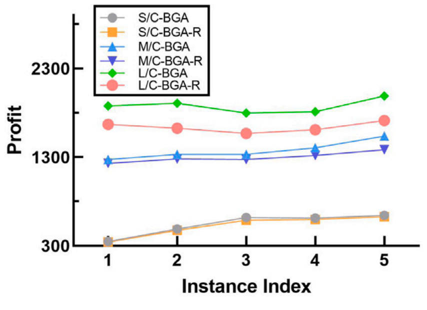
Fig. 9. Comparison of C-BGA and DRL.

Table 3
Wilcoxon rank sum test.

Algorithm pair	Instance set	Indicator	#Wins	#Ties	#Losses	p-value
C-BGA vs. KBGA	Low density (15)	Max	14	1	0	7.92E−05
		Avg	15	0	0	4.64E−05
	High density (15)	Max	15	0	0	2.25E−05
		Avg	15	0	0	9.56E−06
C-BGA vs. ALNS/TPF	Low density (15)	Max	14	1	0	8.58E−05
		Avg	15	0	0	9.21E−05
	High density (15)	Max	15	0	0	2.48E−05
		Avg	15	0	0	1.89E−05
C-BGA vs. FWA	Low density (15)	Max	14	1	0	2.11E−05
		Avg	15	0	0	1.98E−04
	High density (15)	Max	15	0	0	5.09E−05
		Avg	15	0	0	5.85E−05

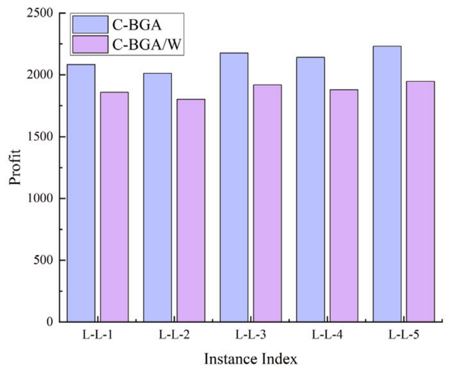


(a) low density instances

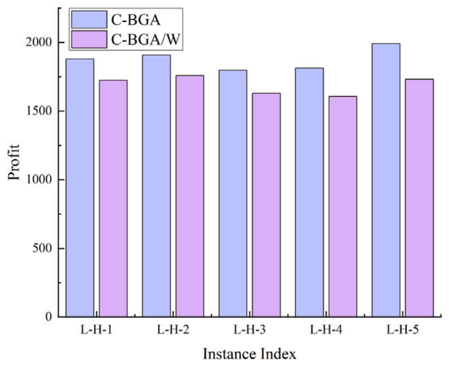


(b) high density instances

Fig. 10. Results of different population initialization strategies.



(a) low density instances



(b) high density instances

Fig. 11. Results of different population evolution strategies.

parameter settings of DRL remain the same as in the literature. Fig. 9 shows the experimental comparison results for a small-scale scenario, with C-BGA as the best value for the algorithm operation. The C-BGA obtains approximately equal gain values with the DRL method for instances of 50 and 100 task scales. In contrast, the solution performance of DRL is not as good as C-BGA when the task scale increases. This reflects that the proposed algorithm has stronger global search performance and is suitable for handling complex problem instances.

5.3. *Algorithm strategies analysis*

5.3.1. *Effectiveness of population initialization strategy*

To verify the effectiveness of improvements in C-BGA, a C-BGA with a random initialization method is used to compare with the heuristic population initialization method (denoted as C-BGAR). The results of the average performance of algorithms are shown in Fig. 10(a) and (b). In both low-density and high-density instances, it can be seen that the heuristic population initialization strategy has a more obvious improvement in profit as the scale of instances increases. This heuristic population initialization strategy makes it easier for the algorithm to find a high-quality solution when the distribution of instances is dense.

5.3.2. *Effectiveness of population evolution strategy*

This part compares C-BGA and C-BGA without cluster-based genetic operators (denoted as C-BGA/W) in large-scale instances. It can be seen from Fig. 11 that cluster-based genetic operators can help C-BGA to obtain better scheduling performance. This reflects that the K-means approach can fully use the data features to guide the population evolution process. Traditional genetic operators, on the other hand, are more stochastic and may consistently fail to find higher-quality solutions. These experiments demonstrate the effectiveness of cluster-based crossover and cluster-based mutation in C-BGA.

5.3.3. *K change method analysis*

In our solution method, it is mentioned that the K value changes within an interval, and the number of cluster categories is determined dynamically. To verify the rationality of the K value change method, we selected the other four K value change methods for comparison. The first method is based on the triggering value change condition (denoted as C-GBAK1). The value of K is reduced by one until it reaches the lower bound and remains unchanged. The second method is to change the K value directly to the lower bound value after reaching a certain percentage of the maximum optimization generation (denoted as C-GBAK2), and the percentage here is set to 0.2. The third method is based on triggering the K value change condition (denoted as C-GBAK3). The value is reduced by one until it reaches the lower bound, warms to the value of the average of upper and lower bounds, and then keeps K unchanged after falling again to the lower bound. The last method is that K is a fixed value and remains unchanged during the whole optimization process (denoted as C-GBAK4).

As shown in Figs. 12(a) and 12(b), adopting the K value change method in the C-BGA algorithm can bring significant optimization improvement in the vast majority of instances. Only two instances of L-H-3 and L-H-3 do not show significant improvement. However, there is little difference between the other four K value change methods. The results also indicate that the K change method in the C-BGA is more effective when the density is low. It is effective to adopt the K value change method in the algorithm.

It can be seen from the above experimental results that the proposed algorithm has achieved good optimization performance in multiple instances. It is also evidence that C-BGA can effectively solve the SRSP problem. Besides, it is not difficult to find that the C-BGA performs well in large-scale instances, which shows that if there are numerous tasks in practical applications in the future, this algorithm can still obtain a satisfied task execution plan.

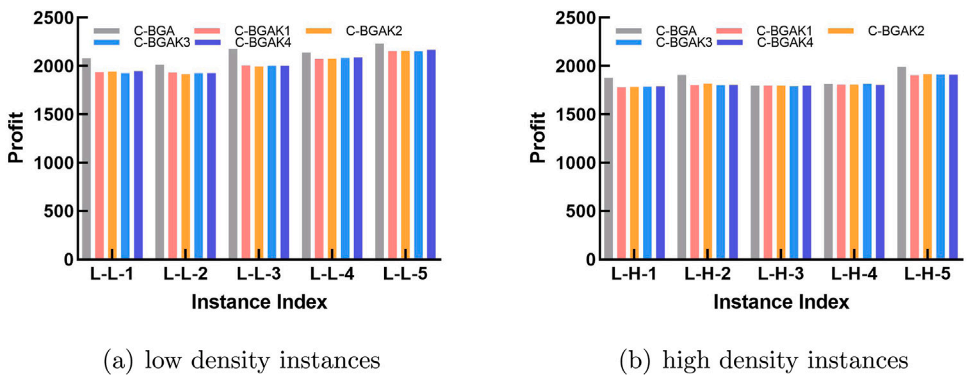


Fig. 12. Results of changing methods.

6. Conclusion

This paper focuses on the satellite ranging scheduling problem in an effort to improve satellite management capabilities and promote aerospace technologies. An improved genetic algorithm based on clustering is designed to solve the SRSP problem. C-BGA adopts a variety of strategies to improve algorithm optimization performance and obtain higher-quality solutions. A heuristic initialization strategy can obtain a good initial solution while ensuring differences between individuals. This method helps a lot in the subsequent optimization process. Clustering methods and cluster-based operations are two key parts of the C-BGA. Solutions are found through feature information updates and cluster-based population evolution. The proposed algorithm solves the SRSP problem well and has the prospect of being well used in actual satellite range scheduling systems.

In future research, more artificial intelligence methods will be considered, such as reinforcement learning and deep learning. How to use the trained model to solve online scheduling problems will also become an important direction for improving the operational capability of a satellite system. More complex conditions will also be studied.

CRediT authorship contribution statement

Yanjie Song: Methodology, Software, Writing – original draft. **Junwei Ou:** Writing – original draft, Data curation. **Jian Wu:** Visualization, Investigation. **Yutong Wu:** Writing – review & editing. **Lining Xing:** Validation, Methodology, Funding acquisition, Methodology. **Yingwu Chen:** Formal analysis, Resources.

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

Please send an email to the author with a description of the purpose of the data.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (72001212), the Science and Technology Innovation Team of Shanxi Province (2023-CX-TD-07), the Special Project in Major Fields of Guangdong Universities (2021ZDZX1019), and the Hunan Key Laboratory of Intelligent Decision-making Technology for Emergency Management (2020TP1013).

References

[1] X. Chen, X. Li, X. Wang, Q. Luo, G. Wu, Task scheduling method for data relay satellite network considering breakpoint transmission, IEEE Trans. Veh. Technol. 70 (1) (2020) 844–857.
[2] 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 Evol. Comput. (2023) 101236.
[3] D. Karapetyan, S.M. Minic, K.T. Malladi, A.P. Punnen, Satellite downlink scheduling problem: A case study, Omega 53 (2015) 115–123.
[4] G. Peng, R. Dewil, C. Verbeeck, A. Gunawan, L. Xing, P. Vansteenwegen, Agile earth observation satellite scheduling: An orienteering problem with time-dependent profits and travel times, Comput. Oper. Res. 111 (2019) 84–98.
[5] G. Peng, G. Song, L. Xing, A. Gunawan, P. Vansteenwegen, An exact algorithm for agile earth observation satellite scheduling with time-dependent profits, Comput. Oper. Res. 120 (2020) 104946.
[6] N. Zufferey, M. Vasquez, A generalized consistent neighborhood search for satellite range scheduling problems, RAIRO-Oper. Res. 49 (1) (2015) 99–121.
[7] J. Zhang, L. Xing, An improved genetic algorithm for the integrated satellite imaging and data transmission scheduling problem, Comput. Oper. Res. 139 (2022) 105626.
[8] X. Chen, G. Reinelt, G. Dai, A. Spitz, A mixed integer linear programming model for multi-satellite scheduling, European J. Oper. Res. 275 (2) (2019) 694–707.
[9] P. Tangpattanakul, N. Jozefowiez, P. Lopez, A multi-objective local search heuristic for scheduling Earth observations taken by an agile satellite, European J. Oper. Res. 245 (2) (2015) 542–554.
[10] L. He, J. Li, M. Sheng, R. Liu, K. Guo, D. Zhou, Dynamic scheduling of hybrid tasks with time windows in data relay satellite networks, IEEE Trans. Veh. Technol. 68 (5) (2019) 4989–5004.
[11] X. Zhu, J. Wang, X. Qin, J. Wang, Z. Liu, E. Demeulemeester, Fault-tolerant scheduling for real-time tasks on multiple earth-observation satellites, IEEE Trans. Parallel Distrib. Syst. 26 (11) (2014) 3012–3026.
[12] K.-J. Zhu, J.-F. Li, H.-X. Baoyin, Satellite scheduling considering maximum observation coverage time and minimum orbital transfer fuel cost, Acta Astronaut. 66 (1–2) (2010) 220–229.
[13] T.D. Gooley, Automating the Satellite Range Scheduling Process, Tech. rep., Air Force Inst of Tech Wright-PattersonAFB Oh School of Engineering, 1993.
[14] K. Luo, H. Wang, Y. Li, Q. Li, High-performance technique for satellite range scheduling, Comput. Oper. Res. 85 (2017) 12–21.
[15] N. Zufferey, P. Amstutz, P. Giaccari, Graph colouring approaches for a satellite range scheduling problem, J. Sched. 11 (4) (2008) 263–277.
[16] F. Marinelli, S. Nocella, F. Rossi, S. Smriglio, A Lagrangian heuristic for satellite range scheduling with resource constraints, Comput. Oper. Res. 38 (11) (2011) 1572–1583.
[17] Z. Zhang, N. Zhang, Z. Feng, Multi-satellite control resource scheduling based on ant colony optimization, Expert Syst. Appl. 41 (6) (2014) 2816–2823.
[18] Z. Zhang, F. Hu, N. Zhang, Ant colony algorithm for satellite control resource scheduling problem, Appl. Intell. 48 (10) (2018) 3295–3305.
[19] Y.-J. Song, X. Ma, X.-J. Li, L.-N. Xing, P. Wang, Learning-guided nondominated sorting genetic algorithm II for multi-objective satellite range scheduling problem, Swarm Evol. Comput. 49 (2019) 194–205.
[20] 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 Evol. Comput. (2023) 101233.
[21] D.A. Parish, A Genetic Algorithm Approach to Automating Satellite Range Scheduling, Tech. rep., Air Force Inst of Tech Wright-Patterson AFB Oh School of Engineering, 1994.
[22] J. Sun, F. Xhafa, A genetic algorithm for ground station scheduling, in: 2011 International Conference on Complex, Intelligent, and Software Intensive Systems, IEEE, 2011, pp. 138–145.

[23] F. Xhafa, J. Sun, A. Barolli, A. Biberaj, L. Barolli, Genetic algorithms for satellite scheduling problems, Mob. Inf. Syst. 8 (4) (2012) 351–377.
[24] F. Xhafa, A. Barolli, M. Takizawa, Steady state genetic algorithm for ground station scheduling problem, in: 2013 IEEE 27th International Conference on Advanced Information Networking and Applications, AINA, IEEE, 2013, pp. 153–160.
[25] 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.
[26] Z. Zheng, J. Guo, E. Gill, Swarm satellite mission scheduling & planning using hybrid dynamic mutation genetic algorithm, Acta Astronaut. 137 (2017) 243–253.
[27] B. Song, F. Yao, Y. Chen, Y. Chen, Y. Chen, A hybrid genetic algorithm for satellite image downlink scheduling problem, Discrete Dyn. Nat. Soc. 2018 (2018).
[28] J. Berger, E. Giasson, M. Florea, M. Harb, A. Teske, E. Petriu, R. Abielmona, R. Falcon, N. Lo, A graph-based genetic algorithm to solve the virtual constellation multi-satellite collection scheduling problem, in: 2018 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2018, pp. 1–10.
[29] M. Barkaoui, J. Berger, A new hybrid genetic algorithm for the collection scheduling problem for a satellite constellation, J. Oper. Res. Soc. 71 (9) (2020) 1390–1410.
[30] Y.-B. Woo, B.S. Kim, Matheuristic approaches for parallel machine scheduling problem with time-dependent deterioration and multiple rate-modifying activities, Comput. Oper. Res. 95 (2018) 97–112.
[31] L.R. de Abreu, B. de Athayde Prata, A genetic algorithm with neighborhood search procedures for unrelated parallel machine scheduling problem with sequence-dependent setup times, J. Model. Manag. (2020).
[32] H.A. Khojah, M.A. Mosa, Multi-objective optimization for multi-satellite scheduling task: Multi-objective optimization for multi-satellite scheduling task, J. Soft Comput. Explor. 3 (1) (2022) 19–30.
[33] Y. Song, B. Song, Y. Huang, L. Xing, Y. Chen, Solving large-scale relay satellite scheduling problem with a dynamic population firework algorithm: A case study, in: 2021 IEEE Symposium Series on Computational Intelligence, SSCI, IEEE, 2021, pp. 1–7.
[34] M.E. Celebi, Partitional Clustering Algorithms, Springer, 2014.
[35] F. Murtagh, P. Contreras, Algorithms for hierarchical clustering: an overview, Wiley Interdiscip. Rev.: Data Min. Knowl. Discov. 2 (1) (2012) 86–97.
[36] R. Dondo, J. Cerdá, A cluster-based optimization approach for the multi-depot heterogeneous fleet vehicle routing problem with time windows, European J. Oper. Res. 176 (3) (2007) 1478–1507.
[37] D.B. Coral, M.O. Santos, C. Toledo, L.F. Niño, Clustering-based search in a memetic algorithm for the vehicle routing problem with time windows, in: 2018 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2018, pp. 1–8.
[38] J.G.C. Costa, Y. Mei, M. Zhang, Cluster-based hyper-heuristic for large-scale vehicle routing problem, in: 2020 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2020, pp. 1–8.
[39] N. Bianchessi, G. Righini, Planning and scheduling algorithms for the COSMO-SkyMed constellation, Aerosp. Sci. Technol. 12 (7) (2008) 535–544.
[40] L. Barbulescu, A.E. Howe, J.-P. Watson, L.D. Whitley, Satellite range scheduling: A comparison of genetic, heuristic and local search, in: International Conference on Parallel Problem Solving from Nature, Springer, 2002, pp. 611–620.
[41] A.J. Vazquez, R.S. Erwin, Robust fixed interval satellite range scheduling, in: 2015 IEEE Aerospace Conference, IEEE, 2015, pp. 1–6.
[42] Y. Song, L. Xing, M. Wang, Y. Yi, W. Xiang, Z. Zhang, A knowledge-based evolutionary algorithm for relay satellite system mission scheduling problem, Comput. Ind. Eng. 150 (2020) 106830.
[43] L. He, M. de Weerdt, N. Yorke-Smith, Time/sequence-dependent scheduling: the design and evaluation of a general purpose tabu-based adaptive large neighbourhood search algorithm, J. Intell. Manuf. 31 (4) (2020) 1051–1078.
[44] J. Li, Y. Tan, The bare bones fireworks algorithm: A minimalist global optimizer, Appl. Soft Comput. 62 (2018) 454–462.