Complex & Intelligent Systems (2023) 9:5157–5172
https://doi.org/10.1007/s40747-023-01008-4

ORIGINAL ARTICLE

A co-evolutionary algorithm with elite archive strategy for generating diverse high-quality satellite range schedules

Minghui Xiong[1] · Wei Xiong[1] · Zheng Liu[1]

Received: 21 November 2022 / Accepted: 17 February 2023 / Published online: 10 March 2023

Abstract
Satellite range scheduling, a multi-constrained combinatorial optimization problem, is crucial to guaranteeing the normal operation and application of onboard satellites. Traditional methods are dedicated to finding one optimal schedule, having ignored the problem may process multiple high-quality schedules. To provide a set of alternative schedules while maintaining the solution quality, we propose a co-evolutionary algorithm with elite archive strategy (COEAS) in this article. In COEAS, two populations are evolved to solve the original and relaxed problem in terms of schedule quality and diversity, respectively. During the evolution, the populations maintain a weak cooperation and only share the information in offspring combination phase. Further, an elite archive strategy is derived to identify and preserve potential stagnated and optimal individuals. In this strategy, the promising individuals would further participate in parent mating and offspring replacement for the dual purpose of maintaining potential optima recovery and fine-tuning the population. The experimental results show that the proposed algorithm is better than comparison algorithms in terms of efficacy (obtaining higher quality schedule), diversity (locating more optimal schedules) and flexibility (providing better alternatives).

Keywords Satellite range scheduling · Evolutionary algorithm · Coevolution · Archive technique

Introduction

Satellites are equipped with various space-borne payloads to complete different military and civilian applications, such as reconnaissance and surveillance, missile warning, environmental and disaster detection, navigation and positioning, communications and broadcasting. To support these services, the satellite users request frequent satellite-ground communications with ground stations. These communications include satellite tracking, telemetry, data transmission and command betting. The satellite-ground communication is achieved through requesting visible time windows at the ground station. Then the ground station should serve the corresponding request in the given time window. The scheduling of allocating the time window to the requests is called satellite range scheduling problem [1] (SRSP). The goal of SRSP is to rationally allocate the visible time windows, satisfy the satellite user requests and resource management preference.

In SRSP, all conductions of satellite-ground communication should be subjected to the following time and resource constraints: (1) each request should be served within the visible time window; (2) a request should be served at most once; (3) an antenna supports at most one request at a time; (4) each antenna requires enough switch time before next service. Optimally schedule these multi-constrained resources into an optimal satellite range schedule has been proven to be NP-complete [2]. Furthermore, the ground station antennas are oversubscribed due to the contradiction between increasing user requests and limited ground stations. Hence, finding an optimal satellite range schedule is becoming a challenging problem.

To tackle the oversubscribed characteristic of SRSP under time and resource constrains, some techniques have been developed to satisfy as much requests as possible. In sum-

Wei Xiong and Zheng Liu have contributed equally to this work.

✉ Minghui Xiong
xtkxxmh@163.com

Wei Xiong
13331094335@163.com

Zheng Liu
neverlinever@163.com

[1] Science and Technology on Complex Electronic System Simulation Laboratory, Space Engineering University, Beijing 101416, China

mary, the proposed methods can be roughly classified into the following three types:

(1) *Exact algorithm*: Some researchers formulate the scheduling problem as a linear or nonlinear programming problem and solve it with mathematical programming approaches, such as mixed integer programming with Lagrangian-relaxation [3], branch and cut [4], exact polynomial time algorithm [5] and dynamic programming [6], etc. The exact algorithms explore the entire search space and return the same single optimal solution to the same input. As the scale of input request and resource increases, the computation burden of exact algorithms may become unacceptable due to the exponential growth of search space.
(2) *Local search algorithm*: To reduce the search space, the local search algorithms start the search from an initial solution, iteratively search the neighbor region to improve the schedule quality and finally output a high-quality schedule. For example, the greedy algorithm [7], tabu search [8] and simulated annealing [9] have been adapted to solve SRSP. Based on a conflict-resolution technique, Luo et al. [10] proposed a rescheduling strategy to rapidly improve the low-quality schedule.
(3) *Evolutionary algorithm*: This kind of approaches employ a population to optimize the schedule, which includes two main steps: reproduction and replacement. The overall performance enhancement of introducing objective-specified and problem-specified heuristic information into population reproduction procedure has been demonstrated in the literature [11–15]. Different from exact algorithms, the evolutionary algorithms avoid exploring entire search space and provide a set of solutions for decision makers to choose a final satisfying schedule.

In summary, most previous efforts concentrate on the NP-complete nature of SRSP and seek to locate one optimal schedule, regardless of the fact that there may be multiple high-quality schedules. In SRSP, it is appealing to provide diverse optimal alternatives to decision makers for (1) quicker schedule switching under emergency events; (2) more balanced antenna load due to different choices of antennas and (3) obtaining robustness against imperfect modeling. There are several related studies of seeking diverse high-quality solutions for different problems, such as automatic itinerary planning [16], patient admission scheduling [17], traveling salesman problem [18], permutation flow-shop scheduling [19], etc. However, generating multiple optimal satellite range schedules for decision makers still remains to be explored. Therefore, an effective algorithm is highly desired to solve SRSP with a set of high-quality solutions, which requires two critical issues to be considered: (1) the choice of the baseline algorithm and framework for promising search capability; (2) the technique to maintain population diversity during evolution.

Different from exact algorithm and local search algorithm, the population-based evolution algorithm solves the problem with a set of individuals, which is naturally suitable for exploring and maintaining diverse optimal solutions. However, the population tend to converge to a local optimum during the evolution, affecting the algorithm capability of finding multiple optima. The archive technique, which adapts separate archives for population convergence, diversity and potential optima preservation, has been proposed and employed to solve the problem. The two-archive algorithm [20] is the first evolutionary algorithm to promote population convergence and diversity with two collaborative archives. Li et al. [21] proposed a restricted mating selection mechanism to leverage the complementary effects of both archives, which utilize the diversity-oriented archive to explore the regions under-exploited by the convergence-oriented archive. Differently, the weak cooperation framework is adapted in [22,23], in which the two archives only interact after the offspring generation. In [24], Liu et al. developed a niche-based clearing strategy to guarantee diversity in the search space. Apart from separating the population based on convergence and diversity, the archive technique can be employed as a supporting archive during the evolution. Wang et al. [25] detects the stagnated individual with a stagnation counter and preserve these solutions in the archive. In [26], the archive individuals participate in the mutation operation to facilitate the evolution of population.

Inspired by the successful impletions of the evolutionary algorithm and archive technique, a co-evolutionary algorithm with elite archive strategy is proposed to address the above issues. The algorithm simultaneously maintains two collaborative population and an external archive: the convergence-oriented population P_C, the diversity-oriented population P_D and an elite archive EA. Our aim is to demonstrate that multiple high-quality schedules for SRSP can be found in a single run. The primary contributions of this work are as follows:

(1) A cooperative co-evolutionary mechanism is proposed to evolve the two populations, with a relatively weak collaboration, for solving SRSP. Population P_C drives the population to maintain the convergence and feasibility during evolution; P_D improves the population diversity in decision space without considering the constraints. The two population only interact in the parent mating selection and offspring population combination phase. Furthermore, a learning-guided offspring generation approach is developed to generate the diversified high-quality solutions with more efficiency.

(2) An elite archive strategy is designed and implemented with the dual purpose of helping maintaining the potential optima and fine-tuning the population. EA is updated by current optimal solutions to preserve the potential optima during the evolution and protect the founded global optimal solutions. Moreover, the elite individuals are allowed to participate in the parent mating selection and population fine-tuning. To reduce the computation burden resulted from performing local search in each generation, we detect the population evolution stagnation with a stagnation counter δ. Once δ exceeds the pre-defined threshold, the replacement operation would be performed to fine-tune the population and improve the exploration ability.

(3) The experimental results validate that the above proposed strategies can improve the quality and diversity of final output schedules. Compared with the rivals, our method can obtain more diversified high-quality schedules in a single run. Different from the traditional single optima solution, the set of solutions obtained by COEAS can provide the decision makers with more alternatives to select while guaranteeing the schedule quality. The results of COEAS can be adapted as the baselines for further studies on finding diverse SRSP schedules, and our research idea can provide reference to other similar scheduling problems.

In the remainder of this article, we first describe the SRSP. In the next section, a new cooperative co-evolutionary algorithm for the problem is presented. Experimental results and analysis is described in the subsequent section. Finally, we give our conclusions.

Problem description

Under the time and resource constraints, the satellite range scheduling problem is to determine the supported requests and the corresponding antenna allocation as well as the execution start time. In practical applications, diverse high-quality satellite range schedules are desired. For example, the decision maker can select the most preferred solution between the equally good optima, instead of being limited to one option. In addition, the antenna load could be more balanced with different antenna resource allocation in each schedule. Moreover, switching to an alternative schedule would be easier and faster than re-scheduling to tackle environment uncertainty (e.g., antenna breakdown). Motivated by the application demands, we address the problem of achieving diverse high-quality satellite range schedule in a single run.

As shown in Fig. 1, the crucial point of solving SRSP is to tackle the potential conflicts in the same antenna (e.g., $vtw_{4,2}$ and $vtw_{6,2}$) and the resource competition between the antenna (e.g., $vtw_{3,1}$ and $vtw_{3,2}$) without constraint violation. The visible time window (VTW) intersection in the same antenna results in the potential failure, while the alternative VTW brings the multiple optimal solutions characteristics of SRSP.

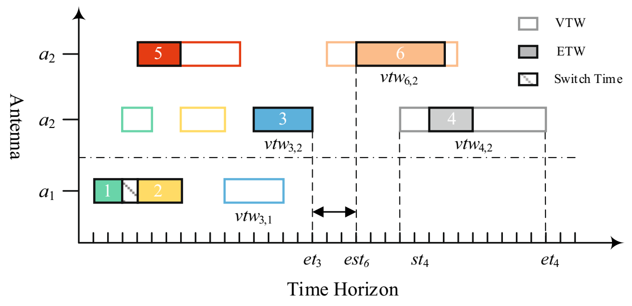

Fig. 1 A SRSP of six requests in a discrete time period

Parameter definition

We first define following parameters to describe the SRSP:

$$\text{SRSP} = \{R, A, S, VTW\}, \tag{1}$$

where

- $R = \{1, 2, ..., r, ...N_R\}$ denotes the set of N_R requests.
- Each request $r = \{s_{id}, erst, duet, dur, p\}$ is determined by the required satellite identification s_{id}, earliest start time $erst$, latest deadline ldt, required duration dur, and the priority weight p.
- $A = \{1, 2, ..., a, ...N_A\}$ is the set of N_A ground station antennas.
- Antenna $a = \{a_{id}, swi\}$ is specified by the antenna identification a_{id} and the switch time swi to serve next request.
- $S = \{1, 2, ..., s, ...N_S\}$ is the set of satellites with $|S| = N_S$.
- $\text{VTW} = \{1, 2, ..., vtw, ...N_{\text{VTW}}\}$ is the set of $|N_{VTW}|$ time windows.
- $vtw_r = \{vtw_{r,a}^k \mid k = 1, 2, ...K\}$ means the ground station antennas have K vtws.
- $\forall vtw_{r,a}^k \in vtw_r$, $vtw_{r,a}^k = \{r, a, k, st, et\}$. $vtw_{r,a}^k$ means the kth time window that the ground station network can support request r, which ranges from start time st to end time et. And the vtw is provided by antenna a. For example, the number 2 in $vtw_{6,2}$ in Fig. 1 denotes the vtw is provided by antenna a_2, rather than the second vtw for request 6.

Mathematical model construction

Based on the above notations, the mathematical model of the SRSP could be formulated as follows.

Objective function

The objective function is to minimize the request failure rate over the planning horizon:

$$\min f = 1 - \sum_{r=1}^{N_R} x_{r,a}^{k} / N_R, \tag{2}$$

where N_R is the number of input requests and $x_{r,a}^{k}$ is a binary variable representing whether request r is supported by antenna a in its kth visible time window.

Constraints

The constraints of SRSP restrict the feasibility of the schedules. In this paper, the constraint conditions we consider are enumerated as follows:

- $Execution\ uniqueness$: Each request is supported at most once;

$$\forall\, r \in R, \quad \sum_{k=1}^{K} x_{r,a}^{k} \le 1 \tag{3}$$

- $Satellite\ uniqueness$: One satellite can interact with as most one antenna simultaneously;

$$\forall t \in T,\ s \in S,\ a \in A, \quad \sum_{t=1}^{|T|} \sum_{s=1}^{|S|} x_{s,a}^{t} \le 1 \tag{4}$$

  where $x_{s,a}^{t}$ is a binary variable representing whether request r is supported by antenna a at time t.
- $Antenna\ uniqueness$: One antenna is unable to support more than two requests simultaneously;

$$\forall t \in T,\ s \in S,\ a \in A, \quad \sum_{t=1}^{|T|} \sum_{a=1}^{|A|} x_{s,a}^{t} \le 1 \tag{5}$$

  where $x_{s,a}^{t}$ is a binary variable representing whether request r is supported by antenna a at time t.
  The difference between constraint (4) and (5) lies in the accumulation way of $x_{s,a}^{t}$.
- $Execution\ feasibility$: Each request should be scheduled within its required time window;

$$\forall\, r \in R, \quad st_r \le est_r < ent_r \le et_r \tag{6}$$

  where $x_{s,a}^{t}$ is a binary variable representing whether request r is supported by antenna a at time t.
- $Switch\ time$: For one antenna, the minimal switch time must be satisfied to support next request. Suppose request r_1 and r_2 are adjacent requests on antenna a;

$$\left[st_{r_1}, et_{r_1} + swi_a\right] \cap \left[st_{r_2}, et_{r_2} + swi_a\right] = \emptyset \tag{7}$$

Proposed algorithm

In this section, we propose a cooperative co-evolutionary algorithm with elite archive strategy (COEAS) to generate diverse high-quality solutions for SRSP. The general framework and key procedures of the algorithm would be described in turn.

General framework

COEAS evolves convergence-oriented population P_{C} with the assistance of diversity-oriented population P_{D} and elite archive EA. The relationships between P_{C}, P_{D} and EA are presented in the main loop of Fig. 2. Specifically,

- P_{C}, solves the original problem $\boldsymbol{f}_o$, is used to maintain the quality and feasibility of the solutions;
- P_{D}, solves a helper problem $\boldsymbol{f}_h$ that considers less constrains, is used to assist P_{C} in exploring the search space and jumping over the local optima;
- EA, stores all elite solutions found during evolution, is used to assist P_{C} maintain the potential optima recovery, fine-tune the population and enhance the exploration capability.

As shown in Algorithm 1 and Fig. 2, COEAS starts with randomly generated initial population P_{C}, P_{D} and elite archive EA as well as an initial parameter setting. In the main loop, two parent mating sets $P_{\mathrm{C}}^{'}$ and $P_{\mathrm{D}}^{'}$ are selected from $P_{\mathrm{C}} \cup \mathrm{EA}^{'}$ and P_{D} by mating selection, respectively. Then, we create new individuals from $P_{\mathrm{C}}^{'}$ and $P_{\mathrm{D}}^{'}$ with learning-guided offspring generation. In combination phase, the offspring population would be shared, while the fitness is evaluated by $\boldsymbol{f}_o$ and $\boldsymbol{f}_h$, respectively. Afterwards, the P_{C}, P_{D} and EA are updated by the environmental selection and elite archive strategy. During the updating process, the population would be fine-tuned once stagnation counter δ exceeds the predefined threshold Δ. The evolution would repeat until the maximum number of fitness evaluations is exhausted. Finally, the subset selection strategy is applied to truncate EA and obtain the final output solution set $\boldsymbol{S}$ of size μ.

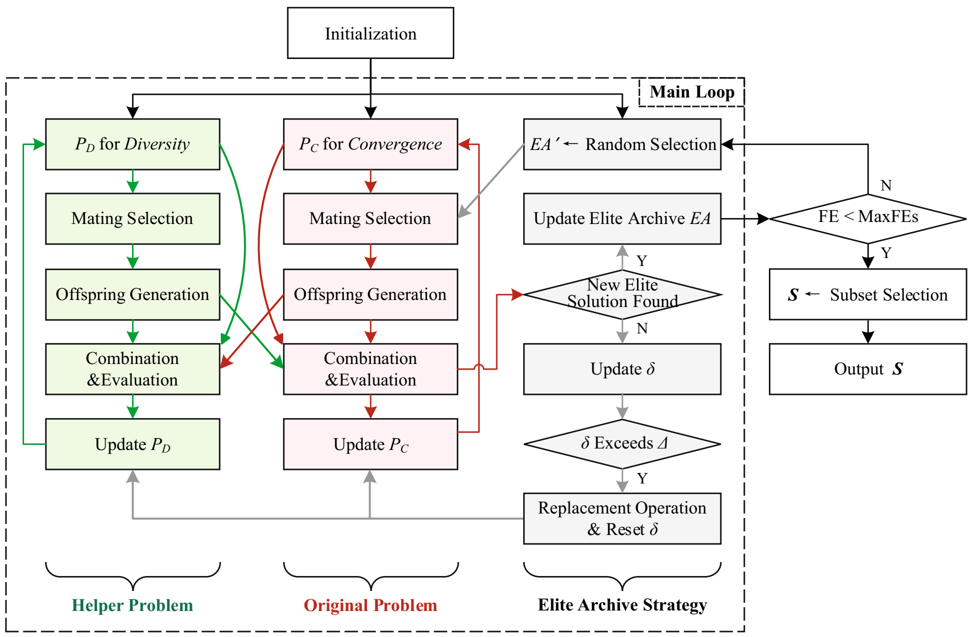

Fig. 2 The flowchart of the proposed algorithm

Algorithm 1 General Framework.

```
Require: Original problem $f_o$, Helper problem $f_h$, Replacement threshold $\Delta$, Population size $N$, Final solution size $\mu$
Ensure: Selected subset $S$
1: /* Initialization */
2: $P_C \Leftarrow$ Random_Initialization($N$,$f_o$);
3: $P_D \Leftarrow$ Random_Initialization($N$,$f_h$);
4: $p\_opt \Leftarrow$ Initialize_PreviousOptimaSet($P_C$);
5: EA $\Leftarrow p\_opt$; $\delta \Leftarrow 0$;
6: while termination criterion not met do
7:     /* Reproduction */
8:     EA$'$ $\Leftarrow$ RandomSelection(EA,$N/2$);
9:     $P_C' \Leftarrow$ MatingSelection($P_C$, $EA'$, $N/2$);
10:    $P_D' \Leftarrow$ MatingSelection($P_D$, $N/2$);
11:    $O_C' \Leftarrow$ LearningGuided_OffspringGeneration($P_C'$, $N/2$);
12:    $O_D' \Leftarrow$ LearningGuided_OffspringGeneration($P_D'$, $N/2$);
13:    $P_C \Leftarrow$ Combination&Evaluation($P_C$, $O_C'$, $O_D'$,$f_o$);
14:    $P_D \Leftarrow$ Combination&Evaluation($P_D$, $O_C'$, $O_D'$,$f_h$);
15:    /* Updation */
16:    [EA, $\delta$] $\Leftarrow$ Update_EliteArchive(EA, $P_C$, $p\_opt$, $\mu$)
17:    $P_C \Leftarrow$ Update_ConvergencePopulation($P_C$, $p\_opt$, $N$);
18:    $P_D \Leftarrow$ Update_DiversityPopulation($P_D$, $p\_opt$, $N$);
19:    [$P_C$, $\delta$] $\Leftarrow$ ReplacementOperation($P_C$, $EA$, $\delta$, $\Delta$);
20:    $p\_opt \Leftarrow$ Update_PreviousOptimaSet($p\_opt$, $EA$);
21: end while
22: $S \Leftarrow$ SubsetSelection(EA, $\mu$);
```

Cooperative co-evolutionary mechanism

Individual encoding

For an input request r, the schedule should decide: whether to support r, the corresponding time window resource and execution start time. In addition, the frequent constraint violation detect would lead to heavy computational burden.

In this paper, each individual x is encoded by the following integer array:

$$x = \{x_1, x_2, ..., x_r, ..., x_{N_R}\} \tag{8}$$

$$x_r = \begin{cases} k & \arg\left(x_{r,a}^k = 1 \wedge \sum\limits_{k\in K} x_{r,a}^k = 1\right) \\ 0 & \sum\limits_{k\in K} x_{r,a}^k = 0 \end{cases} \tag{9}$$

Each request r occupies one gene of the individual, and the length of x is thus equal to the size of input requests. The gene value k represents assigning the kth VTW to request r, which is selected from 0 to K. If $x_r = 0$, request r would not be supported. Fig 3 gives a simple example of individual encoding. Suppose the antenna set has 3 visible time windows to support request 2, hence its variation upper bound is 3. Moreover, the gene value 3 for request 2 means $vtw_{2,4}^3$ is selected, and the resource is provided by antenna 4. The request satisfaction status and time window allocation are encoded in each individual, while the exact start time would be determined in latter decoding procedure. Furthermore, Eq. 9 guarantees each request is assigned with no more than one time window. In this way, constraint 3 and 4 can be naturally satisfied, hence the computation burden is reduced.

Mating selection

The difference between the mating selection of P_C and P_D lies in the union of parent population. To generate high-quality offspring, we fill the mating pool P_C' and P_D' by selecting individuals from $P_C \cup \mathrm{EA}'$ and P_D, respectively. For better understanding, we formulate the input population

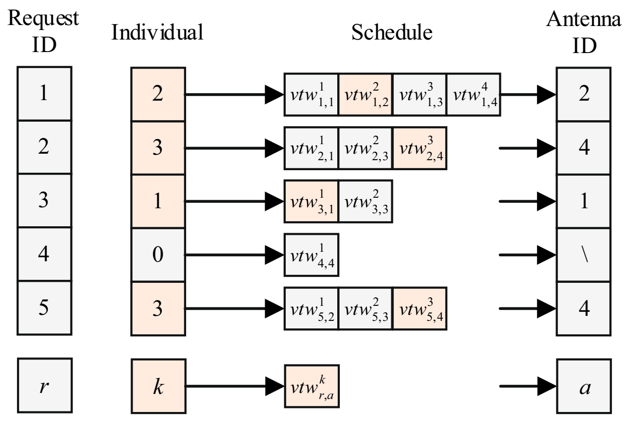
Request ID
Individual
Schedule
Antenna ID
1 2 3 4 5 r
2 3 1 0 3 k
$vtw^1_{1,1}$ $vtw^2_{1,2}$ $vtw^3_{1,3}$ $vtw^4_{1,4}$
$vtw^1_{2,1}$ $vtw^2_{2,3}$ $vtw^3_{2,4}$
$vtw^1_{3,1}$ $vtw^2_{3,3}$
$vtw^1_{4,4}$
$vtw^1_{5,2}$ $vtw^2_{5,3}$ $vtw^3_{5,4}$
$vtw^k_{r,a}$
2 4 1 \ 4 a

Fig. 3 Simple example for individual encoding

as parent population P. Suppose solution y is the most similar solution to x, if y outperforms x in objective space, the closer distance in decision space would lead to higher possibility to select y as the mating parent.

The gene value of each request represents the serial number of assigned time window. Accordingly, the similarity between individuals is related to the number of sharing gene sites, rather than the specific gene value. The distance between solution x and y is calculated by:

$$D(x,\ y) = \frac{|x \cap y|}{N_R}, \tag{10}$$

where $x \cap y$ means the intersection set of decision variables; N_R is the dimension of decision variables. The smaller metric value indicates greater similarity.

The main steps for mating selection are illustrated in Algorithm 2. First, the algorithm randomly select a solution x and find the solution $y \in P$ that is closest to x. Then, the parameter value θ is computed by:

$$\theta(x, y) = \frac{D(x, y) - \min(\min D)}{\max(\min D) - \min(\min D)}, \tag{11}$$

Algorithm 2 Mating Selection

```
Require: Parent Population $P$, Population size $N$
Ensure: Mating Population $P^{'}$
1: $P^{'} \Leftarrow \emptyset$;
2: while $\|P^{'}\| < N/2$ do
3:     Randomly select a solution $x$ from $P$
4:     Find the solution $y$ closest to $x$, $y \in P$
5:     if $rand < \theta \ \vee \ f(y) < f(x)$ then
6:         $P^{'} \Leftarrow P^{'} \cup y$
7:     else
8:         $P^{'} \Leftarrow P^{'} \cup x$
9:     end if
10: end whilereturn $P^{'}$;
```

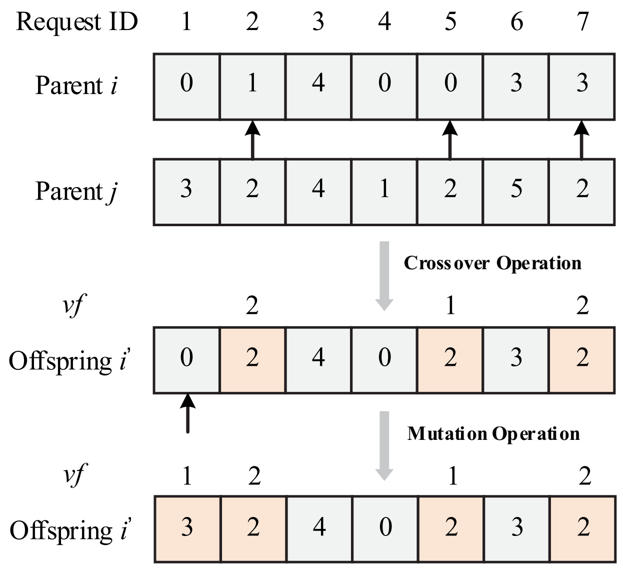

Fig. 4 The crossover and mutation operation

where $\min D = \min D_1, \min D_2, \ldots, \min D_r, \ldots, \min D_{N_R}$ denotes the minimum pairwise distance of each solution. For each individual $p \in P \wedge p \neq r$, $minD_r = minD(r, p)$.

Next, if $rand < \theta \wedge f(y) < f(x)$ is satisfied, we add y instead of x to $P^{'}$. The above procedures are repeated until the mating pool is filled with $N/2$ solutions.

Offspring generation

Taking evolutionary algorithm as the baseline, the offspring generation of the proposed algorithm consists of crossover, mutation, combination and fitness evaluation. However, the randomly generated offspring often yield poor schedules in context of satellite range scheduling.

To generate diverse high-quality solutions with more efficiency, we utilize the request execution knowledge learnt from parent solution to guide the reproduction procedure. During the reproduction, the learning guidance works in three aspects: the variation probability, the variation flag and the decoding sequence. The details of the learning-guided offspring generation are given below:

(1) $Crossover$: The multi-point crossover is adapted to randomly exchange the genes of selected sites. Given two parent solutions as shown in Fig. 4, the requests that failed to be satisfied in parent solution (e.g., request 1, 4, 5 in parent i) would be assigned a higher crossover probability of $pc^{'}$ = min $(2pc$,0.95). The upper bound of $pc^{'}$ is set to be 0.95 rather than 1, intending to avoid greedy search and premature convergence. Based on the modified crossover probability, multiple genes would be randomly selected and moved from parent j to parent i.

To further enhance the solution diversity, we designed a variation flag to detect the gene value changes. For gene r, the variation flag vf_r is determined by the variation information in offspring solution and the execution information in parent solution:

$$vf_r = \begin{cases} 1 & vx_{i',r} = 1 \land x_{i,r} = 0 \\ 2 & vx_{i',r} = 1 \land x_{i,r} \neq 0 \\ 3 & vx_{i',r} = 0, \end{cases} \qquad (12)$$

where $x_{i,r} = 0$ denotes request r is not scheduled in parent solution i;$vx_{i',r}$ is a binary variable for identifying whether gene r is variated in offspring solution i'. And the default variation flag for each gene site is 3. For the variated requests, $vf = 1$ if were not scheduled in parent solution, or $vf = 2$ otherwise.

(2) $Mutation$: As shown in Fig. 4, the mutation operator randomly mutates multiple genes from the offspring, then determines the vf for each gene site. Unlike the crossover operator, the mutation probability is pm/N_{req}, which has a lower occurrence probability.

(3) $Combination\ and\ Fitness\ Evaluation$: First, the current parent population P_{C}' and P_{D}' is combined with offspring populations $O_{\mathrm{C}}' \cup O_{D}'$. Then, the fitness of $O_{C}' \cup O_{D}'$ in updated P_{C} and P_{D} is evaluated by $\boldsymbol{f}_o$ and $\boldsymbol{f}_h$, respectively. Based on Equation 2, the fitness of the population for $\boldsymbol{f}_h$ can be easily obtained. However, the initial time window resource allocation encoded in variated operation could be infeasible in $\boldsymbol{f}_o$. Moreover, multiple individual encodings may be mapped to the same schedule by the decoding strategy. Out of this concern, we design a tri-level decoding strategy to determine the final schedule while maintaining the feasibility, quality and diversity.

During the tri-level decoding, the genes with different vf are decoded in the order of vf = 1–3. To be specific:

(1) *First level decoding*: The $vf = 1$ means the request is variated in offspring solution but is not served in parent generation. Decoding these requests in first level can allocate resources to the selected failed requests with priority, which would improve the schedule quality.

(2) *Second level decoding*: We perform second level decoding for the requests with $vf = 2$. The definition of vf indicates that the resources allocated to these requests are changed, while these requests are successfully served in parent generation.

(3) *Third level decoding*: With regard to the requests with $vf = 3$, the genes corresponding to these requests remain unchanged. The decoding process is time consuming. To accelerate the decoding, we detect the conflicts between the original time window arrangement in parent schedule and current resource allocation. If the constraint is not violated, the original time window arrangement of genes with $vf = 3$ would be directly copied.

After deciding the decoding sequence by the vf, the request execution start time est needs to be selected within the encoded time window. Therefore, the time window allocation strategy is embedded to construct the corresponding schedule. For an input request r, the position is randomly initialized the at the top, middle and bottom of assigned time window. If the request is executable under time constraints, we determine the execution start time est, end time edt and update the resource occupied time $ocupt$; else we remove the request to unscheduled request set $usdr$. Then, we reschedule the requests in $usdr$ based on current available time windows and $ocupt$. And the gene value corresponding to successfully reassigned request would be replaced with the new time window number. The reassignment would be iteratively repeated until $usdr$ is empty or no available time window is left. Finally, the feasible schedule is obtained and the fitness can be evaluated.

To summarize, the variation operation improves the solution quality with the guidance of parent knowledge; the tri-level decoding enhances the solution diversity by decoding in diverse sequence; the time window allocation strategy guarantees the feasibility of the final schedule.

For example, in Fig. 5, suppose that the requests served on the same antenna are conflicting. Therefore, the four ground station antennas can support at most four requests. It can be observed that the initial schedule is infeasible, where request 1, 3 and request 2, 5 are conflicting. Then, we decode the solution encoding under different vf. During the i-th level decoding, we decode the genes with $vf = i$. The requests that conflicting with current resource occupation (requests marked in blue) would be stored and reassigned. The genes of successfully reassigned requests would be modified as the new time window number (e.g., r_2 in Fig. 5), while that of the failed requests would be assigned a 0 value (e.g., r_3 in Fig. 5). Finally, different feasible schedules are constructed by decoding under different vf.

Update mechanism of the P_C

To retain randomness while keeping selection pressure towards the optima, the quality $q(x)$ of solution x is defined by:

$$q(x) = \begin{cases} f(x) & f(x) \geq f(p\_opt) \\ \min\limits_{y \in P_C} (f(y))/2 & f(x) < f(p\_opt), \end{cases} \qquad (13)$$

where p_opt is the optimal solution in previous generation of P_C. Then the roulette selection is used to update P_C, in

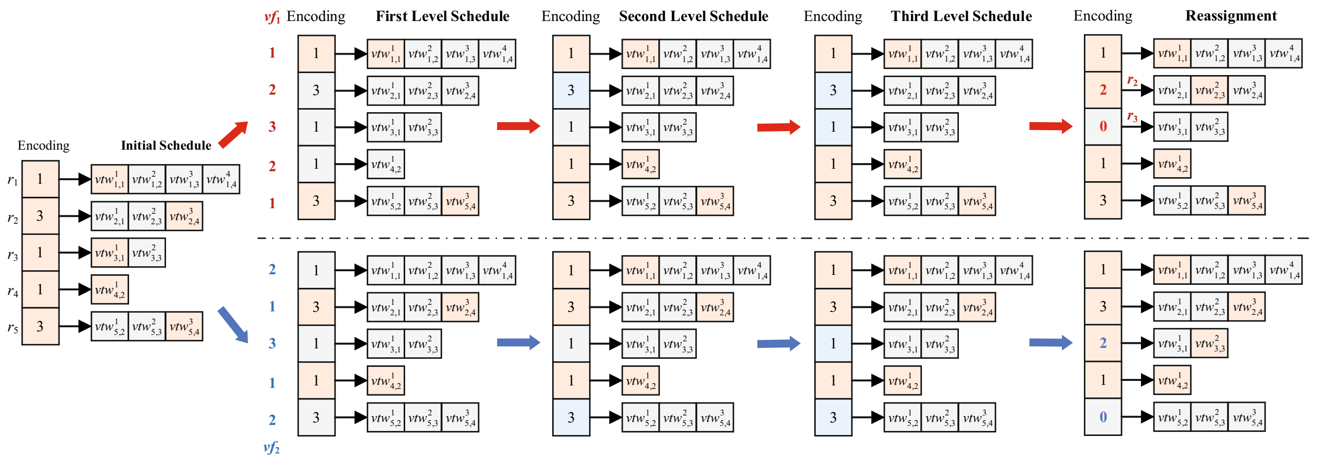

Fig. 5 The decoding of a solution under different variation flags

which the possibility $p(x)$ of selecting solution x is:

$$p(x) = q(x)/\sum_{y \in P_C} q(y). \tag{14}$$

Update mechanism of the P_D

Different from the P_C, population P_D solves the easier problem and aims at generating diversified solutions. Considering less constraints indicates P_D usually has better fitness than P_C, which can possibly improve the performance of P_C in objective space. Moreover, COEAS adapts the weak cooperation framework proposed in [22], in which the populations only interact in the combination phase. Sharing the offspring in combination operation can assist P_C in crossing the infeasible regions and help P_D search towards the promising region.

Algorithm 3 Update Mechanism of the P_D

```
Require: Combined population $P_D$, Population size $N$, Previous optima set $p\_opt$
Ensure: Updated population $P_D^{'}$
1: $[P_D^1, P_D^2] \Leftarrow$ Population_Split($P_D$);
2: if $|P_D^1| \leq N$ then
3:     $P_D^{2*} \Leftarrow$ Roulette_Selection($P_D^2, N - |P_D^1|$);
4:     $P_D^{'} \Leftarrow P_D^1 \cup P_D^{2*}$;
5: else
6:     $P_D^{'} \Leftarrow$ OnebyOne_Elimination($P_D^1, N$)
7: end if
8: return $P_D^{'}$;
```

The details of updating P_D are presented in Algorithm 3. First, we split the population based on previous optimal solution:

$$P_D = \begin{cases} P_D^1 & f\left(P_D^1\right) \leq f\left(p\_opt\right) \\ P_D^2 & f\left(P_D^2\right) > f\left(p\_opt\right) \end{cases} \tag{15}$$

Then, population P_D is updated based on the following two cases:

(1) If $|P_D^1| < N$, meaning that P_D is worse converged. The P_D^1 is directly added to next generation. Based on the individual fitness, the algorithm selects $N - |P_D^1|$ solutions from P_D^2 by roulette. And the selected solutions are combined with P_D^1 to form the new P_D.
(2) If $|P_D^1| \geq N$, implying that P_D is better converged. The redundant solution of current P_D^1 is identified by the distance to N_k nearest solutions in decision space:

$$x^r = \operatorname*{argmin}_{y \in P_D} \left(\sum_{y=1}^{N_k} \min D(x, y) \right) \tag{16}$$

where $N_k = \sqrt{2N}$. Then we eliminate x^r and repeat the above procedure until the size of P_D^1 is decreased to N. In other words, only diverse and high-quality individuals can survive to next generation.

Elite archive strategy

Update mechanism of the EA

The elite archive is in essence a memory of the promising position that the algorithm has reached in previous generations. As illustrated in Algorithm 4, the EA is updated by the combined P_C instead of updated P_C to avoid the effect of genetic drift.

To be specific, we compare the EA with P_C to check whether the new elite solutions are generated. A new solution x means $x \notin EA \wedge f(x) \leq f(p\_opt)$. If the new elite solutions are not detected, we consider the population evolution is stagnated, and add 1 to the stagnation counter δ. If the new

Algorithm 4 Update Mechanism of the EA

```
Require: Elite archive EA, Combined $P_C$, Stagnation counter $\delta$, Minimal archive size $\mu$, Previous optima set $p\_opt$
Ensure: Updated archive EA
1: $[sf, c\_opt] \Leftarrow$ Stagnation_Detection(EA, $P_C$)
2: if $sf = 0$ then
3:     /* population evolution stagnation detected */
4:     $\delta \Leftarrow \delta + 1$
5:     break
6: else
7:     /* new elite solution detected */
8:     $EA \Leftarrow EA \cup c\_opt$
9:     if $|EA| > \mu \wedge \overline{f(EA)} > f(c\_opt)$ then
10:        /* worse $h\_opt$ detected in $EA$ when $|EA|$ exceeds $\mu$ */
11:        $EA \Leftarrow$ Truncation(EA, $c\_opt$, $\mu$)
12:    end if
13: end if
14: return $EA$
```

Algorithm 5 Replacement Operation

```
Require: Elite archive $EA$, $P_C$, $P_D$, Stagnation counter $\delta$
Ensure: Updated $P_C$, $P_D$, $\delta$
1: if $\delta > \Delta$ then
2:     $[x_c, x_d] \Leftarrow$ Identification($P_C$, $P_D$)
3:     $ite \Leftarrow 0$
4:     while $ite < 5$ do
5:         $ite \Leftarrow ite + 1$
6:         $P_e \Leftarrow$ Randomly select $N$ solutions from EA
7:         $P_{en} \Leftarrow$ Neighbor_search(EA)
8:         $[x_{cn}, x_{dn}] \Leftarrow$ Identification($P_{en}$, $P_D$)
9:         if $f(x_{cn}) < f(x_c) \vee rand < \xi$ then
10:            $[P_C, P_D] \Leftarrow$ Replacement($P_C$, $P_D$, $x_c$, $x_d$, $x_{cn}$, $x_{dn}$)
11:            $\delta \Leftarrow 0$
12:            break
13:        end if
14:    end while
15: end if
16: return $P_C$, $P_D$, $\delta$
```

elite solutions are generated, we combine the current optima set c_opt with EA.

Moreover, we define the minimal archive size μ to help the algorithm explore the search space with different behavior. At the early iterations, the elite solutions are directly added to the archive until the archive size reaches to μ. These widely distributed historical optimal solutions h_opt enables the algorithm to better explore the search space. Then we gradually eliminate the worst h_opt through truncation. In the late iterations, the archive is filled with multiple local optima. Restarting the search from these individuals can help the algorithm locate better optimal solutions.

Replacement operation

The population individuals tend to converge to the early-found local optima during the evolution, which eventually leads to the premature convergence and diversity lost. Out of this concern, we detect and count the population stagnation in previous update procedure. Once the stagnation counter δ exceeds the predefined threshold Δ, we replace the worst/crowded solution in P_C/P_D with neighbor solution of EA to fine-tune the population.

The procedures of replacement operation are described in Algorithm 5. First, the algorithm randomly selects N elite solutions from EA, and finds the solution x_c with worst fitness in P_C, the most crowded solution x_d in P_D. Next, the neighbor region of the selected solutions is exploited through mutation operation. The neighbor solution with best fitness and maximum distance to P_D is denoted by x_{cn} and x_{dn}, respectively. In the following, we compare the fitness of x_{cn} and x_c. If the fitness of x_{cn} is better than x_c or a randomly generated number is less than a threshold ξ, we reset δ and replace the x_c and x_d with x_{cn} and x_{dn}, respectively. Otherwise, the above procedures would be repeated for five times until the replacement is conducted. The threshold ξ is computed by

$$\xi = \exp\left(-\frac{f(x_{cn}) - f(x_c)}{f(x_{cn}) - \min(f(P_C))}\right). \tag{17}$$

Moreover, the ability of revisiting parts of the search space can yield a better balance between seeking greedily for new solutions and seeking for high-quality solutions locally [27]. In the proposed algorithm, the elite archive individuals are allowed to participate in the mating selection of P_C for the purpose of maintaining the potential optima recovery.

Experimental analysis

Experimental instances

Based on actual engineering, the generated experimental instances are involved with both LEO, MEO, GEO satellites and globally distributed ground station. Part of the ground stations is equipped with multiple antennas to support the requests, which can increase the possibility of finding multiple optimal schedules. The size of input requests ranges from 150 to 350 and the scheduling horizon is set to be 24 h.

Contrast algorithms

Three algorithms are selected for performance comparison: (a) CCMO [22]: a coevolutionary constrained multi-objective optimization algorithm, which first proposed the weak-cooperation framework; (b) GA-PE [28]: a genetic algorithm with population perturbation and elimination strategy for SRSP; (c) GA [29]: a classic evolutionary algorithm, which is adapted as the performance baseline.

For fair comparison, each algorithm adapts the same default parameter settings as shown in Table 1. The algorithm

Table 1 The default algorithm parameter settings

Parameter	Denotion	Value
Instance MaxFEs	150–200	12,000
	225–275	16,000
	300–350	20,000
Population size	p_z	16
Crossover probability	p_c	0.9
Mutation probability	p_m	0.05
Output archive size	μ	50

termination criterion is the maximum fitness evaluations (MaxFEs). Moreover, the elite archive is incorporated to the comparison algorithms for the purpose of preserving the historical optimal solutions. Limited to the search ability, the final archive size for each algorithm in different runs varies. Hence, we modify the output of each algorithm as μ elite solutions from EA, instead of the final population. And the representative μ elite solutions are selected by the truncation operation in elite archive strategy.

Performance indicators

Three evaluation indicators are adapted to measure the quality and diversity of obtained solutions:

(1) F_β *Measure*

F_β Measure, derives from information retrieval and pattern recognition [30], is a comprehensive indicator to evaluate the solution quality. For a ground-truth solution set $\boldsymbol{G}$, the F_β measure of solution set $\boldsymbol{S}$ is calculated as

$$F_\beta = \frac{\left(1+\beta^2\right)\cdot prec \cdot rec}{\beta^2 \cdot prec + rec}, \tag{18}$$

where β^2 is a parameter to control the effect of precision value $prec$ and the recall value rec on the metric value. In this paper, the β^2 is set to 0.3 as advised in [31]. The $prec$ shows the fraction of optimal solutions in $\boldsymbol{S}$:

$$prec = \frac{\text{TP}}{\text{TP}+\text{FP}}, \tag{19}$$

where TP and FP are the number of optimal and non-optimal solutions in $\boldsymbol{S}$. And the rec denotes the fraction of optimal solutions the algorithm found in $\boldsymbol{G}$:

$$rec = \frac{\text{TP}}{\text{TP}+\text{FN}}, \tag{20}$$

where FN are the number of optimal solutions the algorithm failed to locate in $\boldsymbol{G}$.

Particularly, the F_β score equals to 1/0 when all/none of the solutions in $\boldsymbol{S}$ are optimal.

(2) *Diversity Indicator*

The F_β scores for the algorithms, which fail to locate any problem optima, are 0. To further distinguish the algorithm performance, the diversity indicator (DI) is adapted to measure the average maximum similarity between $\boldsymbol{S}$ and $\boldsymbol{G}$ in decision space:

$$\text{DI}(\boldsymbol{S},\boldsymbol{G}) = \frac{\sum_{x\in\boldsymbol{S}} \max D(x,y)|_{y\in\boldsymbol{G}}}{|\boldsymbol{S}|}. \tag{21}$$

The DI describes the convergence degree of the obtained solutions in decision space. And the greater metric value indicates the better performance.

The above two indicators give a comprehensive comparison on finding multiple optimal solutions, and more detailed descriptions can be found in [18,31]. During the performance evaluation, the final optimal solution set that all the algorithms have obtained for each instance is adapted as an approximation to the ground-truth solution set $\boldsymbol{G}$.

(3) *Request Failure Rate*

The request failure rate (RFR) evaluates the output solution set $\boldsymbol{S}$ in terms of satisfying the input requests:

$$\text{RFR}(\boldsymbol{S}) = \frac{\sum\limits_{x\in\boldsymbol{S}} f(x)}{|\boldsymbol{S}|}. \tag{22}$$

The RFR shows the convergence degree of obtained solutions in objective space (i.e., the capability to optimize the SRSP).

Exploring the proposed method

To analyze the significance of the proposed elite archive strategy, learning-guided offspring generation and mating selection in COEAS, we compare the COEAS with its five variants: COEAS with fixed elite archive size (COEAS-f); COEAS that adapts global optima as the population update reference point (COEAS-g); COEAS that randomly generates the offspring (COEAS-r); COEAS without replacement operation (COEAS-nr) and COEAS without parent mating selection (COEAS-nm). Table 2 presents the mean metric value comparisons from 20 runs for each algorithm. For each instance, the best and second-best result is highlighted in bold and italicized, respectively. The Wilcoxon rank-sum test is applied to further investigate the effect at significant level $\alpha = 0.05$, where symbols '+', '−', and '≈' indicates that the algorithm performance is significantly better than, significantly worse than, or similar to that of COEAS.

As seen in the table, the proposed strategies can effectively enhance the overall performance of COEAS. Specifically, the

Table 2 Average metric score in terms of F_β, DI and RFR on each instance over 20 runs

Inst.	Metric	COEAS	COEAS-f	COEAS-g	COEAS-r	COEAS-nr	COEAS-nm
	F_β	**0.075**	0.056	*0.069*	0	0.063	0.005
150	DI	**0.926**	0.837	*0.870*	0.443	0.865	0.590
	RFR	**0.061**	0.074	*0.064*	0.091	0.068	0.078
	F_β	**0.076**	0.059	*0.061*	0	0.049	0.002
175	DI	**0.935**	*0.852*	0.816	0.439	0.798	0.547
	RFR	**0.053**	0.062	*0.054*	0.076	0.061	0.076
	F_β	**0.109**	0.033	*0.077*	0	0.029	0.003
200	DI	**0.976**	0.659	*0.754*	0.410	0.638	0.473
	RFR	**0.047**	0.065	*0.054*	0.070	0.064	0.063
	F_β	**0.065**	0.061	*0.062*	0	0.046	0.010
225	DI	**0.873**	*0.866*	0.820	0.440	0.793	0.715
	RFR	**0.042**	0.049	*0.044*	0.065	0.054	0.052
	F_β	**0.096**	*0.062*	0.029	0	0.012	0
250	DI	**0.750**	*0.678*	0.570	0.451	0.536	0.491
	RFR	**0.060**	0.072	*0.066*	0.091	0.078	0.074
	F_β	**0.067**	0.053	*0.061*	0	0.058	0.004
275	DI	**0.939**	0.883	0.877	0.421	*0.902*	0.57
	RFR	**0.040**	0.044	*0.041*	0.064	0.043	0.053
	F_β	**0.094**	0.046	0.042	0	*0.050*	0.007
300	DI	**0.936**	0.745	0.681	0.422	*0.758*	0.554
	RFR	**0.048**	0.062	*0.054*	0.073	0.062	0.060
	F_β	**0.086**	0.044	*0.047*	0	0.011	0.018
325	DI	**0.683**	*0.545*	0.525	0.397	0.520	0.476
	RFR	**0.095**	0.112	*0.100*	0.129	0.114	0.109
	F_β	**0.102**	0.057	*0.061*	0	0.032	0
350	DI	**0.682**	*0.615*	0.572	0.424	0.532	0.449
	RFR	**0.134**	0.147	*0.136*	0.164	0.149	0.141
	F_β	8/1/0	8/1/0	6/3/0	0/6/3	6/3/0	\
+/−/≈	DI	9/0/0	7/2/0	7/2/0	0/0/9	5/4/0	\
	RFR	9/0/0	2/7/0	8/1/0	0/1/8	2/6/1	\

The proposed method is compared with its five variations

COEAS-f and COEAS-g achieve most of the second-best performance. Without learning-guided offspring generation, the COEAS-r fails to locate the optima in given iterations. Due to the limitation of multi-constrained and combinatorial characteristics of SRSP, the initial randomly generated solutions could hardly maintain quality and feasibility, simultaneously. Without replacement operation, COEAS-nr obtains the second-best performance on instance 275 and 300, and performs badly with respect to the RFR value. Consequently, by exploiting the neighbor region of elite archive candidates, the replacement operation can effectively improve the population quality. COEAS-nm cancels the mating selection of parent solutions and directly reproduces new solutions from P_C and P_D, respectively. In terms of F_β and DI, the COEAS-nm shows worse performance, which indicates the worse ability to explore the search space. This result is due to the potential optima recovered from historical optima in EA, and the competition between the similar solutions.

To present a further investigation of the performance concerning different indicators, the mean metric value for all algorithm variations on instance 225 is shown in Fig. 6. At early stage of the evolution, COEA-f and COEAS-g converge faster than COEAS. As the evolution progresses, COEAS outperforms the variations, which shows that the assistance of P_D and EA can help P_C better explore the search space and facilitate the population evolution. As for COEAS-nm, it is outperformed by all the peer variations except COEAS-r. Its failure reflects that the interactions between P_C and EA in parent mating can effectively avert being trapped in local optima.

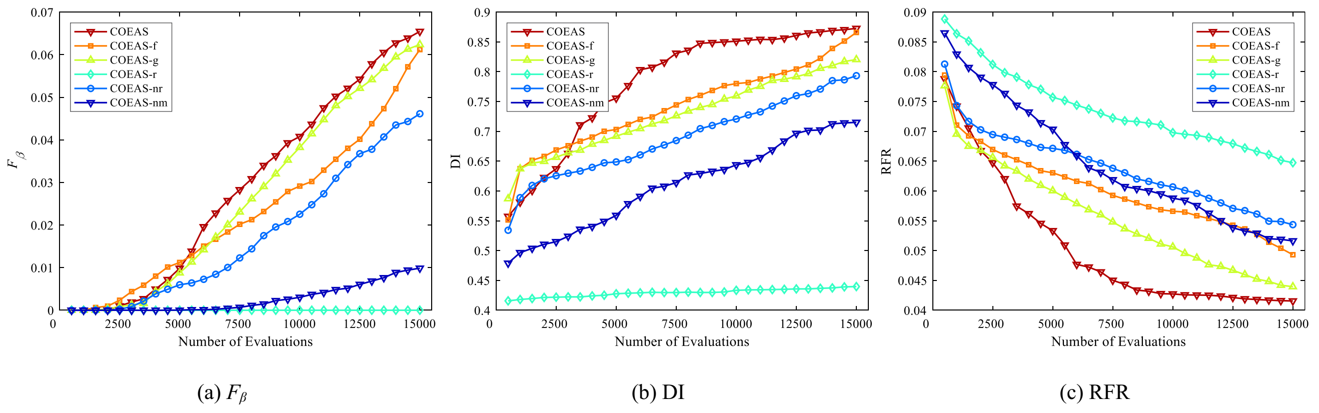

Fig. 6 Mean metric value for different algorithm variations during the evolution

Comparing with the related algorithm

Then, we compare COEAS with recently proposed methods including coevolutionary algorithm that adapts weak cooperation framework (CCMO) [22], population perturbation and elimination strategy based genetic algorithm (GA-PE) [28] and classical genetic algorithm (GA) [29]. As analyzed in previous section, the randomly generated offspring in original evolutionary algorithm has difficulty to solve SRSP with efficiency. To make a meaningful comparison, the proposed offspring generation approach is implemented in all algorithms on each instance for efficiency enhancement. Table 3 provides the comparison results of each algorithm. The original and modified version of the algorithm is denoted by ‘random’ and ‘/’, respectively. The best result for each instance and the better result within each pair of algorithms is highlighted with bold and italicized, respectively.

It can be observed that COEAS processes the superior performance over compared algorithms in terms of F_β, DI and RFR. Specifically,

(1) The F_β score for the modified version of each algorithm significantly outperforms that of the original algorithm, which validates that better solving efficiency can be achieved with the guidance of heuristic information learnt from parent schedule. For each instance. COEAS reaches the highest F_β value and provides most of the optimal solutions. CCMO achieves the second-best results in most instances. The improved GA-PE and GA can find a limited number of optimal schedules, but still perform poorly on instances with larger scale.

(2) Higher DI indicates the better solution diversity. COEAS can maintain the best diversity among the algorithms. GA-PE and GA show similar performance on providing diverse solutions, while CCMO presents better performance by providing more distinct optima.

(3) Request failure rate reflects the ability of the algorithm to tackle SRSP. We present a quick view of RFR score by plotting the average metric value for different instances in Fig. 7. The corresponding original version is marked with ‘r’. It can be observed that the final output solutions of COEAS have a good convergence to the optimum of each instance. And all the algorithms are encountered with difficulty to satisfy more requests for the instances with a large size.

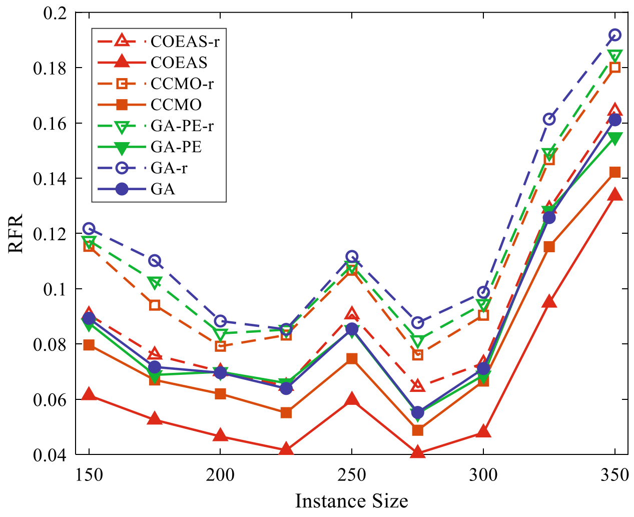

Fig. 7 The average RFR values of the solutions obtained by COEAS, CCMO, GA-PE and GA

To present an intuitive algorithm performance comparison concerning F_β, DI and RFR, we adapt the performance indicator $p(Alg_i)$. For algorithm Alg_i, $p(Alg_i)$ reflects the number of algorithms that are significantly better than Alg_i. The lower indicator value means better algorithm capability. More detailed description of Alg_i can be found in [32]. It can be observed that COEAS and CCMO turn to have the best performance. The basic GA-PE outperforms basic GA, while the overall performances are closed with the introduction of heuristic information.

Figure 8 compares the quality, diversity and convergence of elite archive solutions on instance 225. It can be seen

Table 3 Average metric score in terms of F_β, DI and RFR on each instance over 20 runs. The proposed method is compared with its four related algorithms and the corresponding improved version

Inst.	Metric	COEAS		CCMO		GA-PE		GA	
		Random	/	Random	/	Random	/	Random	/
150	F_β	0	***0.181***	0	*0.043*	0	*0.028*	0	*0.041*
	DI	0.426	***0.929***	0.422	*0.612*	0.423	*0.582*	0.407	*0.602*
	RFR	0.091	***0.061***	0.115	*0.080*	0.117	*0.088*	0.122	*0.089*
175	F_β	0	***0.109***	0	*0.053*	0	*0.047*	0	*0.023*
	DI	0.436	***0.933***	0.421	*0.690*	0.410	*0.713*	0.407	*0.613*
	RFR	0.076	***0.053***	0.094	*0.067*	0.103	*0.069*	0.110	*0.072*
200	F_β	0	***0.158***	0	*0.046*	0	*0.020*	0	*0.009*
	DI	0.399	***0.977***	0.400	*0.641*	0.390	*0.508*	0.398	*0.552*
	RFR	0.070	***0.047***	0.079	*0.062*	0.084	*0.070*	0.088	*0.070*
225	F_β	0	***0.165***	0	*0.053*	0	*0.009*	0	*0.011*
	DI	0.422	***0.868***	0.409	*0.642*	0.416	*0.526*	0.405	*0.526*
	RFR	0.065	***0.042***	0.083	*0.055*	0.085	*0.066*	0.085	*0.064*
250	F_β	0	***0.150***	0	*0.021*	0	*0.008*	0	*0.005*
	DI	0.434	***0.744***	0.418	*0.470*	0.428	*0.480*	0.412	*0.470*
	RFR	0.091	***0.060***	0.107	*0.075*	0.108	*0.085*	0.112	*0.085*
275	F_β	0	***0.134***	0	*0.036*	0	*0.019*	0	*0.040*
	DI	0.416	***0.938***	0.403	*0.619*	0.407	*0.556*	0.399	*0.658*
	RFR	0.064	***0.040***	0.076	*0.049*	0.081	*0.055*	0.088	*0.055*
300	F_β	0	***0.180***	0	*0.031*	0	*0.015*	0	0
	DI	0.411	***0.934***	0.396	*0.568*	0.397	*0.509*	0.392	*0.423*
	RFR	0.073	***0.048***	0.090	*0.067*	0.095	*0.069*	0.099	*0.071*
325	F_β	0	***0.183***	0	0	0	0	0	*0.001*
	DI	0.376	***0.671***	0.375	*0.398*	0.372	*0.395*	0.357	*0.408*
	RFR	0.129	***0.095***	0.147	*0.115*	0.149	*0.128*	0.161	*0.126*
350	F_β	0	***0.163***	0	*0.049*	0	0	0	0
	DI	0.409	***0.676***	0.390	*0.489*	0.395	*0.420*	0.383	*0.415*
	RFR	0.164	***0.134***	0.180	*0.142*	0.185	*0.155*	0.192	*0.161*

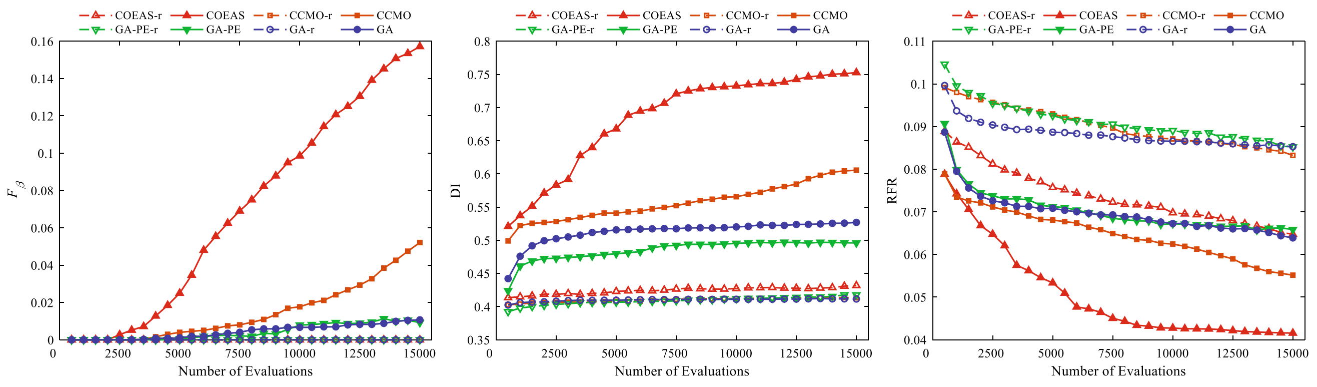

Fig. 8 Mean metric value for different related algorithms during the evolution

that COEAS can identify and preserve the optimal schedules with compared algorithms, COEAS can jump out of the local optimum easily with the assistance of P_D and EA.

Influence of replacement threshold

To investigate the effect of replacement threshold, we conduct the parameter sensitivity verification experiments in this section. Six groups of experiments, corresponding to

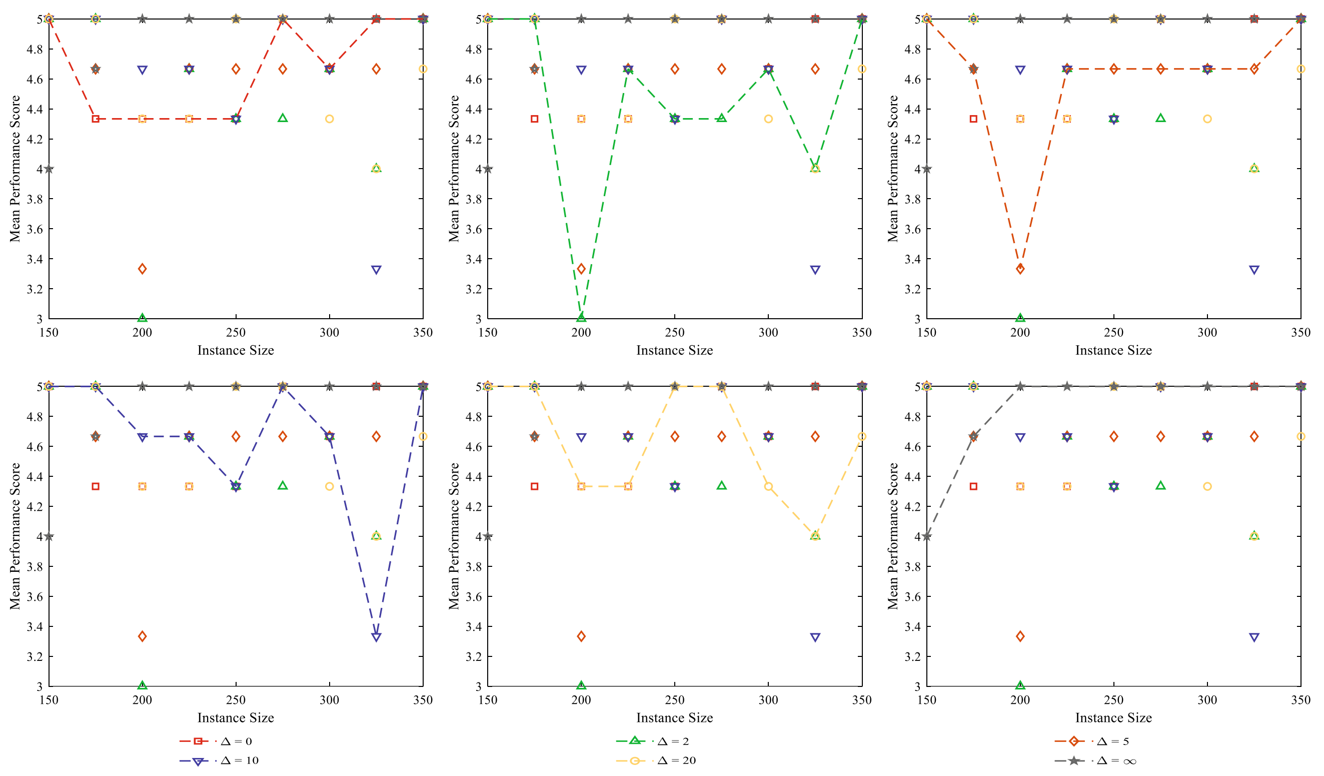

Fig. 9 Mean performance score of different Δ on different instances

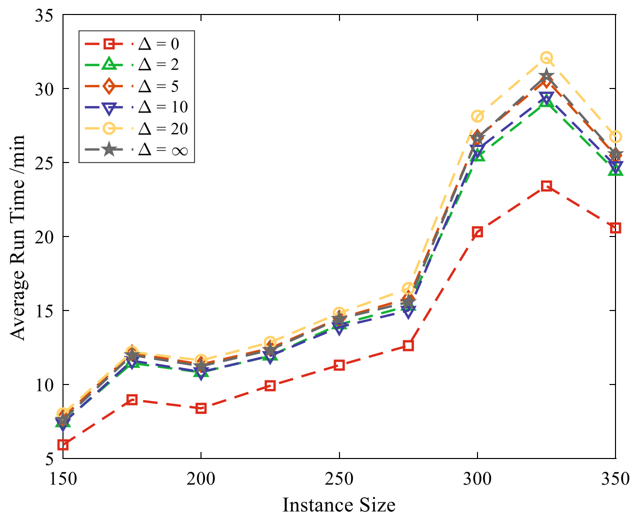

Fig. 10 Mean execution time of different Δ on different instances

replacement threshold 0, 2, 5, 10, 20 and ∞, are carried out. Particularly, the value 0/∞ means replacement operation in each/no generation, respectively.

For the sake of brevity, we present the mean performance score of different Δ in terms of F_β, DI and RFR on different instances in Fig. 9. Compared with $\Delta = \infty$, the proposed replacement operation can help the algorithm to maintain good overall performance in terms of both solution quality and diversity on different, especially large-scale, instances. In regard of $\Delta = 0$, the frequent local search can handle the instances with small size, but have difficulty to tackle the large-scale instances. Δ at other parameter levels show similar performance on most instances.

In Fig. 10, we further compare the algorithm execution time, which rises dramatically along with the increasing input request size. In the replacement operation, the local search would be repeated for five times until a desired solution is found. $\Delta = 0$ allocates most computational resource to find the neighbor solution, hence consumes the least execution time on each instance. In this paper, the Δ is finally assigned to 2 for the purpose of fine-tuning population quality and avoiding unnecessary exploitation.

Effect of obtaining multiple solutions for SRSP

In this section, the effect of providing multiple high-quality solutions for decision makers are investigated.

To describe the preference of decision makers for selecting the final schedule, the following two indicators are adapted:

(1) *Load Balance Rate*: It evaluates the working load balance rate of each ground station antenna.

$$LBR = \frac{\sigma(L_A)}{\bar{L}_A}, \tag{23}$$

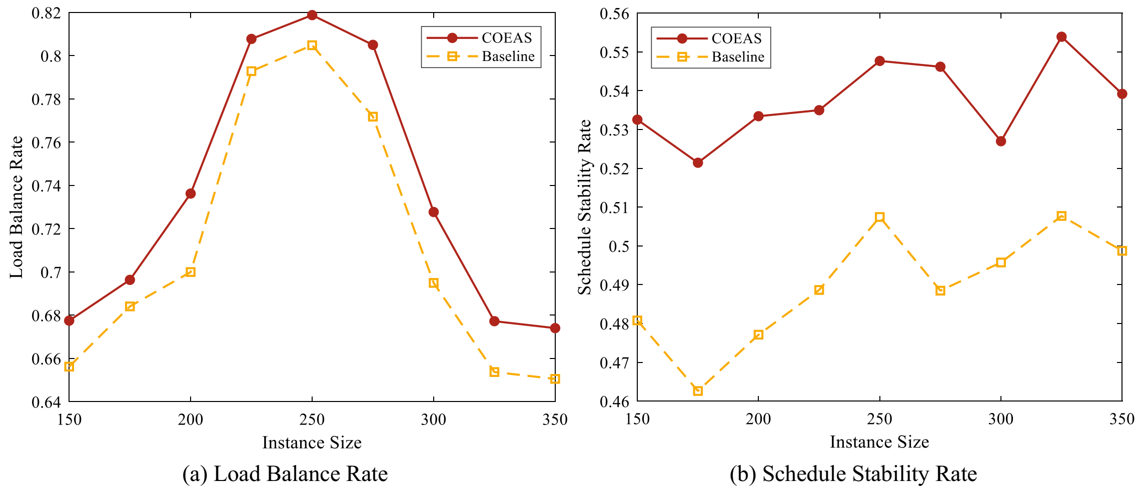

Fig. 11 Performance comparisons between obtaining multiple high-quality solutions and finding single optima

where $\sigma(L_A)$ and $\bar{L}_A$ denotes the standard deviation and average of antenna load. Selecting an optimal schedule can support the input requests with a more rational ground station resource utilization.

(2) *Schedule Stability Rate* : It measures the stability of original schedule to emergency requests.

$$SSR = \frac{|I|}{|I| + \sum_{i \in I} N_i}, \tag{24}$$

where I denotes the set of emergency input requests; N_i means the number of original requests that are affected by request i. The higher schedule stability rate means the ground station resource can tackle the emergencies with faster response.

During the experiment, the single best solution from final population P_C is utilized as the baseline. For SSR analysis, we randomly input emergency requests with 10% of the instance size. Figure 11 provides the comparison results. It can be seen that COEAS possesses the superior performance over traditional method. The diversified high-quality schedules can provide the decision makers with more alternatives to select based on their practical concerns.

Conclusion

This paper develops a co-evolutionary algorithm with elite archive strategy for tackling SRSPs, which can provide a set of diverse high-quality schedules in a single run. Two techniques are effectively employed to improve the performance of the algorithm: (1) cooperative co-evolutionary mechanism; (2) elite archive strategy. The cooperative co-evolutionary mechanism drives two populations explore the problem space during evolution with a weak collaboration. The elite archive strategy helps identify and preserve potential optima, so as to avoid the effect of genetic drift. Moreover, the archive individuals participate in parent mating for enabling the algorithm to revisit the historical promising region. The replacement operation adaptively fine-tunes the population to alleviate the premature convergence and avert unnecessary local search. To further enhance the search ability, the offspring generation approach reinitializes and decodes the new schedules with the guidance of request satisfaction information in parent schedule, so as to improve the quality and diversity of the new schedules.

Based on these techniques, the COEAS shows a promising performance both in reaching better objective value and locating more optimal schedules per run. The experimental results also the flexibility of COEAS for providing more high-quality alternative schedules.

For future works, we will extend the algorithm to tackle more complicated instances, such as instances with larger request scales or considering more objectives. In addition, some effective techniques can be further explored and utilized to solve the problem, such as niching techniques.

Funding This article was funded by Science and Technology on Complex Electronic System Simulation Laboratory (Grant No. 6142401003022109).

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Barbulescu L, Watson J-P, Whitley LD, Howe AE (2004) Scheduling space-ground communications for the air force satellite control network. J Sched 7(1):7–34
2. Vazquez AJ, Erwin RS (2015) On the tractability of satellite range scheduling. Optim Lett 9(2):311–327
3. Marinelli F, Nocella S, Rossi F, Smriglio S (2011) A Lagrangian heuristic for satellite range scheduling with resource constraints. Comput Oper Res 38(11):1572–1583
4. Wang J, Demeulemeester E, Qiu D (2016) A pure proactive scheduling algorithm for multiple earth observation satellites under uncertainties of clouds. Comput Oper Res 74:1–13
5. Vazquez AJ, Erwin RS (2014) Optimal fixed interval satellite range scheduling. In: ICORES, pp 401– 408
6. Liu Z, Feng Z, Ren Z (2019) Route-reduction-based dynamic programming for large-scale satellite range scheduling problem. Eng Optim 51(11):1944–1964
7. Barbulescu L, Watson J-P, Whitley LD, Howe AE (2004) Scheduling space-ground communications for the air force satellite control network. J Sched 7(1):7–34
8. Sarkheyli A, Bagheri A, Ghorbani-Vaghei B, Askari-Moghadam R (2013) Using an effective Tabu search in interactive resources scheduling problem for Leo satellites missions. Aerosp Sci Technol 29(1):287–295
9. Waiming Z, Xiaoxuan H, Wei X, Peng J (2019) A two-phase genetic annealing method for integrated earth observation satellite scheduling problems. Soft Comput 23(1):181–196
10. Luo K, Wang H, Li Y, Li Q (2017) High-performance technique for satellite range scheduling. Comput Oper Res 85:12–21
11. Li Y, Wang R, Liu Y, Xu M (2015) Satellite range scheduling with the priority constraint: an improved genetic algorithm using a station id encoding method. Chin J Aeronaut 28(3):789–803
12. Zhang Z, Hu F, Zhang N (2018) Ant colony algorithm for satellite control resource scheduling problem. Appl Intell 48(10):3295–3305
13. Song Y, Xing L, Wang M, Yi Y, Xiang W, Zhang Z (2020) A knowledge-based evolutionary algorithm for relay satellite system mission scheduling problem. Comput Ind Eng 150:106830
14. Du Y, Xing L, Zhang J, Chen Y, He Y (2019) Moea based memetic algorithms for multi-objective satellite range scheduling problem. Swarm Evol Comput 50:100576
15. Song Y-J, Ma X, Li X-J, Xing L-N, Wang P (2019) Learning-guided nondominated sorting genetic algorithm ii for multi-objective satellite range scheduling problem. Swarm Evol Comput 49:194–205
16. Huang T, Gong Y-J, Zhang Y-H, Zhan Z-H, Zhang J (2019) Automatic planning of multiple itineraries: a niching genetic evolution approach. IEEE Trans Intell Transp Syst 21(10):4225–4240
17. Do AV, Guo M, Neumann A, Neumann F (2022) Niching-based evolutionary diversity optimization for the traveling salesperson problem. arXiv preprint arXiv:2201.10316
18. Huang T, Gong Y-J, Kwong S, Wang H, Zhang J (2019) A niching memetic algorithm for multi-solution traveling salesman problem. IEEE Trans Evol Comput 24(3):508–522
19. Zou P, Rajora M, Liang SY (2021) Multimodal optimization of permutation flow-shop scheduling problems using a clustering-genetic-algorithm-based approach. Appl Sci 11(8):3388
20. Yao X (2006) A new multi-objective evolutionary optimisation algorithm: the two-archive algorithm. In: 2006 International Conference on computational intelligence and security, vol 1:286–291, IEEE
21. Li K, Chen R, Fu G, Yao X (2018) Two-archive evolutionary algorithm for constrained multiobjective optimization. IEEE Trans Evol Comput 23(2):303–315
22. Tian Y, Zhang T, Xiao J, Zhang X, Jin Y (2020) A coevolutionary framework for constrained multiobjective optimization problems. IEEE Trans Evol Comput 25(1):102–116
23. Li Y, Feng X, Yu H (2022) A constrained multiobjective evolutionary algorithm with the two-archive weak cooperation. Inf Sci
24. Liu Y, Yen GG, Gong D (2018) A multimodal multiobjective evolutionary algorithm using two-archive and recombination strategies. IEEE Trans Evol Comput 23(4):660–674
25. Wang Z-J, Zhan Z-H, Lin Y, Yu W-J, Yuan H-Q, Gu T-L, Kwong S, Zhang J (2017) Dual-strategy differential evolution with affinity propagation clustering for multimodal optimization problems. IEEE Trans Evol Comput 22(6):894–908
26. Sheng W, Wang X, Wang Z, Li Q, Chen Y (2021) Adaptive memetic differential evolution with niching competition and supporting archive strategies for multimodal optimization. Inf Sci 573:316–331
27. Gravina D, Liapis A, Yannakakis GN (2018) Quality diversity through surprise. IEEE Trans Evol Comput 23(4):603–616
28. Chen M, Wen J, Song Y-J, Xing L-N, Chen Y-W (2021) A population perturbation and elimination strategy based genetic algorithm for multi-satellite tt&c scheduling problem. Swarm Evol Comput 65:100912
29. Whitley D (1994) A genetic algorithm tutorial. Stat Comput 4(2):65–85
30. Flach P, Kull M (2015) Precision-recall-gain curves: Pr analysis done right. In: Advances in neural information processing systems 28
31. Huang T, Gong Y-J, Zhang J (2018) Seeking multiple solutions of combinatorial optimization problems: a proof of principle study. In: 2018 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1212– 1218. IEEE
32. Bader J, Zitzler E (2011) Hype: an algorithm for fast hypervolume-based many-objective optimization. Evol Comput 19(1):45–76