Hindawi
Discrete Dynamics in Nature and Society
Volume 2017, Article ID 7345941, 15 pages
https://doi.org/10.1155/2017/7345941

Research Article

A Branch and Bound Algorithm for Agile Earth Observation Satellite Scheduling

Xiaogeng Chu, Yuning Chen, and Lining Xing

College of Information Systems and Management, National University of Defense Technology, Changsha, Hunan 410073, China

Correspondence should be addressed to Xiaogeng Chu; chuxiaogeng0702@nudt.edu.cn

Received 15 May 2017; Revised 25 July 2017; Accepted 31 July 2017; Published 7 September 2017

Academic Editor: Filippo Cacace

The agile earth observing satellite scheduling (AEOSS) problem consists of scheduling a subset of images among a set of candidates that satisfy imperative constraints and maximize a gain function. In this paper, we consider a new AEOSS model which integrates a time-dependent temporal constraint. To solve this problem, we propose a highly efficient branch and bound algorithm whose effective ingredients include a look-ahead construction method (for generating a high quality initial lower bound) and a combined use of three pruning strategies (which help to prune a large portion of the search space). We conducted computational experiments on a set of test data that were generated with information from real-life scenarios. The results showed that the proposed algorithm is efficient enough for engineering application. In particular, it is able to solve instances with 55 targets to optimality within 164 seconds on average. Furthermore, we carried out additional experiments to analyze the contribution of each key algorithm ingredient.

1. Introduction

Agile Earth observing satellites (AEOSs) have a stronger ability to collect images of specified areas of the Earth's surface. And AEOSs play key roles in application such as environment surveillance, reconnaissance, resource investigation, and other fields. However, satellite resources are still scarce with respect to the increasing demands for imaging. Whereas nonagile satellites have only one degree of freedom on roll axis for acquiring images, the agile satellites have mobility on the three axes (roll, pitch, and yaw), giving opportunities for a more efficient use of the satellite imaging capabilities. As a result, the scheduling of the agile Earth observing satellites, which is a complex combinatorial optimization problem that has been proved to be NP-hard [1], is a key issue in the management of satellites. The scheduling problem of the AEOSs was studied by a number of researchers.

It is believed that [1] was the first work introducing AEOS. They provided a comprehensive description of AEOS and analyzed four different methods. They studied the increased observation freedom provided by the agility and the complicated transition times, modeled a simplified version of the complete problem, and drew comparisons with four algorithms. The authors in [2] describe a synthesis of permutation-based search and constraint propagation for AEOS scheduling which incorporates the advantages of both techniques. They obtained the large neighborhood behavior of permutation search for oversubscribed resource scheduling problems and exploited the power of constraint propagation to retain as much flexibility as possible while building the schedule. Bianchessi [3, 4] considered the observation scheduling problem for a constellation of agile satellites. A tabu search algorithm was devised to produce solutions whose qualities were evaluated by a column generation algorithm on a linear programming relaxation of the problem. Habet et al. [5–7] proposed a tabu search based on the saturated and consistent neighborhood to solve the AEOS problem involving stereoscopic and time window constraints. The authors used a formulation of constrained optimization problems and a convex evaluation function. Moreover, the upper bounds were computed by relaxing the constraints and linearization of the objective function. Florio [8] developed a heuristic with look-ahead ability, in order to obtain a near-optimal schedule for an agile satellite constellation. They also tested the performance of various constellation configurations based on the schedules generated by the heuristic. Wang and Reinelt [9] discussed the AEOS scheduling problem for

the first environmental and disaster monitoring and prediction satellite constellation of China. They proposed a nonlinear model of the problem and developed a heuristic with conflict avoidance, limited backtracking, and download-as-needed features. Meanwhile, a decision support system based on the model and the heuristic is also provided. Kananub et al. [10] show different observation situations of agile and nonagile EOSs. AEOS improves observation efficiency but also increases the difficulties of observation scheduling task. Tangpattanakul et al. [11] modeled the AEOS problem as a multiobjective optimization problem which maximizes the total profits and minimizes the maximum profit difference between users simultaneously. They proposed an indicator-based multiobjective local search to solve the problem. Xu et al. [12] assumed that different observation tasks may have priority levels, and the objective is to maximize the total priority of selected tasks. Meanwhile, they developed constructive algorithms to solve the problem, which adopt a priority-based sequential construction procedure to avoid conflicts and generate feasible solutions. Other works related to the AEOS problems can be found in [13–17].

However, the scheduling problem is more difficult to solve as the attitude maneuver capability is more excellent. Meanwhile, the resource constraint is not as tight as before in the new generation of agile satellites. So the critical constraint is the time-dependent temporal constraint, which means the minimum transition time taken by a maneuver between the end of the former acquisition and the start of the following one is not constant and depends on the precise time at which the transition is triggered [1]. This aspect is close to work on time-dependent scheduling [18, 19], where transition times take particular forms, piecewise constant or piecewise linear.

Pralet and Verfaillie [20] proposed the time-dependent simple temporal networks (TSTN), which cover temporal constraints for which the minimum and maximum distances required between two temporal positions x and y are not necessarily constant but may depend on the assignments of x and y. The TSTN are applied by several researchers [20, 21] on the management of temporal constraints for agile satellites.

On the basis of this, a branch and bound algorithm for the optimal solution of the agile satellite scheduling problem is proposed in this paper.

The remainder of this paper is organized as follows. In Section 2, we proposed the mathematical model of the scheduling problem of the AEOS and analyzed the characteristics of the attitude maneuver function. In Section 3, the details of the branch and bound algorithm are illustrated, which contains the expression of a solution, the initialization method, and three pruning strategies. Experimental examples are designed according to the actual use requirements of the AEOSs and the examples are used to validate the efficiency of the algorithm in Section 4. Section 5 concludes the paper.

2. Agile Earth Observing Satellite Scheduling Problem

2.1. Problem Description. It is assumed that the resource constraints such as memory and energy are no longer the tight constraints of the scheduling problem with the development of the satellite platform. Therefore, temporal constraints are imperative in the agile satellite scheduling problem considered in this paper, which include visible time window constraint and time-dependent transition constraint.

Since satellites are orbital spacecraft, a target is visible to a satellite only when it is flying over it. Compared to nonagile satellites, agile satellite's pitching ability extends the visible time window [1]. Figure 1 illustrates the temporal constraints of agile satellites. As is shown in Figure 1(a), the visible time window of target_1 is [ws_1,we_1]. At any point of the time window, the satellite has a unique attitude for observing target_1. The attitude at ws_1 is attitude_ws_1. Figure 1(b) shows that an attitude maneuver is needed between two consecutive observations. Suppose that the end attitude of target_1 is atie_1 and the start attitude of target_2 is atis_2, and the difference between the start time of the latter observation and the end time of the former observation must be larger than the attitude maneuver time: that is, $\mathrm{st}_2 - \mathrm{et}_1 > d\min(\mathrm{atie}_1, \mathrm{atis}_2)$, where $d\min(\mathrm{atie}_1, \mathrm{atis}_2)$ denotes the minimum attitude maneuver time between atie_1 and atis_2. One notices that the minimum transition time taken by a maneuver between two observations is not constant and it depends on the precise time at which the transition is triggered.

2.2. Problem Formulation. In the modeling, we assume that all the targets have been processed into strip tasks which are along-track for good imaging quality as in [22]. The tasks such as point target and stereo imaging could be regarded as target with different duration times.

A set of notations of the model is shown as follows:

p_i: the profit of target i

ws_i: the start time of the time window of target i

we_i: the end time of the time window of target i

dur_i: the duration time of target i

st_i: the start imaging time of target i

et_i: the end imaging time of target i

atis_i: the attitude of the start imaging time of target i, including the pitch, roll, and azimuth

atie_i: the attitude of the end imaging time of target i, including the pitch, roll, and azimuth

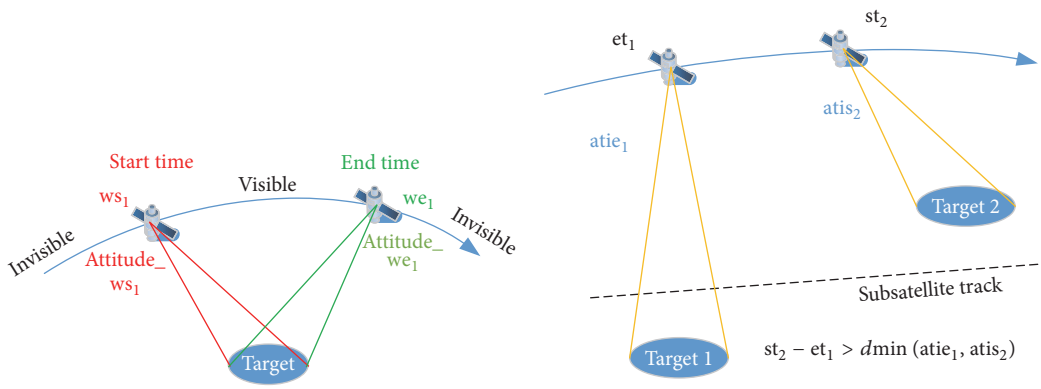
$d\min(\cdot, \cdot)$: the minimum transition time between two attitudes

NT: the total number of targets

S: the set of all combinations of targets.

Objective function is

$$\max \sum_{i=1}^{i=\mathrm{NT}} x_i \cdot p_i. \tag{1}$$



Time window constraint (a)

Time-dependent transition constraint (b)

FIGURE 1: Temporal constraint of agile satellite.

Decision variables are

$$x_i = \begin{cases} 1, & \text{if task } i \text{ is executed,} \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

$$y_{ij} = \begin{cases} 1, & \text{if task } j \text{ is executed after task } i, \\ 0, & \text{otherwise.} \end{cases}$$

Constraints are

$$\text{ws}_i \le \text{st}_i < \text{et}_i < \text{we}_i, \quad \forall x_i = 1; \tag{3}$$

$$\text{et}_i = \text{st}_i + \text{dur}_i, \quad \forall x_i = 1; \tag{4}$$

$$\text{et}_i + d\text{min}\left(\text{atie}_i, \text{atis}_j\right) < \text{st}_j \quad \forall y_{ij} = 1; \tag{5}$$

$$y_{ii} = 0; \tag{6}$$

$$x_0 = 1; \tag{7}$$

$$x_{\text{NT}+1} = 1; \tag{8}$$

$$x_j = \sum_{i=0}^{i=\text{NT}} y_{ij}, \quad j = 1, \ldots, \text{NT} + 1; \tag{9}$$

$$x_i = \sum_{j=1}^{j=\text{NT}+1} y_{ij}, \quad i = 0, \ldots, \text{NT}; \tag{10}$$

$$\sum_{i \in S} \sum_{j \in S} y_{ij} \le |S| - 1, \quad S \subseteq \{1, \ldots, \text{NT}\}, \ S \ne \emptyset; \tag{11}$$

$$x_i = 1 \wedge x_j = 1, \quad \forall y_{ij} = 1; \tag{12}$$

$$x_i \in \{0, 1\}, \quad y_{ij} \in \{0, 1\}, \quad i, j = 1, \ldots, \text{NT}. \tag{13}$$

Equation (1) states that the objective is to maximize the sum of the profits of the executed targets. Equation (2) introduces the decision variables of the model, where x_i takes a value of 1 if target i is executed and $x_i = 0$; otherwise; y_{ij} takes a value of 1 if target j is executed after target i and $y_{ij} = 0$ otherwise.

Equations (3)–(13) give the constraints of the problem. Equation (3) states that any scheduled target should be executed in the time window. Equation (4) states that the end imaging time of a scheduled target is the sum of the start imaging time and the duration time of the target. Equation (5) states that the start imaging time of the following target should be bigger than the sum of the end imaging time of the former target and the maneuver time. Equation (6) states that a target could not be executed following itself. Equations (7) and (8) state that there are a virtual starting target and a virtual ending target whose start time and duration time are both 0. Equations (9) and (10) state that any executed target could only have one following target and one previous target. Equation (11) is the classical Dantzig, Fulkerson, and Johnson (DFJ) subtour elimination constraint [23]. Equation (12) states that target i and target j are both executed if target j is executed after target i. Equation (13) states the domain of the variables.

The key to optimizing this model is to determine a feasible sequence of targets, which maximize the overall profits while satisfying all temporal constraints. Therefore, it is necessary to further analyze the attitude maneuver characteristics of agile satellites.

2.3. The Characteristics of the Function dmin. In this subsection, we introduce how to calculate the function $d$min and the method to compute the earliest start time of a target on the basis of the analysis of the characteristics of $d$min.

The function $d$min returns the minimum transition time given two attitudes. The maneuver process includes three phases: acceleration, plateau, and deceleration. The satellite first uniformly accelerates to a maximum speed and then remains in the maximum speed for a period and hereafter

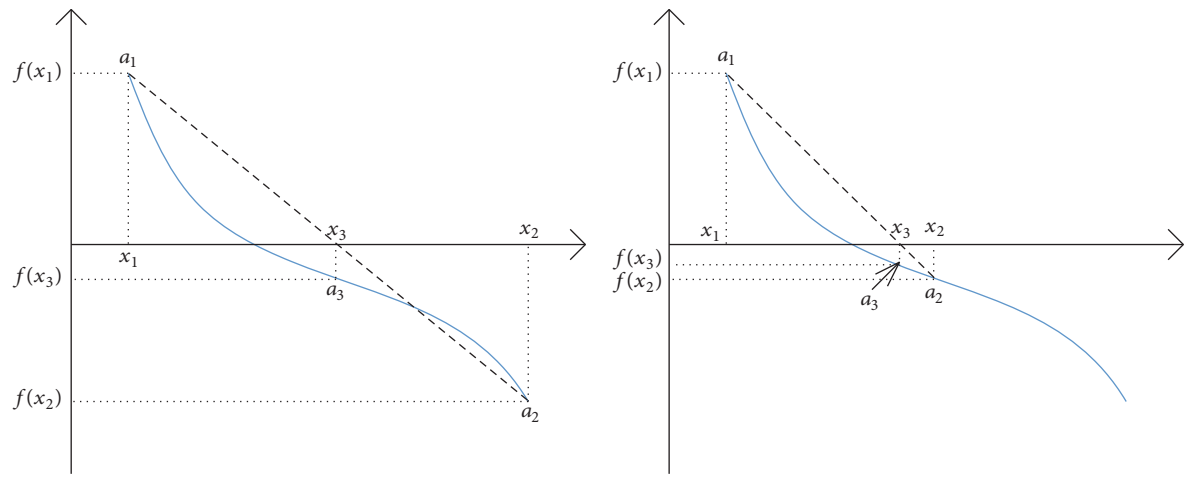


Figure 2: Interpolation procedure used for computing the earliest start time.

decelerates until stop. After maneuver, a tranquilization time is required to reach the pointing stability. Attitudes remain fixed during imaging and all targets are preprocessed to be along-track strips.

Equation (5) is a key constraint of the AEOS scheduling problem. We note that the "delay function" defined in [20] is in fact a deformation of (5).

Definition 1. The delay function associated with an attitude maneuver constraint ct: $(\mathrm{et}_i, \mathrm{st}_j, d\min)$ is function $\mathrm{delay}_{\mathrm{ct}}$: $D() \times D() \to \mathbb{R}$ defined by $\mathrm{delay}_{\mathrm{ct}}(\mathrm{et}_i, \mathrm{st}_j) = \mathrm{et}_i + d\min(\mathrm{atie}_i, \mathrm{atis}_j) - \mathrm{st}_j$, where $y_{ij} = 1$.

Informally, $\mathrm{delay}_{\mathrm{ct}}(\mathrm{et}_i, \mathrm{st}_j)$ is the delay obtained in st_j if a transition in minimum time from atie_i to atis_j is triggered at time et_i. This delay corresponds to the difference between the minimum arrival time associated with the transition $\mathrm{et}_i + d\min(\mathrm{atie}_i, \mathrm{atis}_j)$ and the required arrival time st_j. A strictly negative delay corresponds to a transition ending before deadline st_j, which means the constraint is satisfied. A strictly positive delay indicates that the constraint is violated. A null delay corresponds to an arrival right on time.

Definition 2. An attitude maneuver constraint ct: $(\mathrm{et}_i, \mathrm{st}_j, d\min)$ is said to be delay-monotonic iff its delay function $\mathrm{delay}_{\mathrm{ct}}(\cdot,\cdot)$ satisfies the conditions below, where $D(\ )$ is the domain of the time variable:

$$\begin{aligned}
&\left(\mathrm{et}_i \le \mathrm{et}_i'\right) \longrightarrow \left(\mathrm{delay}_{\mathrm{ct}}\left(\mathrm{et}_i, \mathrm{st}_j\right) \le \mathrm{delay}_{\mathrm{ct}}\left(\mathrm{et}_i', \mathrm{st}_j\right)\right) \\
&\qquad\qquad \forall \mathrm{et}_i, \mathrm{et}_i' \in D\left(\mathrm{et}_i\right),\ \forall \mathrm{st}_j \in D\left(\mathrm{st}_j\right), \\
&\left(\mathrm{st}_j \le \mathrm{st}_j'\right) \longrightarrow \left(\mathrm{delay}_{\mathrm{ct}}\left(\mathrm{et}_i, \mathrm{st}_j\right) \ge \mathrm{delay}_{\mathrm{ct}}\left(\mathrm{et}_i, \mathrm{st}_j'\right)\right), \\
&\qquad\qquad \forall \mathrm{et}_i \in D\left(\mathrm{et}_i\right),\ \forall \mathrm{st}_j, \mathrm{st}_j' \in D\left(\mathrm{st}_j\right),
\end{aligned} \tag{14}$$

where $y_{ij} = 1$.

This definition means that, for being delay-monotonic, an attitude maneuver constraint ct: $(\mathrm{et}_i, \mathrm{st}_j, d\min)$ on two successive observations i and j must verify that, on the one hand, the later the transition is triggered in et_i, the greater is the delay in st_j, and on the other hand the earlier the transition must end in st_j, the greater the delay is. The notion of delay-monotonicity is related to the notions of monotonic constraints, defined, for instance, in [24]. The delay-monotonicity of the attitude maneuver time constraint is the theoretical basis of the segment domination based pruning strategy proposed in Section 3.5.

The transition temporal constraint of the agile satellite is delay-monotonic as shown in [20]. Now, we introduce a method to compute the earliest start time of the next target given the end time and the attitude of the current target. Such an evaluation is an optimization problem in itself, and an iterative method for approximating the earliest start time of the next target is illustrated in Figure 2 and formally defined in Algorithm 1. This method corresponds to the standard false position method, used to find a zero of an arbitrary function. Applied to the case of t-simple temporal constraints, the method works as follows. If leftmost point $a_1 = (x_1, f(x_1))$ has a negative delay $f(x_1) \le 0$, then a_1 is directly returned. Otherwise, if rightmost point $a_2 = (x_2, f(x_2))$ has a strictly positive delay $f(x_2) > 0$, then $+\infty$ is returned, which means there is no start time satisfying the temporal constraint. Otherwise, points a_1 and a_2 have opposite delay-signs ($f(x_1) \le 0 \wedge f(x_2) > 0$) and then compute delay $f(x_3)$ in x_3, the x-value of the intersection between segment (a_1, a_2) and the x-axis. See Figure 1 for an illustration. If the delay in $a_3 = (x_3, f(x_3))$ is negative, then the mechanism is applied again by taking $a_2 \leftarrow a_3$, otherwise taking $a_1 \leftarrow a_3$ if the delay is positive. The procedure stops when value $f(x_3)$ is less than a given precision. Algorithm 1 gives a possible way to compute the earliest start time of the next target.

3. A Branch and Bound Algorithm

In this section, the branch and bound (B&B) algorithm proposed in this paper is introduced. The B&B algorithm

```
      Input: wts_j = [t_1, t_2] // the time window of the start time
             et_i // the end time of the previous target i
             maxIter // a maximum number of iterations
             pre // a desired precision
      output: t_2 // the earliest start time of the next target
(1)   f_1 ← delay(et_i, t_1); if f_1 ≤ 0 then return t_1
(2)   f_2 ← delay(et_i, t_2); if f_2 > 0 then return +∞
(3)   for i = 1 to maxIter
(4)      t_3 ← (f_1 * t_2 − f_2 * t_1)/(f_1 − f_2)
(5)      f_3 ← delay(et_i, t_3)
(6)      if |f_3| < pre then return t_3
(7)         else if f_3 > 0 then (t_1, f_1) ← (t_3, f_3)
(8)              else (t_2, f_2) ← (t_3, f_3)
(9)   end for
(10)  return t_2
```

Algorithm 1: Iterative interpolation algorithm.

is actually a combined algorithm, and different components of the algorithm will be introduced, respectively, such as the expression of a solution, the initialization method, the pruning mechanism, and current best solution updating method.

3.1. General Framework. In this subsection, the general framework of the B&B algorithm is introduced. Components that need to be elaborated in a B&B algorithm include the branching strategy and variable branching order.

Algorithm 2 shows the general procedure of the B&B algorithm. The algorithm adopts depth-first search strategy to search the solution space. At each node of the search tree, the algorithm first computes the total profit of the corresponding solution and updates the current best solution if an improved solution is discovered. The algorithm then tests if the search space below the current node can be pruned. Specifically, three conditions are tested in the following order: (1) whether the symmetry breaking based pruning condition (lines (13)~(15)) is satisfied; (2) whether the segment domination based pruning condition (lines (16)~(19)) is satisfied; and (3) whether its upper bound is lower than the global lower bound (the total profit of the current best solution, lines (24)~(27)). Once a condition is satisfied, the subtree following the current node is pruned and the algorithm backtracks. The node needs to be extended if none of the three conditions is verified. The algorithm returns a current best solution in the first case and an optimal one in the second case. The details of the three pruning strategies (the symmetry breaking based pruning strategy, the segment domination based pruning strategy, and the bound based pruning strategy) will be presented in the following Sections 3.4~3.6.

In order to improve the efficiency of the B&B algorithm, it is a common way to provide a good lower bound of the problem at the beginning of the algorithm. So a look-ahead construction method is proposed to initialize the lower bound of the B&B algorithm, which is illustrated in Section 3.3.

3.2. Solution Space and Solution Expression. Before presenting the search space of the B&B algorithm, two concepts are first introduced: "sequence solution" and "schedule solution." A "sequence solution" is represented by a sequence of targets whose ordering is determined, while feasibility of the solution is not tested and the execution time of the observations is not arranged yet. The "schedule solution" is, on the contrary, a feasible solution with all temporal constraints satisfied and the execution time of the observations arranged.

The B&B algorithm searches in the "sequence solution" space. Every time the B&B algorithm examines a node in the search tree where a sequence solution lies, the algorithm applies a "schedule builder" (SB) to build a "schedule solution" based on the sequence solution and then computes the total profits of the schedule solution.

SB uses a greedy construction heuristic. Given a sequence solution ps, SB considers targets in the same order as they appear in ps. SB tries to arrange the observation of each target at its earliest possible start time. If a target cannot be scheduled due to the violation of temporal constraints, SB discards it and continues to examine the next one. The schedule solution is obtained when all targets in the sequence solution are examined by SB. The total profits of the schedule solution are the sum of the profits of all selected targets.

Proposition 3. *There must be a sequence solution in the search space that can be mapped onto an optimal schedule solution by the proposed schedule builder.*

Proof. The schedule solution space can be divided into two subsets, say $S1$ and $S2$. $S1$ contains all solutions whose selected observations are arranged at their earliest possible start time. In other words, in these solutions, no observation can be started earlier without violating the maneuver and time window constraints. While $S2$ is the complementary set of $S1$. We note that there must be an optimal schedule solution in $S1$. This is because, for any solution ss' of $S2$, there must be a solution ss from $S1$ whose objective value is not worse

```
Input: tasks // the target list
       atiState // the initial attitude state of the satellite
       psC // the current sequence solution
       taskOrder // the extending order of the targets
       psOptC // the current best sequence solution
       proOptC // the current best profit
       num // the number of look-ahead targets
Output: proOpt // the optimal sequence solution
        psOpt // the optimal profit
(1)  if IsEmpty(psOptC)
(2)       [psOptC, proOptC] ← LookaheadCon(tasks, atiState, num); //Initialization
(3)  end if
(4)  ssC ← []; proC ← 0;
(5)  if ~IsEmpty(psC)
(6)    ssC ← ScheduleBuilder(psC, tasks, atiState);
(7)    proC ← Profit(ssC); // Computing the current profit
(8)    if proC > proOptC && IsConsistent(psC, ssC)
(9)      proOptC ← proC; psOptC ← psC; //Updating the current best plan
(10)   end if
(11) end if
(12) unscheduledTasks ← SelectTask(ssC); // Selecting the targets which still
     have a visible time window
(13) if IsEmpty(unscheduledTasks) || ~IsConsistent(psC, ssC)
(14)   return; // Backtracking if psC is not a consistent solution
(15) else
(16)   dominatedBool ← CurDominated(psC, psOptC, tasks)
(17)   if dominatedBool
(18)     return; // Backtracking if current solution is dominated
(19)   end if
(20)   upBound ← BasicUpBound(ssC, tasks); // Computing the upper bound
(21)   if proC + upBound < proOptC
(22)     return; // Backtracking if the upper bound of current solution is less than proOptC
(23)   else
(24)     dominatingBool ← OptDominated(psC, psOptC, tasks)
(25)     if dominatingBool //Updating the current best solution
(26)       [psOptC, proOptC] ← UpdateOpt(psC, psOptC, tasks);
(27)     end if
(28)     numOfNext ← NumberOfTask(unscheduledTasks); // Determining the number of targets that
         have not been tried
(29)     for i ← 1 to numOfNext
(30)       nextNode ← SelectTask(unscheduledTasks, i);
(31)       psC ← Apendix(psc, nextNode); //Extending the current solution
(32)       [proOptC, psOptC] ← BranchBound(tasks, atiState, psC, taskOrder, proOptC, psOptC)
(33)     end for;
(34)   end if
(35) end if
(36) proOpt ← proOptC;
(37) psOpt ← psOptC;
```

Algorithm 2: The branch and bound algorithm.

than that of ss′. In fact, ss can be obtained by anticipating all selected observations of ss′ at their earliest possible start time. We also note that there must exist a sequence solution ps in the sequence solution space (which is also the search space of B&B) that can be mapped onto ss with the proposed schedule builder. The ps is the sequence solution having the same observation list and order as in ss. □

3.3. Look-Ahead Construction Method. It is known that a good initial solution will contribute to speed up the pruning process. We propose a look-ahead construction method (LACM) to provide a good lower bound at the very beginning. Let us call the branch and bound algorithm without the initialization, pruning method, and the updating method (lines (1)~(3), (16)~(19), and (24)~(27)) a basic branch and

```
Input: tasks // the target list
       atiState // the initial attitude state of the satellite
       num // the number of look-ahead targets
Output: ps // the sequence solution
        proC // the profit of the sequence solution
(1)  ps ← [];
(2)  taskOrder ← SortByTime(tasks);
       // Ascending order by the middle time of time windows
(3)  laTasks ← ExtractTasks(tasks, num, atiState);
       //Extracting the nearest num targets
(4)  parOpt ← BasicBranchBound(laTasks, atiState, [], taskOrder, 0, []);
(5)  while ~isempty(parOpt)
(6)       ps ← Apendix(ps, parOpt(1)); //selecting the first task in parOpt
(7)       atiState ← ComAtitude(laTasks, ps, atiState); //Updating the attitude
(8)       laTasks ← ExtractTasks(tasks, num, atiState); // Extracting at most num near targets
(9)       parOpt ← BasicBranchBound(laTasks, atiState, [], taskOrder, 0, []);
(10) end while
(11) ss ← ScheduleBuilder(ps, tasks, atiState);
(12) proC ← Profit(ss);
```

Algorithm 3: Look-ahead construction method.

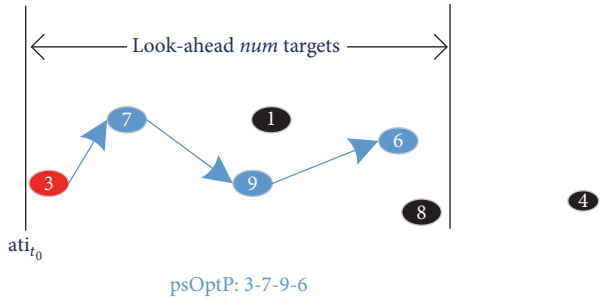


Figure 3: Look-ahead construction method.

bound algorithm (b-B&B for short), which could rapidly find the optimal solution of a small scale problem which contains about 12 targets. We design a heuristic algorithm based on the idea of rolling planning. The algorithm decides each target to be scheduled or not progressively according to the "natural" order which sorts the targets in their increasing start time of time windows. Given the current satellite attitude state, the algorithm finds the optimal solution of the next 12 targets with b-B&B and then selects the first target in the optimal partial solution. Afterwards, the algorithm updates the satellite attitude state and looks for the next optimal solutions of the subsequent 12 targets. The algorithm stops until the last target is considered. The pseudocode of the LACM is shown in Algorithm 3. It should be noted that the while loop in the algorithm is not an infinite loop. This is because the number of the targets is limited, and each time the satellite selects a target to observe, it removes the targets that violate the temporal constraints from the candidate set. So the candidate set is shrinking during the process and the while loop stops when the candidate set becomes empty.

An example is provided to illustrate the idea of our look-ahead initialization method in Figure 3. Given the current attitude state ati_{t_0}, the algorithm looks ahead num (num = 12) targets and finds that the optimal partial solution is $3 \rightarrow 7 \rightarrow 9 \rightarrow 6$. Then it selects target 3 to be scheduled, which is contained in the optimal solution, and updates the state to atie_3 (the attitude state at the end time of target 3).

3.4. Symmetry Breaking Based Pruning Strategy. First a definition is introduced here, which is useful for presenting the details of the symmetry breaking based pruning strategy.

Definition 4. A sequence solution ps is said to be consistent with the schedule solution ss provided by the schedule builder when ss contains all the observations in ps (i.e., |ps| = |ss|, where | | is the number of observations in the solution).

As is shown in Figure 4, the sequence solution ps2 is not consistent with its schedule solution ss, as target_1 and target_8 are not included in ss due to the violation of the attitude maneuver constraint. In contrast, the sequence solution ps1 is consistent with ss since |ps1| = |ss|.

It could be learned from Figure 4 that different sequence solutions may lead to the same schedule solution using the proposed SB. In other words, there are many symmetry configurations in the B&B search tree. Examining these symmetry configurations is a waste of time and does not lead to any improvement. To avoid this, a symmetry breaking based pruning strategy (SBBPS) is proposed and used during the searching process. The idea of the SBBPS is to break symmetries by pruning the subtrees below the sequence solutions that are inconsistent with their schedule solutions. This allows us to prevent examining duplicate sequence solutions that correspond to the same schedule solution. Specifically, at each node of the B&B algorithm, given a sequence solution ps, the algorithm builds a schedule solution ss with SB. If ps is inconsistent with ss, the algorithm prunes the subtrees below the node under consideration and alters the search

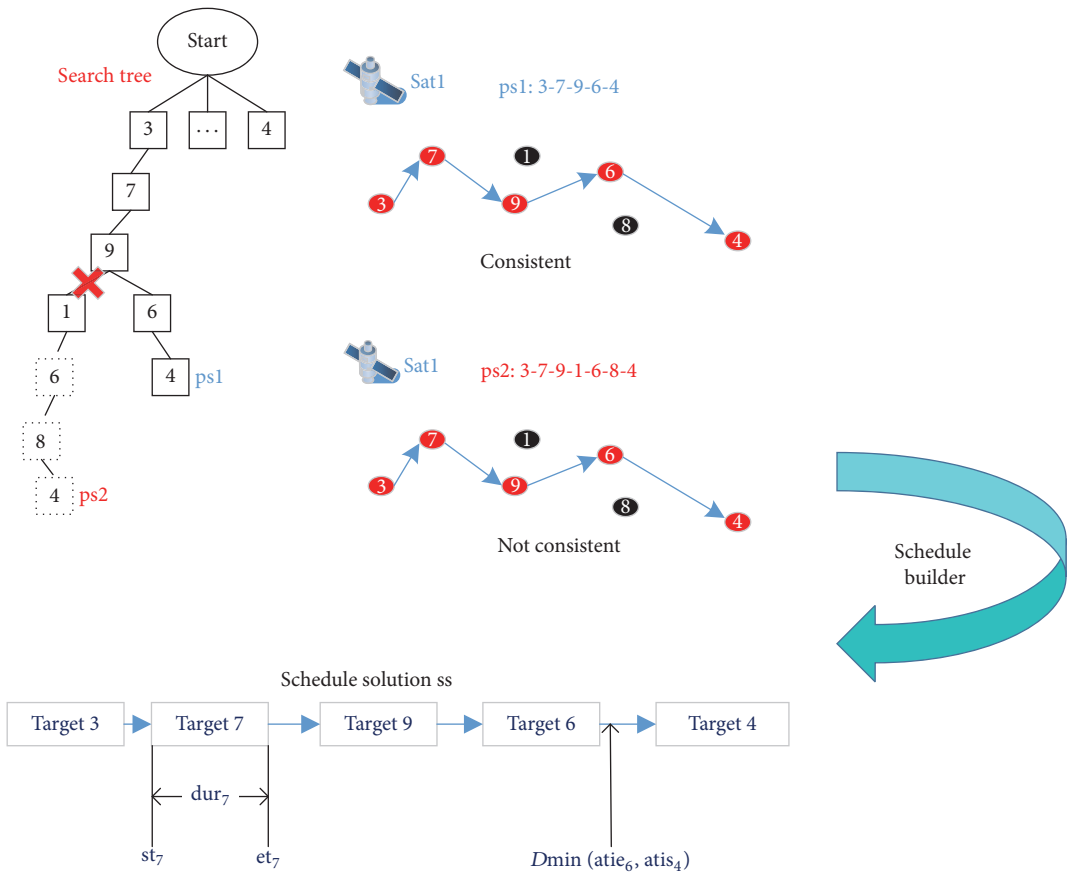


FIGURE 4: Sequence solution and scheduling builder.

to another branch. In the example shown in Figure 4, the subspace under the node of ps2($3 \to 7 \to 9 \to 1$) is pruned since ps2 is inconsistent with its schedule solution. Specifically, target_1 of ps2 cannot be scheduled since time is not enough for the satellite to maneuver from target_9 to target_1. SBBPS is used in lines (13)~(15) of Algorithm 2.

3.5. Segment Domination Based Pruning Strategy. In this subsection, we propose a segment domination based pruning strategy (short for SDBPS hereafter) to further improve the search efficiency of the B&B algorithm. Before introducing the method, we first give a set of notations regarding the sequence solution ps. Let ps^j denote the jth target in the sequence and "+" connect two subsequences. Then $ps = ps^{1,\dots,j} + ps^{j+1,\dots,|ps|}, \forall j \le |ps|$. Pro(ps) returns the total profits of the schedule solution ss generated by SB. Notice that $\text{Pro}(ps) \neq \text{Pro}(ps^{1,\dots,j}) + \text{Pro}(ps^{j+1,\dots,|ps|})$ if ps is inconsistent with its schedule solution.

The main idea of the SDBPS is to prune sequence solutions that are dominated by a segment of the current best feasible solution. A node psC can be pruned if there is a partial solution ps′ that has a higher profit and consumes less execution time than ps and all the targets observed in ps have no visible time window after the end time of psC.

Theorem 5. *Given a current best sequence solution psOptC and a current node psC($|psC| = k$), psC can be pruned if $\exists j \le |psOptC|$ such that $\{Pro(psOptC^{1,\dots,j}) > Pro(psC^{1,\dots,k-1})$ and $st_{psC^k} \ge et_{cBPlan^j} + dmin(atie_{cBPlan^j}, atis_{psC^k})\}$.*

Proof. $\text{Pro}(\text{psOptC}^{1,\dots,j}) > \text{Pro}(\text{psC}^{1,\dots,k-1})$ indicates that $\text{psC}^{1,\dots,k-1}$ is dominated by $\text{psOptC}^{1,\dots,j}$ in terms of total profit and $\text{st}_{\text{psC}^k} \ge \text{et}_{\text{psOptC}^j} + d\text{min}(\text{atie}_{\text{psOptC}^j}, \text{atis}_{\text{psC}^k})$ ensures the subsequence $\text{psOptC}^{1,\dots,j}+\text{psC}^k$ is consistent with its schedule solution. □

As is shown in Figure 5, the current node psC is $3 \to 7 \to 1 \to 6$ and the current best solution psOptC is $3' \to 7' \to 9' \to 6' \to 4'$. The target with "′" means it is in the current best solution psOptC. If (15) and (16) are simultaneously true, psC is worse than $3' \to 7' \to 9' \to 6$ (which is for sure consistent with its schedule solution). For this reason, the current node psC can be pruned. Equation (16) also requires that the imaging start time of target_6 in $3' \to 7' \to 9' \to 6$ must be earlier than the imaging start time of target_6 in $3 \to 7 \to 1 \to 6$

$$\text{Pro}\left(3' \longrightarrow 7' \longrightarrow 9'\right) > \text{Pro}\left(3 \longrightarrow 7 \longrightarrow 1\right), \tag{15}$$

$$\text{st}_6 \ge \text{et}_{9'} + d\text{min}\left(\text{atie}_{9'}, \text{atis}_6\right). \tag{16}$$

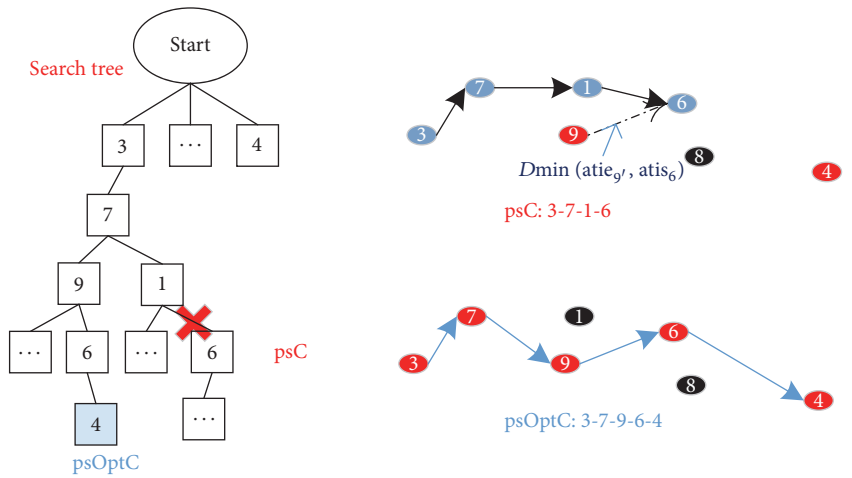


Figure 5: Schematic diagram of the segment domination based pruning strategy.

To test whether the condition of SDBPS is verified for the current sequence solution psC, the algorithm identifies a critical position j of psOptC that satisfies

$$\begin{aligned}\text{Pro}\left(\text{psOptC}^{1,\ldots,j}\right) &\geq \text{Pro}\left(\text{psC}^{1,\ldots,k-1}\right)\\ &> \text{Pro}\left(\text{psOptC}^{1,\ldots,j-1}\right).\end{aligned} \tag{17}$$

If there is no such j, it means an improved best solution psC is found and the current best solution psOptC is updated with psC. Otherwise, the algorithm tests (18) and prunes psC if it is true

$$\text{st}_{\text{psC}^k} \geq \text{et}_{\text{psOptC}^j} + d\,\text{min}\left(\text{atie}_{\text{psOptC}^j}, \text{atis}_{\text{psC}^k}\right). \tag{18}$$

The following corollary justifies the practice of judging the domination of the current sequence solution with respect to the current best solution via the critical position j.

Corollary 6. *If $psOptC^{1,\ldots,j}$ does not dominate the current sequence solution $psC^{1,\ldots,k-1}$, no other subsequences of psOptC can dominate $psC^{1,\ldots,k-1}$.*

Proof. We have

$$\text{Pro}\left(\text{psOptC}^{1,\ldots,h}\right) < \text{Pro}\left(\text{psC}^{1,\ldots,k-1}\right), \quad \forall h < j, \tag{19}$$

according to

$$\begin{aligned}\text{Pro}\left(\text{psOptC}^{1,\ldots,j}\right) &\geq \text{Pro}\left(\text{psC}^{1,\ldots,k-1}\right)\\ &> \text{Pro}\left(\text{psOptC}^{1,\ldots,j-1}\right).\end{aligned} \tag{20}$$

And $\forall h > j$, $\text{et}_{\text{psOptC}^h} > \text{et}_{\text{psOptC}^j}$, according to the delay-monotonic (Definition 2), so

$$\begin{aligned}&\text{delay}_{\text{ct}}\left(\text{et}_{\text{psOptC}^j}, \text{st}_{\text{psC}^k}\right) \leq \text{delay}_{\text{ct}}\left(\text{et}_{\text{psOptC}^h}, \text{st}_{\text{psC}^k}\right),\\ &\therefore \text{et}_{\text{psOptC}^j} + d\text{min}\left(\text{atie}_{\text{psOptC}^j}, \text{atis}_{\text{psC}^k}\right) - \text{st}_{\text{psC}^k}\\ &\leq \text{et}_{\text{psOptC}^h} + d\text{min}\left(\text{atie}_{\text{psOptC}^h}, \text{atis}_{\text{psC}^k}\right) - \text{st}_{\text{psC}^k},\\ &\because \text{st}_{\text{psC}^k} < \text{et}_{\text{psOptC}^j} + d\text{min}\left(\text{atie}_{\text{psOptC}^j}, \text{atis}_{\text{psC}^k}\right),\\ &\therefore 0 < \text{et}_{\text{psOptC}^j} + d\text{min}\left(\text{atie}_{\text{psOptC}^j}, \text{atis}_{\text{psC}^k}\right) - \text{st}_{\text{psC}^k}\\ &\leq \text{et}_{\text{psOptC}^h} + d\text{min}\left(\text{atie}_{\text{psOptC}^h}, \text{atis}_{\text{psC}^k}\right) - \text{st}_{\text{psC}^k},\\ &\therefore \text{st}_{\text{psC}^k} < \text{et}_{\text{psOptC}^h} + d\text{min}\left(\text{atie}_{\text{psOptC}^h}, \text{atis}_{\text{psC}^k}\right).\end{aligned} \tag{21}$$

So there is no position h other than j in psOptC such that $\text{cBPlan}^{1,\ldots,h}$ dominates the sequence solution $\text{psC}^{1,\ldots,k-1}$. □

3.6. Bound Based Pruning Strategy

3.6.1. Upper Bounding Method. An efficient method is adopted to estimate the upper bound, which simply adds the profit of the current branch and the sum of profits of all the targets which still have a time window to be imaged according to the current state of the satellite. This upper bounding method is demonstrated to be effective according to our experiments.

As shown in Figure 6, the current node is psC($3 \rightarrow 7 \rightarrow 9 \rightarrow 6$). The imaging end time of the last target observation is et_6. The target_8 and target_4 still have a time window to be observed, since et_8 and et_4 are both greater than et_6. So the upper bound of the current node psC is $(p_3 + p_7 + p_9 + p_6) + (p_8 + p_4)$.

3.6.2. Segment Replacement Based Lower Bound Updating Method. In this subsection, a segment replacement based

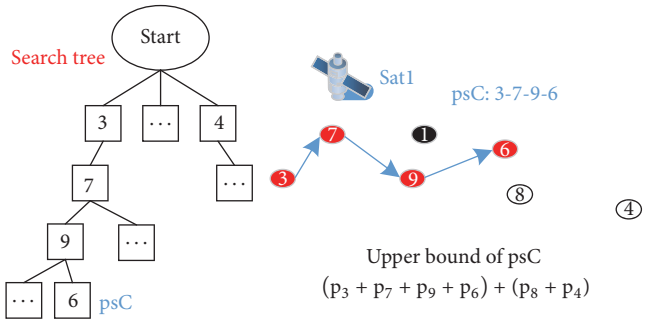


Figure 6: Schematic diagram of the upper bound.

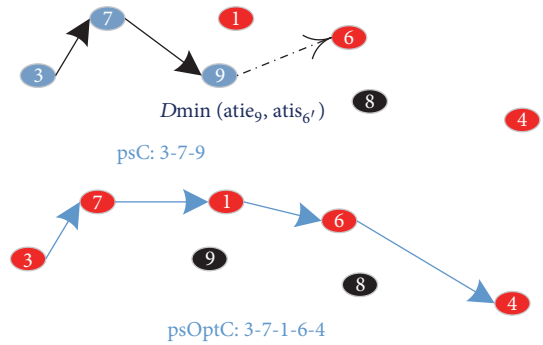


Figure 7: Schematic diagram of the segment replacement based lower bound updating method.

lower bound updating method (short for SRBLBUM hereafter) is proposed to update the current best solution psOptC with the similar idea of the pruning method, which attempts to check whether the current node psC($|\text{psC}| = k$) dominates a partial solution of the current best solution.

If there is some $j < |\text{psOptC}|$ such that $\{\text{Pro}(\text{psOptC}^{1\cdots j}) < \text{Pro}(\text{psC})$ and $\text{st}_{\text{psOptC}^{j+1}} \geq \text{et}_{\text{psC}^k} + d\text{min}(\text{atie}_{\text{psC}^k}, \text{atis}_{\text{psOptC}^{j+1}})\}$, then the current best solution could be updated by (22), where psOptC' means the new current best solution

$$\text{psOptC}' = \text{psC} + \text{psOptC}^{j+1\cdots|\text{pOptC}|}. \tag{22}$$

As is shown in Figure 7, if the current node psC($3 \to 7 \to 9$) satisfies the following inequalities (23), then the partial solution $3' \to 7' \to 1'$ could be replaced with $3 \to 7 \to 9$, where targets with "′" mean they are in the current best solution psOptC as before. And a new better current best solution $3 \to 7 \to 9 \to 6' \to 4'$ is generated

$$\begin{aligned} &\text{Pro}\left(3' \longrightarrow 7' \longrightarrow 1'\right) < \text{Pro}\left(3 \longrightarrow 7 \longrightarrow 9\right), \\ &\qquad \text{st}_{6'} \geq \text{et}_9 + d\text{min}\left(\text{atie}_9, \text{atis}_{6'}\right). \end{aligned} \tag{23}$$

And then, an effective method is developed to judge whether the current node psC dominates a partial solution of the current best solution psOptC. The critical dominated position j of psOptC could be found according to (24), where $|\text{psC}| = k$

$$\begin{aligned} &\text{Pro}\left(\text{psOptC}^{1,\ldots,j}\right) < \text{Pro}\left(\text{psC}\right) \\ &\qquad \leq \text{Pro}\left(\text{psOptC}^{1,\ldots,j+1}\right). \end{aligned} \tag{24}$$

If there is no such j, it means the current node psC could not be better than a partial solution of the current best solution psOptC. Otherwise, judge j whether to meet (25). Replace psOptC with $\text{psOptC}' = \text{psC} + \text{psOptC}^{j+1\cdots|\text{pOptC}|}$, if it is satisfied,

$$\text{st}_{\text{psOptC}^{j+1}} \geq \text{et}_{\text{psC}^k} + d\,\text{min}\left(\text{atie}_{\text{psC}^k}, \text{atis}_{\text{psOptC}^{j+1}}\right). \tag{25}$$

Corollary 7 illustrates that it is rational to judge whether the current branch dominates the partial current best solution via the critical dominated position j.

Corollary 7. *If the current node psC does not dominate the partial solution $\text{psOptC}^{1,\ldots,j}$, where j is the critical dominated position satisfying (24), there is no other position h of psOptC that makes $\text{psOptC}^{1\cdots h}$ be dominated by psC.*

Proof. We have

$$\text{Pro}\left(\text{psOptC}^{1,\ldots,h}\right) > \text{Pro}\left(\text{psC}\right), \quad \forall h > j, \tag{26}$$

according to

$$\begin{aligned} &\text{Pro}\left(\text{psOptC}^{1,\ldots,j}\right) < \text{Pro}\left(\text{psC}\right) \\ &\qquad \leq \text{Pro}\left(\text{psOptC}^{1,\ldots,j+1}\right); \end{aligned} \tag{27}$$

And $\forall h < j$, $\text{st}_{\text{psOptC}^{h+1}} < \text{st}_{\text{psOptC}^{j+1}}$, according to the delay-monotonic (Definition 2), and then

$$\begin{aligned} &\text{delay}_{\text{ct}}\left(\text{et}_{\text{psC}^k}, \text{st}_{\text{psOptC}^{h+1}}\right) \\ &\quad \geq \text{delay}_{\text{ct}}\left(\text{et}_{\text{psC}^k}, \text{st}_{\text{psOptC}^{j+1}}\right), \\ &\therefore \text{et}_{\text{psC}^k} + d\text{min}\left(\text{atie}_{\text{psC}^k}, \text{atis}_{\text{psOptC}^{h+1}}\right) - \text{st}_{\text{psOptC}^{h+1}} \\ &\quad \geq \text{et}_{\text{psC}^k} + d\text{min}\left(\text{atie}_{\text{psC}^k}, \text{atis}_{\text{psOptC}^{j+1}}\right) \\ &\qquad - \text{st}_{\text{psOptC}^{j+1}}, \\ &\because \text{st}_{\text{psOptC}^{j+1}} < \text{et}_{\text{psC}^k} + d\text{min}\left(\text{atie}_{\text{psC}^k}, \text{atis}_{\text{psOptC}^{j+1}}\right), \\ &\therefore 0 \\ &\quad < \text{et}_{\text{psC}^k} + d\text{min}\left(\text{atie}_{\text{psC}^k}, \text{atis}_{\text{psOptC}^{j+1}}\right) \\ &\qquad - \text{st}_{\text{psOptC}^{j+1}} \\ &\quad \leq \text{et}_{\text{psC}^k} + d\text{min}\left(\text{atie}_{\text{psC}^k}, \text{atis}_{\text{psOptC}^{h+1}}\right) \\ &\qquad - \text{st}_{\text{psOptC}^{h+1}}, \\ &\therefore \text{st}_{\text{psOptC}^{h+1}} < \text{et}_{\text{psC}^k} + d\,\text{min}\left(\text{atie}_{\text{psC}^k}, \text{atis}_{\text{psOptC}^{h+1}}\right). \end{aligned} \tag{28}$$

Table 1: Parameters of the instances and satellite.

Target attribute		Maneuverability	
Number of targets	[30, 55]	Max maneuver speed	1°/s
Target distribution	Uniform distribution	Acceleration	$0.5^{\circ}/s^2$
Duration distribution	$N(15, 9)$	Deceleration	$0.25^{\circ}/s^2$
Profit distribution	$N(2 * \mathrm{dur}_i - 5{,}100)$	Tranquilization time	5 s

So there is no other h that could make the partial solution $\mathrm{psOptC}^{1,\ldots,h}$ of the current best solution psOptC be dominated by the current node psC. □

4. Experimental Study

In this section, the performance of the proposed B&B algorithm is evaluated. First, test data is designed according to the actual requirements of the AEOSs in China. And then, these data are employed to analyze the impact of different methods on the performance of the B&B algorithm.

4.1. Test Data. The examples of China's agile satellite platform are used to validate the B&B algorithm. The maneuver process includes acceleration, uniform maneuver, and deceleration. Tranquilization time is required to reach the pointing stability allowing for good image quality as well. Maneuvers of 45° could be completed in about 60 seconds. Overall, the maneuver time is highly dependent on the rotation angle and axis. For good imaging quality, attitude is recommended to remain fixed during imaging, so the targets are all along-track strips. While the across-track tilt angle for nominal imaging acquisitions is ±30°, the satellite could be figured to achieve ±45° off-nadir pointing capability.

In general, an Earth observation satellite has an imaging time of 40 minutes in one track, and the targets are uniform distributed in both sides of the imaging track. In application, the time of the scheduling algorithm is demanded to be less than 10 minutes. In order to test the performance of the algorithm, the number of the targets ranges from 30 to 55. And other parameters such as the duration time and tranquilization time are in Table 1. The parameters for instance generation are listed in Table 1 where the left part of the table associates with scenario attributes and the right part associates with maneuverability. The number of targets ranges from 30 to 55, and targets are uniform distributed in both sides of the imaging track which means each target has an imaging time window. The duration of the targets is a Gauss distribution with a mean value of 15 and a variance of 9. The profit of the targets is a Gauss distribution whose mean value is a function of the duration; this means a more time-consuming target may have a higher profit. The satellite has a maximum maneuver speed at 1°/s, an acceleration speed at $0.5^{\circ}/s^2$, and a deceleration speed at $0.25^{\circ}/s^2$. A tranquilization time of 5 s is needed after maneuver.

A total number of 260 instances were used to test our B&B algorithm, where, for each of the 26 different numbers of targets (from 30 to 55, see Table 1), 10 random instances were generated. The B&B algorithm is coded in MATLAB 2014a and run on a computer with a 2.4 GHz i5 CPU and 3.79 GB RAM.

4.2. Results. This subsection analyzes the effect of different mechanisms, such as initialization solution, segment domination based pruning strategy, and best solution updating method, on the efficiency of the algorithm at different study scales. As the B&B algorithm will always find the optimal solution of the problem, we use the time cost as a measure to evaluate the efficiency of an algorithm.

The effects of different mechanisms on the algorithm efficiency are analyzed by comparing the B&B algorithm with the incomplete algorithms which lack a single mechanism. In the experiment, 10 experiments are randomly generated for each study scale. And then, the average time cost and the maximum time cost are used to analyze the effects of different mechanisms on the efficiency of the algorithm.

Three incomplete algorithms, noninitialization algorithm, nonpruning algorithm, and nonupdating algorithm, are designed for each mechanism.

The noninitialization algorithm, which is referred to as "nonInit," means that the algorithm has no initial solution before searching; that is, the current best solution is empty and the current best profit is 0 at the beginning of the search. However, the algorithm has the pruning method and best solution updating method corresponding to Sections 3.4, 3.5, and 3.6.

The nonpruning algorithm, which is referred to as "nonPrune," only adopts the pruning strategy in Sections 3.4 and 3.6, which prune a branch once the branch is found to be inconsistent, and does not use the pruning strategy SDBPS designed in Section 3.5. The initial solution of the algorithm is generated by the LACM of Section 3.3. And the SRBLBUM in Section 3.6.2 is used.

The nonupdating algorithm, which is referred to as "nonUpdate," is that the algorithm does not adopt the SRBLBUM of Section 3.6.2. The initial solution of the algorithm is generated by the LACM of Section 3.3, and the pruning strategies of Sections 3.4 and 3.5 are adopted.

The time cost statistics of the examples are shown in Figure 8 and Table 2. The left column of Table 2 is the number of targets of the examples. The statistical data of the nonpruning algorithm is only shown in the table because the time cost of the algorithm is too large to test all the examples. However, the available data in Table 2 are sufficient to analyze the effect of the pruning mechanism on the solution efficiency. Figure 8(a) shows the average time cost of different algorithms under different scale cases; Figure 8(b) shows the maximum time cost. By comparing different incomplete

Table 2: The time costs of different algorithms.

Number of targets	Average time cost				Maximum time cost			
	NonPrune	NonIntial	NonUpdate	TDTCBB	NonPrune	NonInt	NonUpdate	TDTCBB
30	509.13	11.09	8.09	7.57	1529.64	23.67	22.31	22.26
31	1035.66	10.17	6.02	5.82	6856.14	20.38	14.48	13.19
32	1388.77	16.96	7.89	7.21	5212.03	25.59	17.12	13.65
33	879.92	16.77	11.04	9.53	1796.53	32.07	27.22	22.58
34	963.38	19.89	10.73	10.06	4515.09	47.50	26.71	21.62
35	6244.71	42.94	24.32	22.78	31498.4	157.88	63.27	56.17
36	—	41.69	23.85	22.35	—	98.36	54.02	44.21
37	—	28.37	20.18	15.01	—	43.52	57.85	23.92
38	—	33.26	18.92	18.55	—	44.95	34.48	30.50
39	—	47.54	19.16	19.10	—	117.78	50.05	49.90
40	—	55.58	26.14	22.41	—	95.87	52.14	44.07
41	—	58.15	36.96	31.23	—	124.96	91.81	66.03
42	—	104.20	60.50	54.53	—	331.95	179.80	152.06
43	—	95.43	33.39	32.31	—	150.99	46.79	45.63
44	—	84.07	33.05	31.58	—	133.42	58.17	58.13
45	—	275.32	107.78	106.79	—	1423.46	565.39	538.09
46	—	136.24	59.92	52.38	—	397.62	191.27	185.32
47	—	146.85	43.02	40.44	—	384.70	73.44	72.85
48	—	110.02	41.82	40.58	—	326.78	63.88	61.91
49	—	162.07	60.65	51.04	—	564.89	121.81	90.99
50	—	204.68	84.44	69.05	—	370.86	172.90	135.38
51	—	228.01	128.46	110.05	—	702.79	382.00	389.69
52	—	357.41	188.84	163.81	—	617.71	659.73	587.63
53	—	443.14	189.53	147.82	—	1005.12	898.90	495.71
54	—	503.18	168.74	149.32	—	1740.40	446.83	440.71
55	—	364.66	141.32	114.38	—	975.19	386.53	253.74

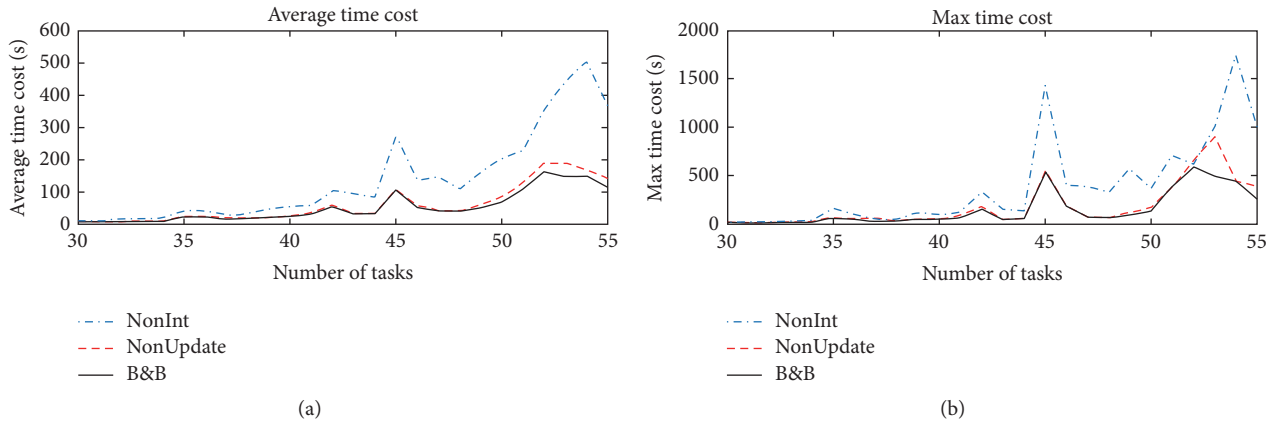


Figure 8: The time cost of different algorithms.

algorithms and B&B algorithms, we can analyze the influence degree of different mechanisms.

The average time cost of the B&B algorithm is less than 150 s when there are 55 targets, which indicates that the algorithm has a high efficiency. Experiments show that the algorithm can effectively deal with temporal constraints composed of the time-dependent maneuvering and imaging time windows. In the extreme case of the experiments, the maximum time cost of the B&B algorithm is 587.6 s, not more than 10 min. This indicates that the B&B algorithm meets the

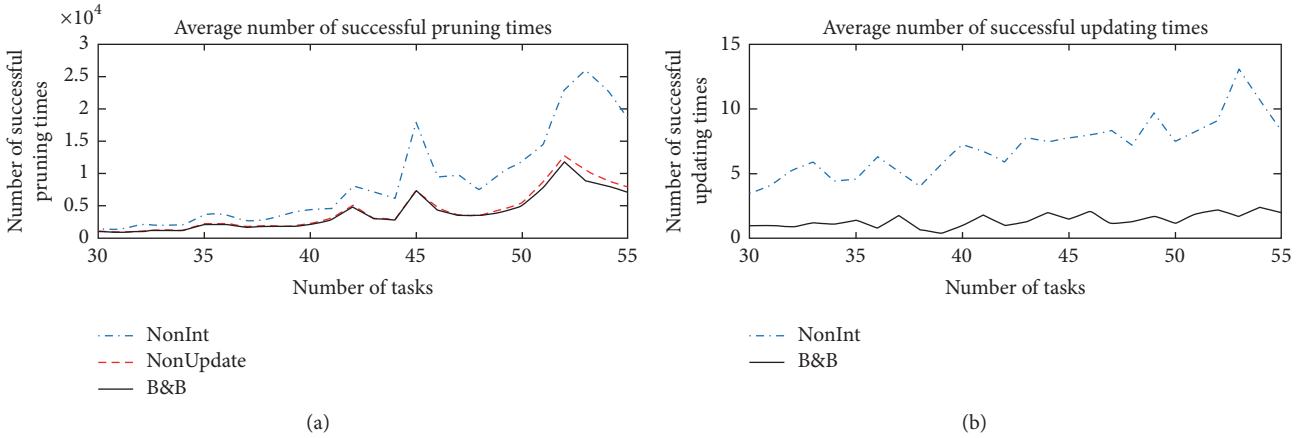


Figure 9: The number of successful pruning and updating attempts.

needs of the use of the agile satellites, and the algorithm could find the optimal solution of a problem that is about twice the size of the actual demand. The effects of the different mechanisms are analyzed in the following.

The time costs of all the algorithms become larger with the increase of the size of the examples, but the growth rates of each algorithm are not the same, which shows that the robustness of each algorithm to the degree of difficulty is not the same. It can be seen from Figure 8 that the B&B algorithm grows the slowest, followed by nonupdating algorithm, noninitialization algorithm, and nonpruning algorithm. This phenomenon indicates that the interaction of various solving strategies can help to increase the stability of the algorithm to difficult problems.

It can be seen from Table 2 that the time cost of the nonpruning algorithm rises a lot with the increase of the size of the examples. When the size of the example reaches 35, the average time of the nonpruning algorithm has reached 6244.71 s, but the average time cost of the B&B algorithm is only 22.78 s. This means that the pruning strategy SDBPS in Section 3.5 can greatly improve the efficiency of the algorithm.

The change tendency of the time cost of the noninitialization algorithm is similar to that of the B&B algorithm. The difference between the two average time costs also rises with the increase of the example size. This shows that as the size of the example increases, the role of the initialization method increases. When the study size is around 55 targets, the time cost of the noninitialization algorithm is about two times that of the B&B algorithm; that is, the optimal solution time cost increases by about 219% in the case of no initialization method. This verifies that it could accelerate the algorithm where a good lower bound is provided at the beginning of the algorithm.

When the number of targets is less than 50, nonupdating algorithm and the B&B algorithm are almost the same. When the scale of the problem is larger than 50, the time difference between the two is gradually increased. According to the analysis of the latter, we can see that the number of successful updating times is not a lot, but there is no small effect on the searching efficiency, and the time cost will increase by 20% when the SRBLBUM in Section 3.6.2 is not adopted.

With the increase of the difficulty of the problem, the difference of the time cost between different algorithms becomes larger, which means the effect of each mechanism on the solution efficiency is increasing when the problem is hard to solve. The B&B algorithm has the best performance under different scale targets, followed by nonupdating algorithm, noninitialization algorithm, and nonpruning algorithm. So the SDBPS in Section 3.5 contributes the most, followed by the LACM in Section 3.3 and finally the SRBLBUM in Section 3.6.2.

The difference of the maximum time cost between the incomplete algorithm and the B&B algorithm is larger than the difference of the average time cost between the algorithms, indicating that the B&B algorithm has better stability. When the time cost of the B&B algorithm is large, the difference between each incomplete algorithm and the B&B algorithm is also large, which indicates that the effect of each mechanism is more obvious in the face of difficult problems.

To further analyze the effect of SDBPS and SRBLBUM on the efficiency of the solution, we also count the number of successes of various methods in different algorithms as shown in Figure 9.

The number of successful SDBPS increases with the increase of the size of the problem, and the change trend is similar to the time cost. The more complex the problem is, the greater the dependence of the pruning strategy is, which verifies that the pruning strategy SDBPS plays a more important role in complex problems in the preceding paper. The number of successful SDBPS of the noninitialization algorithm is the highest, which is about 25000 when there are 53 targets, followed by the nonupdating algorithm and the B&B algorithm. This could be caused by the pruning position of the B&B algorithm that is closer to the root node. Further explanation could be that the other strategies are helpful to pruning earlier, and thus the searching efficiency is improved.

The number of successful SDBPS and SRBLBUM is higher in the noninitialization algorithm, which means that SDBPS and SRBLBUM play a more important role in accelerating the

optimal solution search. The number of successful SRBLBUM in the B&B algorithm is not large but has a slowly rising trend. Combined with the solution time, we can tell that it brings a good effect to the improvement of the searching efficiency, although there is only a small amount of successful SRBLBUM. When the problem is complex, SRBLBUM could save about 20% of the searching time.

5. Conclusion

This paper considers a new AEOSS problem with time-dependent temporal constraints which was not investigated in the literature. To address this new problem, we proposed a highly efficient branch and bound (B&B) algorithm to find its optimal solution. The proposed B&B algorithm uses a look-ahead construction method to generate a high quality initial lower bound at the very beginning. It is also equipped with three complementary pruning strategies, namely, the symmetry breaking based pruning strategy, the segment domination based pruning strategy, and the bound based pruning strategy. The combined use of these three pruning strategies greatly reduces the search space and accelerates the search process. To validate the efficacy of the proposed algorithm, we conducted empirical experiments on a set of instances that are closely related to the real-life scenario. Computational results disclosed that the proposed B&B is highly efficient which meets engineering demand very well. Indeed, it is able to solve instances with 55 targets using only 164 seconds on average and 588 seconds in maximum. Additional experiments were also conducted to evaluate the effectiveness of three algorithm ingredients: the look-ahead construction method (LACM), the segment domination based pruning strategy (SDBPS), and the segment replacement based lower bound updating method (SRBLBUM, which is used within the bound based pruning strategy). The results revealed that all of the three ingredients contribute to improve the performance of the B&B algorithm, and the most contributive one is SRBLBUM, followed by LACM and finally SDBPS.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant nos. 71690233, 61473301, and 61603400).

References

[1] M. Lemaître, G. Verfaillie, F. Jouhaud, J.-M. Lachiver, and N. Bataille, "Selecting and scheduling observations of agile satellites," *Aerospace Science and Technology*, vol. 6, no. 5, pp. 367–381, 2002.

[2] B. Dilkina and B. Havens, *Agile Satellite Scheduling via Permutation Search with Constraint Propagation*, 2005.

[3] N. Bianchessi, *Planning and Scheduling Problems for Earth Observing Satellites: Models and Algorithms*, Universita degli Studi di Milano, Milano, Italy, 2006.

[4] N. Bianchessi, J.-F. Cordeau, J. Desrosiers, G. Laporte, and V. Raymond, "A heuristic for the multi-satellite, multi-orbit and multi-user management of Earth observation satellites," *European Journal of Operational Research*, vol. 177, no. 2, pp. 750–762, 2007.

[5] D. Habet and M. Vasquez, "Saturated and consistent neighborhood for selecting and scheduling photographs of agile earth observing satellite," in *Proceedings of the Metaheuristics International Conference*, 2003.

[6] D. Habet, M. Vasquez, and Y. Vimont, "Bounding the optimum for the problem of scheduling the photographs of an agile earth observing satellite," *Computational Optimization and Applications*, vol. 47, no. 2, pp. 307–333, 2010.

[7] D. Habet, "Tabu search to solve real-life combinatorial optimization problems: a case of study," *Studies in Computational Intelligence*, vol. 203, pp. 129–151, 2009.

[8] S. D. Florio, *Performances Optimization of Remote Sensing Satellite Constellations: A Heuristic Method*, 2006, Performances optimization of remote sensing satellite constellations: a heuristic method.

[9] P. Wang and G. Reinelt, "A heuristic for an earth observing satellite constellation scheduling problem with download considerations," *Electronic Notes in Discrete Mathematics*, vol. 36, pp. 711–718, 2010.

[10] S. Kananub, T. Rukkwamsuk, and P. Arunvipas, "Agile earth observing satellites mission scheduling based on decomposition optimization algorithm," in *Computer Integrated Manufacturing Systems*, vol. 19, pp. 127–136, Integrated Manufacturing Systems, 2013.

[11] P. Tangpattanakul, N. Jozefowiez, and P. Lopez, "A multi-objective local search heuristic for scheduling Earth observations taken by an agile satellite," *European Journal of Operational Research*, vol. 245, no. 2, pp. 542–554, 2015.

[12] R. Xu, H. P. Chen, X. L. Liang, and H. M. Wang, "Priority-based constructive algorithms for scheduling agile earth observation satellites with total priority maximization," *Expert Systems with Applications*, vol. 51, pp. 195–206, 2016.

[13] R. Grasset-Bourdel, Interaction between action and motion planning for an agile earth-observing satellite,.

[14] R. Grasset-Bourdel and G. Verfaillie, "Action and motion planning for agile Earth-observation satellites," *ICOPS2011*, 2011.

[15] A. Globus, J. Crawford, J. Lohn, and A. Pryor, "A comparison of techniques for scheduling earth observing," in *Proceedings of the satellites. National Conference on Artificial Intelligence*, pp. 836–843, San Jose , California, Usa, 2004.

[16] D.-Y. Liao and Y.-T. Yang, "Imaging order scheduling of an earth observation satellite," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 37, no. 5, pp. 794–802, 2007.

[17] G. Wu, J. Liu, M. Ma, and D. Qiu, "A two-phase scheduling method with the consideration of task clustering for earth observing satellites," *Computers & Operations Research*, vol. 40, no. 7, pp. 1884–1894, 2013.

[18] T. C. E. Chenga and B. M. T. Linb, "A concise survey of scheduling with time-dependent processing times," *European Journal of Operational Research*, vol. 152, no. 1, p. 13, 2004.

[19] S. Gawiejnowicz, *Time-Dependent Scheduling*, Monographs in Theoretical Computer Science. An EATCS Series, Springer, Berlin, Germany, 2008.

[20] C. Pralet and G. Verfaillie, "Time-dependent simple temporal networks: Properties and algorithms," *RAIRO - Operations Research*, vol. 47, no. 2, pp. 173–198, 2013.

[21] A. Maillard, *Flexible Scheduling for Agile Earth Observing Satellites. Space Physics[physics.space-ph]*, Institut supérieur de l'Aéronautique et de l'Espace (ISAE), 2015.

[22] S. Tonetti, S. Cornara, and F. Pirondini, "Fully automated mission planning and capacity analysis tool for the DEIMOS-2 agile satellite," in *Proceedings of the 13th International Conference on Space Operations, SpaceOps 2014*, usa, May 2014.

[23] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem," *Journal of the Operations Research Society of America*, vol. 2, no. 4, pp. 393–410, 1954.

[24] P. Hentenryck, Y. Deville, and C.-M. Teng, "A generic arc-consistency algorithm and its specializations," *Artificial Intelligence*, vol. 57, no. 2-3, pp. 291–321, 1992.