Expectation and SAA models and algorithms for scheduling of multiple earth observation satellites under the impact of clouds

Jianjiang Wang[a,b], Erik Demeulemeester[b], Xuejun Hu[c,*], Guohua Wu[d]

[a] *College of Systems Engineering, National University of Defense Technology, Changsha, China*
[b] *Research Center for Operations Management, Department of Decision Sciences and Information Management, Faculty of Economics and Business, KU Leuven, Belgium*
[c] *Business School, Hunan University, Changsha, China*
[d] *School of Traffic and Transportation Engineering, Central South University, Changsha, China*
Email: jianjiangwang@nudt.edu.cn, erik.demeulemeester@kuleuven.be, xuejun_hu@hnu.edu.cn, guohuawu@csu.edu.cn

Abstract

In the explosively increased number of applications of earth observation satellites (EOSs), scheduling is a significative issue to satisfy more requests and obtain a high observation efficiency. This paper investigates the scheduling of multiple EOSs under the impact of clouds. Firstly, we formulate the presences of clouds as stochastic events, and propose an expectation model. Afterwards, for the first time, a branch-and-price algorithm based on Dantzig-Wolfe decomposition is devised to solve the model optimally and efficiently. In order to obtain initial columns for column generation, a dynamic programming algorithm and a heuristic algorithm are suggested, respectively. Furthermore, we discuss the impact of clouds in the case of joint probabilities, and establish a sample average approximation (SAA) model accordingly. With respect to the expectation model, numerical experiment results demonstrate the relative dominance of the proposed branch-and-price algorithm in terms of solution time compared to CPLEX. In addition, the solution of the proposed SAA model is proven to be more robust than that of the expectation model.

Keywords: scheduling; earth observation satellites; cloud coverage; expectation model; branch-and-price; sample average approximation

1. Introduction

Earth observation satellites (EOSs) are the platforms equipped with sensors that orbit the earth to take photographs of special areas at the request of users [5, 13]. EOSs can image, while moving along their orbits, which is shown in Fig.

1. After capturing the photographs, the acquired data will be stored in the onboard memory and transferred to a ground station when the satellites are in the feasible transferring range. Most EOSs operate at low altitudes with the orbital periods being dozens of minutes or several hours. However, it takes several days for a single EOS to view the whole area of the Earth. Hence, multi-satellite collaboration has been applied extensively in order to accelerate the response to users.

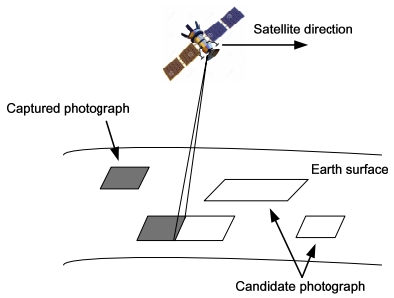


Figure 1: The satellite captures the photographs [32]

Because of some unique advantages, e.g. an expansive coverage area, long-term surveillance, accurate and effective information access and unlimited airspace borders, EOSs have been extensively employed in earth resources exploration, nature disaster surveillance, crop monitoring, etc. With the development of space science and technology, the number of satellites increases continuously. However, satellites are still scarce in comparison with the explosive growth of applications. Hence, scheduling is a significative issue to satisfy more requests and obtain a high observation efficiency.

Up to now, a great number of studies of EOS scheduling have been proposed, and some of them focus on mathematical programming and exact algorithms. Chen et al. [11] defined and analyzed the constraints of EOS scheduling in detail, and formulated the problem as a mixed integer linear programming model. In order to verify the feasibility and effectiveness, they applied the model in several real-world situations. Benoist and Rottembourg [3] formulated the EOS scheduling as a generalized prize collecting traveling salesman problem with time windows, and developed a compact mathematical model. Using valid inequalities based on task interval reasoning, the upper bounds for revenue maximization were improved. Gabrel et al. [16, 17] constructed mathematical programming models based on the directed acyclic graph (DAG) formulation, and some decomposition techniques as well as dynamic programming were employed to get the optimal solutions. In order to maximize the number of observed images, Nag et al. [30] proposed a control framework that includes orbital mechanics, attitude control and scheduling optimization for agile Cubesat constellations. In detail, a mixed integer programming model was suggested for scheduling optimization, and a dynamic programming algorithm with heuristics was developed. Furthermore, the exact solution algorithms based on mathematical programming also contain branch-and-bound [12], dynamic programming [21, 22],

Lagrangian relaxation [4, 49] and Constraint programming [35].

In addition, a large number of metaheuristics were developed for EOS scheduling. Cordeau and Cordeau [13] described the problem as the selection of a subset of requests with maximal profit for a given orbit, and solved the problem using a tabu search heuristic. As an extension of [13], Bianchessi et al. [5] employed the tabu search algorithm for EOS scheduling in the multi-satellite, multi-orbit and multi-user case. Besides, an upper bounding procedure based on column generation was proposed for evaluation. Furthermore, Habet et al. [19] proposed a tabu search algorithm to solve the EOS scheduling problem. Besides, a dynamic programming algorithm was designed to evaluate the performance of the tabu search algorithm. Considering the observation efficiency and fairness of resource sharing, Tangpattanakul et al. [32] established a multi-objective combinatorial optimization model, and presented an indicator-based multi-objective local search algorithm. Liu et al. [27] analyzed the setup time constraints of satellite scheduling, and formulated it as a time-dependent scheduling problem. Based on six removal operators and three insertion operators, an adaptive large neighborhood search (ALNS) algorithm was developed. Considering task merging for EOS observation, Wu et al. [41] proposed a simulated annealing algorithm based on the adaptive neighborhood operators. Moreover, metaheuristics for EOS scheduling also include tabu search algorithms [20, 26, 34, 35, 46], genetic algorithms [23, 31, 39, 42], ant colony algorithms [10, 24, 40, 45], local search algorithms [21, 22] and simulated annealing algorithms [18, 20].

With regard to heuristics, Lemaître et al. [22] proposed greedy algorithms to get feasible solutions for EOS scheduling problems. For the scheduling of COSMO-SkyMed satellite constellation, Bianchessi and Righini [6] proposed a constructive heuristic algorithm with look-ahead and back-tracking capabilities, which can produce feasible schedules in a short time. Wolfe and Sorensen [39] formulated EOS scheduling as a window-constrained packing problem. A fast and simple priority dispatch method was suggested, and the look-ahead mechanism was introduced to improve the performance of solutions. Xu et al. [43] developed a mathematical programming model for the scheduling of EOSs, and designed constructive algorithms to solve the problem, which adopt a priority-based sequential construction procedure to avoid conflicts and to generate feasible solutions. For the other constructive and heuristic algorithms, readers refer to [4, 20, 26, 29].

All the previous studies focus on the deterministic scheduling without considering the impact of clouds. However, in practice, EOS observations are significantly affected by the presences of clouds, since most EOSs are equipped with optical sensors that cannot see through clouds [18]. For instance, around 80% of the observations with the currently operational optical SPOT satellites are useless due to the presences of clouds [1]. Lin et al. [26] formulated the presences of clouds as a set of covered time windows, and forbade the tasks to be observed in the covered time windows. In practice, the drawback and infeasibility of Lin's approach is that there exist a lot of uncertainties of clouds, which are always changing over time and cannot be forecasted deterministically [1, 21]. Hence, the uncertainties of clouds bring much more difficulties for EOS

scheduling. Liao and Tang [25] considered the uncertainties of clouds, formulated the presence of clouds for each observation window as a stochastic event, and established a model with the objective of maximizing the weighted sum of a function of the profits and the expected number of executed tasks. Valicka et al. [36] formulated the scheduling of EOSs under uncertainties of clouds as two-stage and three-stage stochastic mixed integer programming models, respectively, with the objective of maximizing expected collection quality across a set of scenarios. In [1], the online scheduling of a Pleiades satellite that is equipped with a cloud detection instrument was considered, and the decisions were made on board based on the detection results of clouds.

Based on the principle of Dantzig-Wolfe decomposition, column generation algorithms have been proven to be one of the most successful approaches for solving linear programs or for getting bounds for integer programmes. Currently, column generation and branch-and-price have been successfully used in many fields [7, 8, 9, 14, 44]. However, with respect to EOS scheduling, the studies of column generation are still very limited. The column generation technique has been invoked in the deterministic EOS scheduling to provide a better upper bound [17, 28] and to evaluate the feasible solutions derived from some heuristics [5].

In this study, considering the impact of clouds, we formulate the presences of clouds for observations as stochastic events. In addition, an expectation model is proposed to formulate the scheduling of multiple EOSs. Due to the fact that the expectation model is characterized by a block diagonal structure, we decompose it into a set-packing master problem and some subproblems using Dantzig-Wolfe decomposition, and develop a branch-and-price algorithm to solve the model. Furthermore, relaxing the independence assumptions, the coverage of clouds in the case of joint probabilities is discussed, and a sample average approximation (SAA) model is constructed taking into account scheduling each task to multiple resources. By numerous simulation experiments, we prove that the branch-and-price algorithm can solve the expectation model optimally and efficiently. In addition, the solution of the SAA model is proven to be more robust than that of the expectation model.

The remainder of this paper is organized as follows. In the next section we describe the problem in detail, and formulate the problem with an expectation model. In Section 3, we present a branch-and-price algorithm to solve the expectation model. In Section **??**, we discuss the impact of clouds in the case of joint probabilities. Numerical results of our approaches are presented in Section 5. The last section offers conclusions and directions for future research.

2. The EOS scheduling problem

In this study we focus on the scheduling of multiple EOSs in which the presences of clouds for observations are formulated as stochastic events. Furthermore, a mathematical expectation model is suggested to describe the problem.

Table 1: Notations

T	set of tasks, $T = \{1, ..., n\}$
i, j	task index, $i, j \in T \cup \{\text{s,t}\}$, in which s, t are dummy tasks
ω_i	profit of task i, $i \in T$
O	set of orbits, $O = \{1, ..., m\}$
k	orbit index, $k \in O$
b_{ik}	$b_{ik} = 1$ if orbit k is available for the observation of task i, otherwise $b_{ik} = 0$, $i \in T, k \in O$
M_k, E_k	memory capacity and energy capacity of orbit k, $k \in O$
m_k, e_k	memory and energy consumption for each unit time of observation of orbit k, $k \in O$
$[ws_{ik}, we_{ik}]$	imaging period of observation of task i on orbit k, $i \in T, k \in O$
θ_{ik}	slewing angle of observation of task i on orbit k, $i \in T, k \in O$
st_{ij}^k	setup time between task i and task j on orbit k, $i, j \in T, k \in O$
ρ_{ij}^k	energy consumption for slewing between task i and task j on orbit k, $i, j \in T, k \in O$
$\tilde{\lambda}_{ik}$	binary stochastic variable, $\tilde{\lambda}_{ik} = 1$ denotes that task i can be successfully observed on orbit k, otherwise $\tilde{\lambda}_{ik} = 0$, $i \in T, k \in O$
p_{ik}	probability that task i will be successfully observed on orbit k, $i \in T, k \in O$

2.1. Problem description

In this paper, we focus on the scheduling of targets (circles with limited dimensions) rather than polygons that cover a wide geographical area. Besides, the orbits of satellites are formulated as resources. Hence, there will be at most one observation window for each task (target) on each resource. Some notations of this study are summarized in Table 1. Let T be the set of tasks (strips) submitted by users and let O be the set of orbits within the scheduling horizon. With each task $i \in T$ is associated a profit ω_i. Each orbit $k \in O$ is associated with a memory capacity M_k, an energy capacity E_k, a memory consumption for each unit of observation time m_k and an energy consumption for each unit of observation time e_k. Let $b_{ik} = 1$ denote that task i can be observed on orbit k, otherwise $b_{ik} = 0$. $[ws_{ik}, we_{ik}]$ denotes the imaging period for task i on orbit k, and θ_{ik} represents the slewing angle. Many of these notions are illustrated in Fig. 2. In this work, we only consider non-agile satellites that have the maneuverability of rolling (slewing) which indicates a movement that is perpendicular to the direction of the orbit, without the maneuverability of pitching which indicates a movement along the direction of the orbit. Hence, the imaging periods for observations are fixed without flexibility, such that the start and finish time of task i on orbit k will be fixed as $[ws_{ik}, we_{ik}]$, and the duration can be calculated as $we_{ik} - ws_{ik}$.

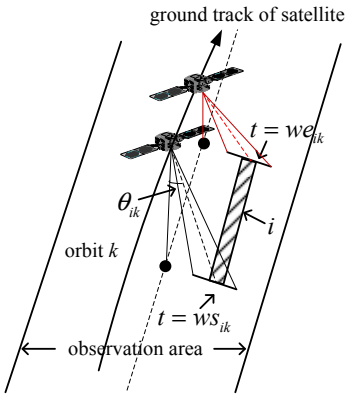


Figure 2: Imaging period for observation

After observing a task, the satellite requires a sequence of transformation operations to observe the next one, which contain sensor shutdown→ slewing→ attitude stability→ startup. Hence, there should be sufficient setup time between two consecutive tasks, and the required setup time can be calculated by the following formula [37]:

$$st_{ij}^{k} = sd_k + |\theta_{ik} - \theta_{jk}|/s_k + as_k + su_k, \tag{1}$$

where st_{ij}^{k} is the setup time between task i and task j on orbit k, and sd_k, as_k and su_k are the times of sensor shutdown, attitude stability and startup of orbit k, respectively. Besides, s_k is the slewing velocity of orbit k, and θ_{ik} and θ_{jk} are the slewing angles of tasks i and j on orbit k, respectively.

For observing task i on orbit k, the memory consumption can be computed by $(we_{ik}-ws_{ik})m_k$. Differently from memory, energy will not only be consumed by observation, but also by sensor slewing. The energy consumption for observing task i on orbit k is $(we_{ik}-ws_{ik})e_k$. Let ρ_{ij}^k denote the energy consumption of slewing between consecutive tasks i and j on orbit k, which can be calculated by the formula below:

$$\rho_{ij}^k = |\theta_{ik} - \theta_{jk}|\pi_k, \tag{2}$$

where π_k is the energy consumption for each unit slewing angle on orbit k.

Considering the impact of clouds, we formulate the presences of clouds for observations as stochastic events, denoted by 0-1 bernoulli random variables $\tilde{\lambda}_{ik}, i \in T, k \in O$. $\tilde{\lambda}_{ik} = 1$ if the observation of task i on orbit k can be successfully observed without the presences of clouds, otherwise $\tilde{\lambda}_{ik} = 0$. Let p_{ik} denote the probability for a successful observation of task i on orbit k, i.e., no presences of clouds, thus we can obtain $p\{\tilde{\lambda}_{ik} = 1\} = p_{ik}$ and $p\{\tilde{\lambda}_{ik} = 0\} = 1 - p_{ik}$. In our assumption, the probabilities of cloud coverage are provided by meteorological departments [50].

2.2. Expectation model

In this study, a mathematical expectation model maximizing the expected profits of successful observations is established. In this model, we use binary decision variables $x_{ij}^k \in \{0,1\}$ $(i,j \in T \cup \{s,t\}, k \in O)$, in which s,t are dummy tasks for starting and terminating, respectively. $x_{ij}^k = 1$ if both tasks i,j are scheduled on orbit k, and task i is the immediate predecessor of task j; otherwise $x_{ij}^k = 0$. Hence, the mathematical expectation model is given below:

$$\max \sum_{i \in T} \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \sum_{k \in O} \omega_i \cdot p_{ik} \cdot x_{ij}^k \tag{3}$$

subject to

$$\sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \sum_{k \in O} x_{ij}^k \leq 1, \forall i \in T \tag{4}$$

$$\sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} x_{ij}^k = \sum_{\substack{j \in T \cup \{s\} \\ j \neq i}} x_{ji}^k, \ \forall i \in T, k \in O \tag{5}$$

$$\sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} x_{ij}^k \leq b_{ik}, \forall i \in T, k \in O \tag{6}$$

$$x_{ij}^k(ws_{jk} - we_{ik} - st_{ij}^k) \geq 0, \forall i,j \in T, k \in O \tag{7}$$

$$\sum_{i \in T} \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} x_{ij}^k(we_{ik} - ws_{ik})m_k \leq M_k, \forall k \in O \tag{8}$$

$$\sum_{i\in T}\sum_{\substack{j\in T\cup\{t\}\\ j\neq i}} x_{ij}^k(we_{ik}-ws_{ik})e_k + \sum_{i\in T}\sum_{\substack{j\in T\\ j\neq i}} x_{ij}^k\rho_{ij}^k \leq E_k, \forall k\in O \tag{9}$$

$$x_{ij}^k \in \{0,1\}, \forall i,j\in T\cup\{s,t\}, k\in O \tag{10}$$

The objective (3) is to maximize the expected value of the profits of the executed tasks under the impact of clouds. The set of constraints (4) guarantees that each task will be observed at most once. Constraints (5) are flow balance constraints that force the number of predecessors to be equal to the number of successors for each task. Constraints (6) enforce that each task can only be scheduled to the orbits that are available for it. There must be sufficient setup times between consecutive tasks for transformations, which is enforced in constraints (7). Apparently, constraints (7) also guarantee that the imaging periods for different tasks on the same orbit cannot overlap. Constraints (8) check that the memory consumption of the scheduled tasks cannot exceed the memory capacity for each orbit. Constraints (9) compute the energy consumption of the task sequence for each orbit, and enforce that the energy consumption must be less than or equal to the capacity.

3. A branch-and-price algorithm

Note that, the proposed expectation model is characterized by a block diagonal structure, which facilitates decomposition. Hence, to solve the expectation model more efficiently, we reformulate it as a set packing (master) problem and some subproblems using Dantzig-Wolfe decomposition.

3.1. Set packing model

To formulate the set packing problem, we define the following notations:

R_k: the set of all feasible solutions (schedules) for orbit k, $k\in O$.

r: a feasible solution, $r\in R_k$, $k\in O$.

φ_{ijrk}: $\varphi_{ijrk}=1$ if task i is the immediate predecessor of task j of solution r on orbit k, otherwise $\varphi_{ijrk}=0$.

c_{kr}: the expected scheduling profits of feasible solution r on orbit k, $r\in R_k$, $k\in O$.

Decision variables:

z_{kr}: $z_{kr}=1$ if feasible solution r is selected for orbit k, $r\in R_k$, $k\in O$, otherwise $z_{kr}=0$.

The set packing model for the scheduling of multiple EOSs under the impact of clouds can be formulated as follows:

$$\max \sum_{k\in O}\sum_{r\in R_k} c_{kr}z_{kr} \tag{11}$$

subject to

$$\sum_{k \in O} \sum_{r \in R_k} \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \varphi_{ijrk} z_{kr} \leq 1, \forall i \in T \tag{12}$$

$$\sum_{r \in R_k} z_{kr} = 1, \forall k \in O \tag{13}$$

$$z_{kr} \in \{0, 1\}, \forall r \in R_k, k \in O \tag{14}$$

The objective (11) is to maximize the expected value of the profits of executed tasks. Constraints (12) are corresponding to constraints (4), which enforce each task to be executed at most once. Constraints (13) are the convexity constraints representing that a feasible solution should be selected for each orbit. Explicitly, the above formulation only take the unicity constraints into account for EOS scheduling. However, the remaining constraints are also involved implicitly to constitute the feasible solutions for each orbit (see Section 3.4).

It should be noted that the above set packing model decreases the number of constraints compared to the original expectation model. However, the number of feasible solutions for each orbit grows exponentially with the problem size. Hence, in order to avoid the "explosion" of the solution time, we intend to solve the linear programming (LP) relaxation of the above set packing problem using column generation.

3.2. Column generation

In essence, column generation is an iterative procedure that starts by solving the problem using a subset of all feasible solutions (columns), which is the so-called Restricted Master Problem (RMP). Then the RMP is solved to optimality. In the next step, in the subproblems the dual variables are used to price out the absent columns that can improve the objective. If one or multiple promising columns are identified (i.e., columns with a positive reduced cost for our problem), the column(s) will be added to the RMP and the RMP is reoptimized. Then, the procedure will terminate if we cannot find any columns to improve the objective (e.g. the reduced costs of all absent columns are negative).

In the first iteration, we solve the RMP using a subset of columns R'_k, $R'_k \subseteq R_k$ for each orbit $k, k \in O$, in which the subset R'_k are provided by algorithms in Section 3.3. Thereafter, for each successive iteration, the following dual variables are passed to the subproblems for identifying feasible columns with positive reduced costs:

- μ_i: dual variables corresponding to constraints (12),
- δ_k: dual variables corresponding to constraints (13).

On the basis of the dual variables from the RMP, we can theoretically calculate the reduced cost for each absent column, and add the columns with positive reduced costs to the RMP. However, due to the large number of columns, it is impractical and much too time consuming to enumerate all absent columns. Hence, we transfer the problem to an optimization problem that searches for the column with the most positive reduced cost.

3.3. Initial feasible solutions

In this paper, in order to obtain the initial feasible solutions R'_k for each orbit k, we first define a directed acyclic graph (DAG) $G^k = (V^k, A^k)$ for each orbit k. Note that, the problems can be solved separately for each orbit. Hence, without provoking ambiguity, we drop the superscript k. Using a task-on-node representation, the nodes in V represent the tasks that are available, plus two special nodes $\{s, t\}$ representing the dummy starting and terminating tasks.

A is the set of arcs, which is defined as below:

- $\forall j \in V \cup \{t\}$, $(s, j) \in A$;
- $\forall j \in V \cup \{s\}$, $(j, t) \in A$;
- $\forall i, j \in V$, $(i, j) \in A$ iff task j can be observed after task i, i.e. the setup time between tasks i and j is sufficient.

It is obvious that a path from the starting node s to the terminating node t that satisfies the memory and energy constraints represents a feasible solution. Based on the above formulation, we devise two heuristic algorithms to generate the initial feasible solutions for each orbit. In addition, the time complexities of the two initialization algorithms are analyzed theoretically in Theorems 1 and 2, respectively.

The first algorithm is a simple labeling-based path construction heuristic subject to capacity checking. First, for each node j, a weight ω'_j, $\omega'_j = \omega_j p_j$ is assigned. For each node j, the state $(i_k, \uparrow l_{p'}, \Omega_p, M'_p, E'_p)$ represents a path from the starting node s to j, namely $p = \{s, i_1, \ldots, i_k, j\}$. In detail, i_k is the immediate predecessor of j of p, $\uparrow l_{p'}$ is a pointer referring to the subpath $p' = \{s, i_1, \ldots, i_k\}$, Ω_p is the sum of weights of the nodes of the path, and M'_p, E'_p are the memory consumption and energy consumption of path p. Besides, $N(j)$ is defined as the number of labels in $P(j)$. In order to prevent the quantity explosion of paths, and to satisfy the computer memory limit, the number of paths for each node is limited to less than a predefined number L, i.e. $N(j) \leq L$. The first initilization algorithm is described in **Algorithm 1**, in which $\Gamma^{-1}(j)$ is the set of all predecessors of j.

By this algorithm, a predefined number L of feasible solutions of maximum weight will be obtained as the initial feasible solutions. In addition, in order to guarantee that the master problem is feasible, we add an "empty" solution that is corresponding to the direct path from s to t for each orbit.

Theorem 1. The time complexity of **Algorithm 1** is $O(n^2L^2)$.

Proof. The maximum number of labels (subpaths) in $P(j)$ is L. Hence, the computational complexity of the inner loop (lines 5-9) is $O(L)$. Subsequently, the maximum number of predecessors for each node is n, and thus the complexity of lines 4-10 is $O(nL)$. Hence, the maximum number of paths in $P(j)$ should be nL. In addition, the complexity of selecting L paths from $P(j)$ with maximum weights (line 11) is nL^2. In conclusion, the time complexity of **Algorithm 1** is $O(n * (nL + nL^2)) = O(n^2L^2)$.

Algorithm 1 Initialization Algorithm I

1: $P(s) \leftarrow \{[null, null, 0, 0, 0]\}$
2: $P(j) \leftarrow \emptyset$,$N(j) \leftarrow 0$ $(j \leftarrow 1, \ldots, n, t)$
3: **for** $j \leftarrow 1, \ldots, n, t$ **do**
4: **for** all $i \in \Gamma^{-1}(j)$ **do**
5: **for** all $l_p \in P(i)$ **do**
6: **if** memory and energy constraints are satisfied **then**
7: $P(j) \leftarrow P(j) \cup \{[i, \uparrow l_p, \Omega_p + \omega'_j, M'_p + m(we_j - ws_j), E'_p + e(we_j - ws_j) + \rho_{ij}]\}$;
8: **end if**
9: **end for**
10: **end for**
11: Select L paths from $P(j)$ with maximum weights and delete the other paths from $P(j)$;
12: **end for**

With respect to **Initialization Algorithm II**, we generate L paths from node s to node t heuristically. The algorithm starts from the dummy terminate node t, and then selects a predecessor i based on preference random sampling. The priority rule is to choose the node with the highest weight $\omega_i^{'}$. If the memory constraints and energy constraints are satisfied, we move to node i, otherwise another predecessor will be reselected. Afterwards, the above procedures are repeated until the dummy starting node s is reached, which implies that a feasible solution has been obtained. Finally, the above steps are repeated L times, and L solutions will be produced for each orbit. In this algorithm, the probability of each predecessor i being selected is $p_i = \omega_i^{'} / \Sigma_{s \in \Gamma^{-1}(j)} \omega_s^{'}$. The heuristic initialization algorithm is outlined in **Algorithm 2**.

Algorithm 2 Initialization Algorithm II

1: $COL_k \leftarrow \emptyset$;
2: **for** $l = 1, \ldots, L$ **do**
3: $j \leftarrow t$ // t is the terminate node of orbit k;
4: **while** $j \neq s$ **do**
5: Select node i, $i \in \Gamma^{-1}(j)$ with biased random sampling, in which $\Gamma^{-1}(j)$ is the set of all predecessors of j. The priority value of node i is the sum of the profit of node i and the profits of all predecessors of node i, and the priority rule is to select the predecessor with the highest priority value.
6: **if** Memory constraints and energy constraints are satisfied **then**
7: $j \leftarrow i$;
8: **end if**
9: **end while**
10: Obtain the column col_{kl}, $COL_k \leftarrow COL_k \cup \{col_{kl}\}$;
11: **end for**

Theorem 2. The time complexity of **Algorithm 2** is $O(n^2 L)$.

Proof. The complexity of selecting a predecessor with biased random sampling (line 5) is $O(n)$. In addition, the maximum number of while loops (lines

4-9) should be n, and thus it will consume $O(n^2)$ time to obtain a feasible solution. Therefore, the time complexity of **Algorithm 2** is $O(L * n^2) = O(n^2 L)$.

Notably, the time complexity of **Algorithm 2** is lower than that of **Algorithm 1**, thus it will consume less time in obtaining initial solutions, which will also be proven in Section 5.

3.4. Subproblem

In each iteration of column generation, we solve m subproblems, one for each orbit k, $k \in O$. In each subproblem, the objective is to find the feasible solution with the most positive reduced cost. Hence, the objective of the subproblem for orbit k, $k \in O$ is outlined as below:

$$\max_{r \in R_k} \{c_{kr} - \sum_{i \in T} \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \varphi_{ijrk} \mu_i - \delta_k\} \tag{15}$$

Note that the index k can be removed, since each subproblem is solved separately. Therefore, the above objective function can be rewritten as:

$$\max_{r \in R} \{c_r - \sum_{i \in T} \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \alpha_{ijr} \mu_i - \delta\} \tag{16}$$

In which $c_r = \sum_{i \in T} \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \omega_i p_i \alpha_{ijr}$, and the index r can also be neglected for the optimization problem, thus the objective is:

$$\max \sum_{i \in T} \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \alpha_{ij} (\omega_i p_i - \mu_i) - \delta \tag{17}$$

Parameters. The additional parameters employed in the subproblem are:
μ_i, δ: the dual variables obtained from the restricted master problem, $i \in T$.
b_i: $b_i = 1$ if it is available to observe task i, otherwise $b_i = 0$, $i \in T$.
M, E: memory capacity and energy capacity.
m, e: memory and energy consumption for each unit time of observation.
$[ws_i, we_i]$: imaging period of observation of task i, $i \in T$.
st_{ij}: setup time between task i and task j, $i, j \in T$.
ρ_{ij}: energy consumption for slewing between task i and task j, $i, j \in T$.
p_i: probability of successful execution of task i, $i \in T$.
Decision variables. The decision variables are:
α_{ij}: $\alpha_{ij} = 1$ if both tasks i, j are scheduled, and task i is the immediate predecessor of task j; otherwise $\alpha_{ij} = 0$.
Subproblem formulation. The subproblem can be formulated as follows:

$$\max \sum_{i \in T} \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \alpha_{ij} (\omega_i p_i - \mu_i) - \delta \tag{18}$$

subject to

$$\sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \alpha_{ij} = \sum_{\substack{j \in T \cup \{s\} \\ j \neq i}} \alpha_{ji}, \ \forall i \in T \tag{19}$$

$$\sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \alpha_{ij} \leq b_i, \forall i \in T \tag{20}$$

$$\alpha_{ij}(ws_j - we_i - st_{ij}) \geq 0, \forall i, j \in T \tag{21}$$

$$\sum_{i \in T} \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \alpha_{ij}(we_i - ws_i)m \leq M \tag{22}$$

$$\sum_{i \in T} \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \alpha_{ij}(we_i - ws_i)e + \sum_{i \in T} \sum_{\substack{j \in T \\ j \neq i}} \alpha_{ij}\rho_{ij} \leq E \tag{23}$$

$$\alpha_{ij} \in \{0, 1\}, \forall i, j \in T \cup \{s, t\} \tag{24}$$

Constraints (19)-(24) are corresponding to constraints (5)-(10) of the original problem, respectively. Since the subproblem is solved separately for each orbit, the complexity of the problem is significantly reduced compared to the original problem.

Based on the above DAG formulation, the subproblem is to search for a path from s to t with the maximum weight respecting the memory and energy constraints, which in essence is a constrained longest weighted path planning problem. Notably, at each iteration of column generation, the weight of each node i should be updated with the dual variables from the RMP, namely $\omega_i' = \omega_i p_i - \mu_i$. On the basis of [16], the subproblem can be solved using a forward-checking dynamic programming algorithm that is composed of labels and dominance rules. In detail, labels are employed to record the state information after a node is accessed, and dominance rules are employed to remove the labels that cannot produce the optimal solution. In this study, the definition of label is the same as that of Section 3.3, and the dominance rules are illustrated as below.

Definition 1: Path domination. From the starting node s to node j, there are two feasible paths p, q, i.e., $l_p, l_q \in P(j)$, path p is dominated by path q if and only if $\Omega_p \leq \Omega_q$, $M_p' \geq M_q'$, $E_p' \geq E_q'$, and at least one strict inequality holds.

3.5. Branch-and-bound

In order to guarantee the integrity, a branch-and-price algorithm, which is a combination of column generation and branch-and-bound, is required to solve the problem. With respect to the branch-and-price algorithm, branching is an important issue that is different from the classical branch-and-bound algorithm. Normally, direct branching strategies on the column variables of the RMP are thought to be inappropriate, because it could cause a significant alteration to the subproblem and yield an unbalanced branch-and-bound tree [2, 33].

In this study, we plan to use the branching on the variables of the original problem. For each subproblem (orbit) k, if there exist fractional variables, there

will be at least two variables that are fractional, say z_{kr} and z_{ks}, due to the convexity constraints (13). Hence, for the two columns corresponding to the fractional variables, if $\sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \varphi_{ijrk} \neq \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \alpha_{ijsk}$ for task $i, i \in T$, we will branch on task i, i.e., set $\sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} x_{ij}^k = 0$ or $\sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} x_{ij}^k = 1$. The main advantage of this branching scheme is that it does not destroy the structure of the subproblem, because the resulting modifications simply entail amending the weight of the corresponding node in the DAG. For instance, if $\sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} x_{ij}^k$ is set to 1, the corresponding weight ω_i' is set to Q for orbit k; otherwise if $\sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} x_{ij}^k$ is set to 0, ω_i' is set to $-Q$ for orbit k (Q is a big constant that is larger than the total profits of all the tasks). The second advantage is the fact that this branching strategy yields a balanced branch-and-bound tree.

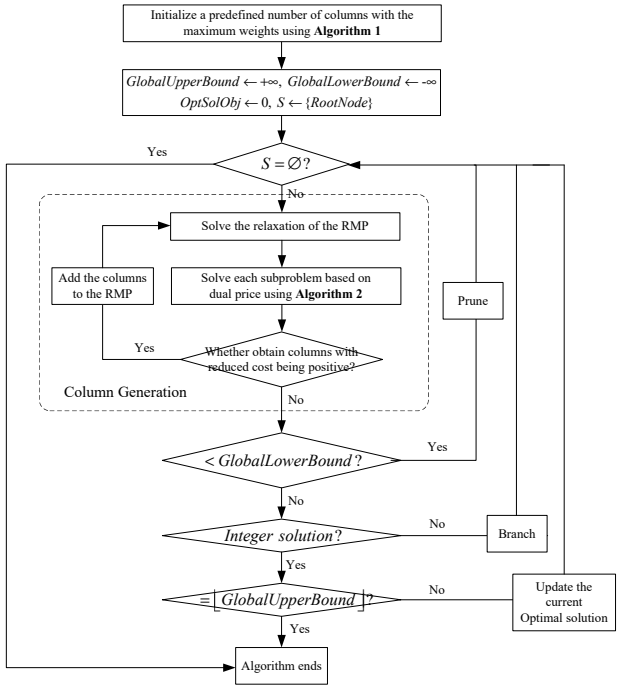


Figure 3: Branch-and-price algorithm

In **Fig. 3**, an overview of the branch-and-price algorithm is given.

In Eq.(**??**), N is the number of tasks, and K_i represents the number of orbits that are available for observing task $i, i = 1, ..., N$, i.e. the number of orbits on which there is at least one observation window for task i. In addition, $\alpha_{ik} = 1$ if task i can be successfully observed on orbit k without cloud coverage, otherwise $\alpha_{ik} = 0$. Moreover, α represents one of the possible values of stochastic vector $\tilde{\lambda}_{ik}, \forall i \in 1, ..., N, \forall k \in 1, ..., K_i$. Note that, in the case of joint probabilities, the probability of successful observation for task i on orbit k should be the following marginal probability:

$$p_{ik} = p\{\tilde{\lambda}_{ik} = 1\} = \sum_{\alpha_{ik}=1} p_\alpha \tag{26}$$

Apparently, if the probabilities of successful observations are replaced by the above marginal probabilities, our model (3)-(10) is also available for the case of joint probabilities.

It should be noted that if a task can be allocated to multiple orbits under the uncertainties of clouds, the probability of successful observation for this task will be higher. Therefore, in order to improve the scheduling efficiency, we will relax the unicity constraints (4) in the case of joint probabilities. Due to the possible multiple allocations of each task, the scheduling model becomes more complicated (nonlinear & nonconvex) [38]. In order to facilitate the solution, the problem is reformulated by introducing some auxiliary variables.

Firstly, the objective of scheduling is still to maximize the profits of the accomplished tasks.

$$\text{maximize} \sum_{i \in T} \omega_i \cdot \phi_i, \tag{27}$$

where $\phi_i \in \{0, 1\}, i \in T$ are auxiliary decision variables. $\phi_i = 1$ if task i can be successfully executed and otherwise $\phi_i = 0$. Besides, the variables $\phi_i, i \in T$ should also satisfy the following constraints:

$$\phi_{ik} \leq \phi_i, \forall i \in T, k \in O, \tag{28}$$

$$\phi_i \leq \sum_{k \in O} \phi_{ik}, \forall i \in T, \tag{29}$$

$$\phi_{ik} \leq \tilde{\lambda}_{ik}, \forall i \in T, k \in O, \tag{30}$$

$$\phi_{ik} \leq \sum_{\substack{j \in T \cup \{t\} \\ j \neq i,}} x_{ij}^k, \forall i \in T, k \in O \tag{31}$$

$$\phi_i, \phi_{ik} \in \{0, 1\}, \forall i \in T, k \in O. \tag{32}$$

With regard to the above constraints, x_{ij}^k and $\tilde{\lambda}_{ik}$ have been defined previously in this article. $\phi_{ik} \in \{0, 1\}, i \in T, k \in O$ are also auxiliary variables. $\phi_{ik} = 1$ if task i can be successfully executed on orbit k, and otherwise $\phi_{ik} = 0$. Constraints (28) imply that if task i can be successfully executed on any one orbit k, it means the task can be successfully executed. Constraints (29) enforce that there is at least one orbit k on which the task can be successfully executed, i.e. $\phi_{ik} = 1$, in order to make the task be successfully accomplished, i.e. $\phi_i = 1$. In addition, two sets of constraints have to be satisfied for the successful observation of task i on orbit k. First, the observation of task i on orbit k cannot be blocked by clouds, which is enforced in constraints (30). Second, task i must be allocated to orbit k, which is enforced in constraints (31).

Hence, in the case of joint probabilities, our scheduling model can be reformulated as: (27)-(32), (5)-(10). Note that there are random variables $\tilde{\lambda}_{ik}$ in the model, and thus the model cannot be solved directly. In this study, for the sake of solution, we reformulate the problem by sample average approximation (SAA).

Let W be a sample of scenarios of the random vector $w(\tilde{\lambda}_{ik}), i \in T, k \in O$, such that $W = \{w_1, w_2, ..., w_N\}$, in which N is the sample size. The problem can be reformulated as follows:

$$\text{maximize} \frac{1}{N} \sum_{w_l \in W} \sum_{i \in T} \omega_i \cdot \phi_{il} \tag{33}$$

subject to

$$\phi_{ilk} \leq \phi_{il}, i \in T, k \in O, w_l \in W \tag{34}$$

$$\phi_{il} \leq \sum_{k \in O} \phi_{ilk}, i \in T, w_l \in W \tag{35}$$

$$\phi_{ilk} \leq \lambda_{ik}^l, i \in T, k \in O, w_l \in W \tag{36}$$

$$\phi_{ilk} \leq \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} x_{ij}^k, i \in T, k \in O, w_l \in W \tag{37}$$

$$\phi_{il}, \phi_{ilk} \in \{0, 1\}, i \in T, k \in O, w_l \in W \tag{38}$$

Table 2: Parameters of satellites

satellite	slewing velocity (deg/sec)	power on time (sec)	power off time (sec)	stability time (sec)	memory /sec	energy /sec	energy /deg
CBERS-2	2	5	8	3	2	1.5	1.5
IKONOS-2	2.5	8	5	6	4	2.5	4
SPOT-5	3	10	10	9	3	3.5	1

In the above model, both $\phi_{il} \in \{0,1\}, i \in T, w_l \in W$ and $\phi_{ilk} \in \{0,1\}, i \in T, k \in O, w_l \in W$ are auxiliary decision variables. $\phi_{il} = 1$ if task i can be successfully accomplished under scenario w_l, and otherwise $\phi_{il} = 0$. $\phi_{ilk} = 1$ if task i can be successfully accomplished on orbit k under scenario w_l, and otherwise $\phi_{ilk} = 0$. Besides, λ_{ik}^l is the realization of the stochastic variable $\tilde{\lambda}_{ik}$ under scenario w_l.

More specifically, the objective (33) is to maximize the average profits of executed tasks on the sample W. Constraints (34)-(37) are similar to constraints (28)-(31) , which implies that the corresponding constraints have to be respected for each scenario. Surely, the decision variables $x_{ij}^k, i, j \in T \cup \{s,t\}, k \in O$ also have to satisfy the constraints (5)-(10). Apparently, the above model of the SAA problem is an integer linear model that can be solved directly by CPLEX.

5. Computational results

In order to verify the effectiveness and efficiency of the proposed models and algorithms, the tasks are randomly generated in the area: latitude 0°-60° and longitude 0°-150°. The profits of tasks are integers, uniformly distributed in the interval [1,10]. In this section, three different satellites are considered. The parameters of the satellites are outlined in Table 2, and the orbit models of the satellites are obtained from the Satellite Tool Kit (STK). In addition, the memory capacity and energy capacity of each orbit are randomly generated based on uniform distributions in the intervals [200,240] and [240,320], respectively. Considering the impact of clouds, for each imaging period of observation, the probability that there is no presences of clouds, i.e. the observation is successful, will be uniformly distributed in [0.5,1).

The algorithms in this study were implemented in C++ using the CPLEX 12.8 API and run on a personal computer equipped with an Intel(R) Core(TM) i3-2120M 3.29 GHz (4 processors) and 8 Gb RAM, with operating system Windows 7.

5.1. Performance evaluation of the branch-and-price algorithm

In this experiment, the number of tasks ranges from 40 to 400 with an increment of 40. The scheduling horizons are set to 12 and 24 hours, which are corresponding to 21 and 42 orbits, respectively. For each parameter setting, we create 10 problem instances randomly.

To verify the superiority of the branch-and-price algorithm, we compare its performance with that of CPLEX. Table 3 shows the comparison results for 21

Table 3: Performance evaluation of branch-and-price algorithm (21 orbits)

Algorithms	N_T	T_S/s	T_I/s	T_R/s	N_N	I_R	$LP\text{-}GAP(\%)$
B&P-IA1	40	**0.208**	**0.000**	**0.177**	**1.8**	**5.3**	**0.06**
	80	**0.477**	**0.000**	**0.220**	**5.4**	**7.5**	**0.34**
	120	**0.445**	**0.000**	**0.184**	**4.6**	**6.4**	**0.13**
	160	**1.831**	0.001	**0.467**	**12.2**	**13.7**	**0.14**
	200	**1.150**	0.002	**0.489**	**6.0**	**12.9**	**0.07**
	240	**0.444**	0.001	**0.292**	**2.4**	**7.4**	**0.06**
	280	**0.316**	0.001	**0.260**	**1.6**	**7.0**	**0.01**
	320	**0.404**	0.002	**0.280**	**2.4**	**7.4**	**0.05**
	360	**0.239**	0.002	**0.210**	**1.2**	**5.3**	**0.01**
	400	**0.331**	0.003	**0.288**	**1.2**	**7.1**	**0.01**
B&P-IA2	40	0.356	**0.000**	0.309	**1.8**	9.4	**0.06**
	80	0.670	**0.000**	0.322	5.6	11.3	**0.34**
	120	0.550	**0.000**	0.278	**4.6**	10.3	**0.13**
	160	2.400	**0.000**	0.572	16.8	16.8	**0.14**
	200	1.358	**0.000**	0.629	6.2	16.8	**0.07**
	240	0.618	**0.000**	0.455	**2.4**	10.9	**0.06**
	280	0.443	**0.000**	0.386	**1.6**	10.8	**0.01**
	320	0.480	**0.000**	0.351	**2.4**	10.3	**0.05**
	360	0.409	**0.000**	0.385	**1.2**	10.6	**0.01**
	400	0.408	**0.000**	0.391	**1.2**	10.4	**0.01**
CPLEX	40	0.346		0.310	**1.0**		1.99
	80	1.381		1.115	9.0		7.36
	120	2.576		1.944	8.2		10.29
	160	10.496		3.526	1734.5		7.57
	200	10.376		5.582	1004.2		8.80
	240	9.449		6.825	122.7		13.23
	280	13.830		9.650	129.9		12.91
	320	17.162		12.656	281.4		13.77
	360	26.090		15.349	1070.1		13.48
	400	29.899		19.492	724.9		13.38

orbits, in which column "N_T" represents the number of tasks. The definitions of the other columns are listed as below:

- T_S: average solution time for the 10 problem instances;
- T_I: average time for producing initial solutions;
- T_R: average solution time for the root node;
- N_N: average number of nodes in the branch-and-bound tree for solving;
- I_R: average iteration number of column generation for the root node;
- $LP\text{-}GAP$: average distance between the optimal integer solution and its linear relaxation upper bound, $LP\text{-}GAP = \frac{(UB_{LR}-OPT)}{OPT} \times 100$, in which OPT represents the optimal integer solution and UB_{LR} represents the optimal solution of the linear relaxation.

B&P-IA1 and B&P-IA2 represent the branch-and-price algorithms with the initialization algorithms being **Algorithm 1** and **Algorithm 2**, respectively. Besides, the bold numbers imply optimal performance compared to the other algorithms. As shown in Table 3, both our branch-and-price algorithm and CPLEX can solve all the problem instances to optimality. However, the solution time of the branch-and-price algorithm is much less than that of CPLEX. The solution time of B&P-IA1 is a bit less than that of B&P-IA2, because **initialization algorithm I** can obtain better initial solutions, reducing the number of exploration nodes (as shown in column N_N). Due to the efficient heuristic for initialization (**Algorithm 2**), the initialization time of B&P-IA2 is significantly less than that of B&P-IA1. The solution time of the root node of our branch-and-price algorithm is less than that of CPLEX (as shown in column T_R), since the column generation algorithm is much more efficient to solve the linear relaxation problem. As shown in the previous studies [47, 48], Dantzig-Wolfe reformulation and column generation can get tighter upper bounds for integer programming problems (as shown in column $LP\text{-}GAP$).

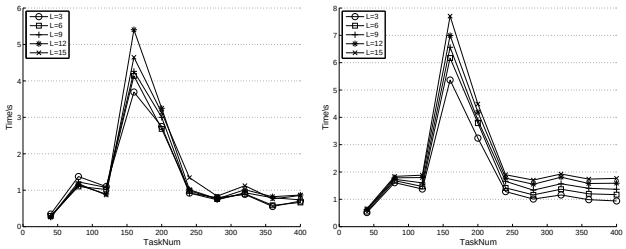
Table 4 describes the performance evaluation results for 42 orbits. Similarly to the results in Table 3, the solution performance of our branch-and-price algorithm is much better than that of CPLEX, with superiorities in solution time (as shown in column T_S), solution time of root node (as shown in column T_R) and the number of nodes (as shown in N_N). Furthermore, the Dantzig-Wolfe reformulation and column generation algorithm can obtain tighter upper bounds than CPLEX from solving the original linear relaxation problems. As mentioned above, the initialization time of B&P-IA2 is much less than that of B&P-IA1, but the derived initial solutions of B&P-IA2 are not as good as those of B&P-IA1. Hence, the iteration times of column generation, the solution time of root node and the total solution time of B&P-IA2 are longer than those of B&P-IA1, respectively.

Table 4: Performance evaluation of branch-and-price algorithm (42 orbits)

Algorithms	N_T	T_S/s	T_I/s	T_R/s	N_N	I_R	$LP\text{-}GAP(\%)$
B&P-IA1	40	**0.282**	**0.000**	**0.281**	**1.0**	**6.8**	**0.00**
	80	**0.806**	**0.000**	**0.692**	**1.6**	**15.5**	**0.02**
	120	**2.555**	**0.000**	**0.885**	**8.0**	**17.9**	**0.09**
	160	**10.176**	0.001	**0.943**	**44.4**	**17.3**	**0.15**
	200	**7.304**	0.001	**0.929**	**26.2**	**16.5**	**0.07**
	240	**3.984**	0.002	**1.056**	**12.0**	**16.8**	**0.08**
	280	**4.761**	0.002	**0.917**	21.2	**15.8**	**0.06**
	320	7.631	0.003	**0.959**	27.0	**15.9**	**0.07**
	360	**4.494**	0.004	**0.923**	16.6	**15.7**	**0.08**
	400	**2.737**	0.009	**0.915**	**8.4**	**14.3**	**0.05**
B&P-IA2	40	0.428	**0.000**	0.428	**1.0**	9.7	**0.00**
	80	1.087	**0.000**	0.915	**1.6**	18.4	**0.02**
	120	2.929	**0.000**	1.135	8.8	21.7	**0.09**
	160	12.224	**0.000**	1.164	48.0	20.7	**0.15**
	200	8.472	**0.000**	1.165	28.0	20.0	**0.07**
	240	5.710	**0.000**	1.387	13.6	21.0	**0.08**
	280	5.401	**0.000**	1.106	**20.4**	18.7	**0.06**
	320	**7.047**	**0.000**	1.304	**23.6**	20.8	**0.06**
	360	5.537	**0.000**	0.989	**16.2**	16.7	**0.08**
	400	4.277	**0.000**	1.187	13.4	18.4	**0.05**
CPLEX	40	0.594		0.588	**1.0**		0.91
	80	2.201		1.872	5.0		2.27
	120	6.708		3.730	503.1		5.13
	160	12.990		7.132	600.1		7.89
	200	53.739		12.672	3145.3		10.73
	240	64.307		17.620	2659.6		11.31
	280	73.269		23.355	2522.1		12.46
	320	113.555		30.517	2757.2		12.84
	360	191.654		42.304	5642.3		13.06
	400	281.025		56.976	5848.1		13.39

5.2. Performance analysis of the number of initial solutions

In addition to the initialization algorithms, the number of initial solutions also affects the performance of our branch-and-price algorithm. With regard to both B&P-IA1 and B&P-IA2, based on the problem instances in Section 5.1, we set the number of initial solutions for each orbit to 3, 6, 9, 12 and 15, respectively. The relevant experimental results are outlined in Figure 4 and Figure 5. Note that, although the number of initial solutions are different, both B&P-IA1 and B&P-IA2 can solve all the problem instances to optimality. Hence, the objective function values for each problem instance can be ignored, and we only need to analyze the impact of the number of initial solutions on the solution time.

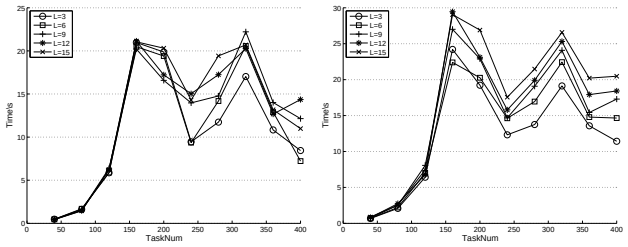


(a) B&P-IA1 (b) B&P-IA2

Figure 4: Performance analysis of the number of initial solutions (21 orbits)

For 21 orbits, when the number of tasks is relatively small (40-120), the solution time of B&P-IA1 decreases as the number of initial solutions increases (as shown in Figure 4(a)). This can be attributed to the fact that the iteration times of column generation decrease with an increase of the number of initial solutions, and thus the solution of root node is accelerated. When the number of tasks are larger (160-400), the solution time of B&P-IA1 increases when the number of initial solutions increases, because more time is required to obtain initial solutions. It is illustrated in Figure 4(b) that the solution time of B&P-IA2 increases when the number of initial solutions increases. This is because a heuristic is employed to obtain initial solutions, which cannot guarantee the qualities of initial solutions. Hence, the increase of the number of initial solutions only increases the sizes of master problems fruitlessly, and the iteration times of column generation cannot be reduced.

For 42 orbits, when the number of tasks is relatively small (40-120), the solution time of B&P-IA1 changes a little with the variances of the number of initial solutions. When the number of initial solutions is larger, the solution time of B&P-IA1 shows a small increment. As shown in Figure 5(b), when the number of tasks is small (40-120), the solution time of B&P-IA2 is stable with small variances. Afterwards, for more tasks (160-400), the solution time increases with the number of initial solutions.



(a) B&P-IA1 (b) B&P-IA2

Figure 5: Performance analysis of the number of initial solutions (42 orbits)

5.3. Performance evaluation of the SAA approach with joint probabilities

In this section, the number of tasks ranges from 20 to 100 with an increment of 20. The number of orbits are also 21 and 42, respectively. For each parameter setting, 10 problem instances are created randomly.Our SAA model (33)-(38), (5)-(10) is solved based on a small sample with the sample size being 100. Both the expectation model and the SAA model are solved by CPLEX, and the solution time is described in Table 5.

Table 5: **Solution time of the expectation model and the SAA model**

Task number	21 orbits		42 orbits	
	$t1$(sec)	$t2$(sec)	$t1$(sec)	$t2$(sec)
20	0.32	1.08	0.36	1.11
40	0.60	2.06	0.61	3.93
60	1.23	4.48	1.78	-
80	1.51	6.69	1.91	-
100	1.62	-	2.58	-

In Table 5, columns $t1$ and $t2$ represent the solution time of the expectation model (3)-(10) and the SAA model (33)-(38), (5)-(10), respectively, in which "-" denotes that the optimal solutions cannot be obtained by CPLEX because of the limited memory. From Table 5, it is observed that the solution time of the SAA model is longer than that of the expectation model, since more variables and constraints are introduced from sampling, making the problem more complicated.

In order to evaluate the performance of our SAA model, the solutions are also compared to those of the expectation problem. For the sake of comparison, the obtained solutions were tested on a large sample with the sample size being 10000.

Tables 6 and 7 show the comparison results for each problem instance, in which column "n" denotes the number of tasks, and column "No." contains the serial number of problem instances. Moreover, columns "MIN", "AVE" and "MAX" represent the minimum, the average, and the maximum scheduling profits obtained over all scenarios on the sample, respectively. It is worth noting that the bold numbers represent the larger profits between the two models. As

illustrated in Tables 6 and 7, almost all the minimum/average scheduling profits of the SAA model are larger than those of the expectation model. This is because we take into account scheduling each task to multiple orbits for the SAA model, which increases the scheduling robustness. However, for a minority of cases, the maximal profits of the SAA model are less than those of the expectation model, which can be attributed to the fact that the SAA model improves the performance of the worst-case solutions, but it cannot guarantee the optimality for all the scenarios.

In Tables 6 and 7, column "DEV" denotes the standard deviations of solutions on the large sample, where the italic numbers represent the smaller deviations between the two models. Apparently, the standard deviations of the SAA model are smaller than those of the expectation model for all the problem instances, which proves that the solutions of the SAA model are more robust and stable.

Although the SAA model outperforms the expectation model in robustness and stability, it is only solvable for small- to medium-scale problems. Hence, the SAA model is still far from large-scale practical application.

6. Conclusions and future work

In this paper, considering the impact of clouds, we formulated the presences of clouds as stochastic events, and then investigated the scheduling of multiple EOSs. Furthermore, an expectation model was proposed to formulate the problem. With respect to the expectation model, a branch-and-price algorithm was devised to solve the model optimally and efficiently. In addition, a SAA model was proposed in the case of joint probabilities. Finally, by a great number of simulation experiments, we proved that the branch-and-price algorithm can solve the problem optimally for all generated instances and is much faster than CPLEX. Besides, we also analyze the impacts of the numbers of initial solutions on the performances of the branch-and-price algorithm. Moreover, the SAA model is proven to be more robust by simulation experiments.

In the future, the first extension of our research is to consider the scheduling of agile satellites under the impact of clouds. Differently from the non-agile satellites in this study, the agile satellites do not only have the maneuverability of slewing, but also have the maneuverability of pitching, along with the orbit. Hence, the satellite will have a long time window for observation. Consequently, we need not only allocate the tasks to the orbits, but also need to decide the start and finish times. Moreover, in the future, considering the sparsity of observation windows for polar satellites, we will construct a more concise and more efficient model. In the new model, the redundant variables will be removed by analyzing the relationships between observation windows. Furthermore, with regard to the SAA problem, it is essential to design more efficient solution algorithms for large-scale instances, making the SAA model available in practice.

Table 6: Performance evaluation on the samples (21 orbits)

n	No.	Expectation model				SAA model			
		MIN	AVE	MAX	DEV	MIN	AVE	MAX	DEV
	0	27	66.60	89	9.81	**45**	**81.36**	**92**	*6.83*
	1	28	78.71	**108**	12.02	**57**	**92.91**	**108**	*8.49*
	2	32	80.23	**104**	11.74	**53**	**95.64**	**104**	*7.72*
	3	24	61.00	**83**	9.94	**45**	**74.54**	**83**	*6.34*
	4	21	54.07	71	8.24	**39**	**64.82**	**75**	*5.95*
20	5	20	59.73	**70**	7.23	**43**	**65.94**	**70**	*5.12*
	6	6	33.79	**49**	7.63	**9**	**36.57**	**49**	*6.90*
	7	23	66.25	**86**	9.29	**32**	**74.77**	**86**	*7.33*
	8	38	81.72	**104**	10.74	**57**	**91.26**	**104**	*8.12*
	9	26	74.32	**101**	11.33	**42**	**80.49**	**101**	*9.73*
	Ave	24.5	65.64	86.5	9.80	**42.2**	**75.83**	**87.2**	*7.25*
	0	62	109.00	139	10.83	**74**	**117.48**	**143**	*10.51*
	1	60	111.11	151	13.55	**93**	**132.73**	**154**	*9.62*
	2	97	164.27	213	16.42	**143**	**194.90**	**227**	*13.34*
	3	52	102.15	**128**	10.54	**65**	**107.02**	**128**	*9.54*
	4	61	110.18	**143**	11.33	**90**	**122.27**	139	*8.36*
40	5	109	171.93	214	15.13	**122**	**181.97**	**218**	*13.85*
	6	33	91.04	**127**	13.04	**57**	**103.91**	126	*10.12*
	7	61	113.86	**146**	12.01	**75**	**121.77**	**146**	*10.96*
	8	58	114.74	159	14.37	**91**	**135.85**	**160**	*10.93*
	9	65	117.45	**156**	13.30	**91**	**135.78**	**156**	*9.93*
	Ave	65.8	120.57	157.6	13.05	**90.1**	**135.37**	**159.7**	*10.72*
	0	100	172.03	228	17.24	**121**	**186.23**	**231**	*15.22*
	1	132	205.36	276	19.28	**170**	**233.06**	**284**	*17.21*
	2	127	201.91	257	17.64	**152**	**219.09**	**263**	*15.85*
	3	105	172.39	**217**	15.18	**122**	**175.04**	200	*11.32*
	4	110	180.72	**238**	16.83	**125**	**191.44**	237	*15.12*
60	5	91	164.70	218	16.76	**114**	**179.09**	**224**	*15.51*
	6	96	168.10	220	16.26	**136**	**199.58**	**230**	*12.41*
	7	83	146.39	195	15.87	**114**	**169.56**	**210**	*14.30*
	8	108	174.84	**227**	17.37	**132**	**185.60**	226	*14.07*
	9	115	191.68	253	18.67	**157**	**220.82**	**255**	*13.71*
	Ave	106.7	177.81	232.9	17.11	**134.3**	**195.95**	**236**	*14.47*
	0	153	236.76	295	18.48	**191**	**253.82**	**305**	*16.67*
	1	159	232.87	**303**	19.99	**174**	**240.38**	292	*16.48*
	2	154	218.68	**272**	16.69	**159**	**221.54**	270	*15.59*
	3	114	193.53	254	19.06	**152**	**218.98**	**267**	*16.18*
	4	110	185.70	245	17.28	**177**	**240.58**	**292**	*16.83*
80	5	123	190.86	245	17.38	**146**	**214.00**	**259**	*15.85*
	6	123	195.04	254	18.25	**155**	**221.05**	**266**	*15.54*
	7	137	199.07	254	17.18	**175**	**238.10**	**288**	*16.43*
	8	143	227.19	290	20.22	**168**	**241.32**	**303**	*18.86*
	9	123	211.60	280	20.02	**158**	**235.72**	**289**	*17.64*
	Ave	133.9	209.13	269.2	18.45	**165.5**	**232.55**	**283.1**	*16.61*

Table 7: Performance evaluation on the samples (42 orbits)

n	No.	Expectation model				SAA model			
		MIN	AVE	MAX	DEV	MIN	AVE	MAX	DEV
20	0	28	78.00	**115**	13.61	**54**	**95.84**	**115**	*9.95*
	1	40	83.96	**107**	10.47	**66**	**98.97**	**107**	*7.02*
	2	37	77.55	**94**	7.98	**75**	**93.09**	**94**	*2.34*
	3	34	83.82	**115**	12.58	**63**	**101.74**	**115**	*8.67*
	4	36	75.22	**94**	9.57	**60**	**87.89**	**94**	*5.83*
	5	26	65.69	**91**	9.78	**63**	**86.40**	**91**	*4.43*
	6	39	81.92	**108**	10.87	**79**	**104.62**	**108**	*4.47*
	7	23	71.13	**98**	10.99	**79**	**96.25**	**98**	*2.95*
	8	28	73.64	**102**	11.17	**78**	**98.82**	**102**	*3.82*
	9	23	71.65	**100**	11.58	**69**	**94.51**	**100**	*4.57*
	Ave	31.4	76.26	**102.4**	10.86	**68.6**	**95.81**	**102.4**	*5.40*
40	0	63	115.85	155	12.48	**116**	**150.34**	**162**	*6.88*
	1	88	149.46	190	14.17	**151**	**184.85**	**191**	*6.03*
	2	76	136.74	180	14.97	**139**	**174.40**	**188**	*7.88*
	3	89	149.62	192	14.08	**137**	**177.46**	**195**	*9.32*
	4	64	126.05	**172**	13.89	**123**	**160.10**	**172**	*8.11*
	5	112	179.86	236	17.76	**176**	**221.55**	**240**	*9.59*
	6	103	162.75	**211**	15.19	**150**	**191.03**	**211**	*9.93*
	7	123	187.34	238	16.37	**169**	**218.20**	**241**	*11.22*
	8	81	163.90	219	16.83	**174**	**212.49**	**224**	*8.25*
	9	75	139.09	**190**	15.60	**116**	**164.83**	188	*10.51*
	Ave	87.4	151.07	198.3	15.13	**145.1**	**185.53**	**201.2**	*8.77*

Acknowledgments

We thank the anonymous referees for their insightful comments. This research was supported by the Research Project of National University of Defense Technology (ZK18-03-16), the National Natural Science Foundation of China under Grant No. 71701067, and No. 71801218. It was also funded by the Natural Science Foundation of Hunan Province, China under Grant No. 2019JJ50039.

References

[1] Beaumet, G., Verfaillie, G., & Charmeau, M.C. (2011). Feasibility of autonomous decision making on board an agile earth-observing satellite. *Computation Intelligence*, *27*(1), 123-139.

[2] Belien, J. (2006). Exact and heuristic methodologies for scheduling in hospitals: problems, formulations and algorithms. PhD thesis, Katholieke Universiteit Leuven, Belgium.

[3] Benoist, T., & Rottembourg, B. (2004). Upper bounds for revenue maximization in a satellite scheduling problem. *4OR - A Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, *2*(3), 235-249.

[4] Bianchessi, N., & Righini, G. (2006). A mathematical programming algorithm for planning and scheduling an earth observing SAR constellation. In *Proceedings of the 5th international workshop on planning and scheduling for space.*

[5] Bianchessi, N., Cordeau, J.F., Desrosiers, J., Laporte, G., & Raymond, V. (2007). A heuristic for the multi-satellite, multi-orbit and multi-user management of earth observation satellites. *European Journal of Operational Research*, *177*(2), 750-762.

[6] Bianchessi, N., & Righini, G. (2008). Planning and scheduling algorithms for the COSMO-SkyMed constellation. *Aerospace Science Technology*, *12*(7), 535-544.

[7] Boyer, V., Gendron, B., & Rousseau, L.M. (2014). A branch-and-price algorithm for the multi-activity multi-task shift scheduling problem. *Journal of Scheduling*, *17*(2), 185-197.

[8] Brunner, J.O., & Edenharter, G.M. (2011). Long term staff scheduling of physicians with different experience levels in hospitals using column generation. *Health Care Management Science*, *14*(2), 189-202.

[9] Brunner, J.O., & Bard, J.F. (2013). Flexible weekly tour scheduling for postal service workers using a branch and price. *Journal of Scheduling*, *16*(1), 129-149.

[10] Chen, X., Reinelt, G., Dai, G., & Wang, M. (2018). Priority-based and conflict-avoidance heuristics for multi-satellite scheduling. *Applied Soft Computing*, *69*, 177-191.

[11] Chen, X., Reinelt, G., Dai, G., & Spitz, A. (2019). A mixed integer linear programming model for multi-satellite scheduling. *European Journal of Operational Research*, *275*(2), 694-707.

[12] Chu, X., Chen, Y., & Tan, Y. (2017). An anytime branch and bound algorithm for agile earth observation satellite onboard scheduling. *Advances in Space Research*, *60*(9), 2077-2090.

[13] Cordeau, J., & Laporte, G. (2005). Maximizing the value of an earth observation satellite orbit. *Journal of the Operational Research Society*, *56*(8), 962-968.

[14] Dayarian, I., Crainic, T.G., Gendreau, M., & Rei, W. (2015). A branch-and-price approach for a multi-period vehicle routing problem. *Computers & Operations Research*, *55*, 167-184.

[15] Du, J.Y., Brunner, J.O., & Kolisch, R. (2014). Planning towing processes at airports more efficiently. *Transportation Research Part E: Logistics and Transportation Review*, *70*, 293-304.

[16] Gabrel, V., & Vanderpooten, D. (2002). Enumeration and interactive selection of efficient paths in a multiple criteria graph for scheduling an earth observing satellite. *European Journal of Operational Research*, *139*(3), 533-542.

[17] Gabrel, V., & Murat, C. (2003). Mathematical programming for earth observation satellite mission planning. In T.A. Ciriani, G. Fasano, S. Gliozzi, R. Tadei (Eds.), *Operations research in space and air* (pp. 103-122). Springer.

[18] Globus, A., Crawford, J., Lohn, J., & Pryor, A. (2003). A comparison of techniques for scheduling fleets of earth-observing. *Journal of the Operational Research Society*, *56*(8), 962-968.

[19] Habet, D., Vasquez, M., & Vimont, Y. (2010). Bounding the optimum for the problem of scheduling the photographs of an agile earth observing satellite. *Computational Optimization and Applications*, *47*(2), 307-333.

[20] Karapetyan, D., Minic, S.M., Malladi, K.T., & Punnen, A.P. (2015). Satellite downlink scheduling problem: a case study. *OMEGA International Journal of Management Science*, *53*, 115-123.

[21] Lemaître, M., Verfaillie, G., Jouhaud, F., Lachiver, J.M., & Bataille, N. (2000). How to manage the new generation of agile earth observation satellites. In *Proceedings of the international symposium on artificial intelligence, robotics and automation in space*.

[22] Lemaître, M., Verfaillie, G., Jouhaud, F., Lachiver, J.M., & Bataille, N. (2002). Selecting and scheduling observations of agile satellites. *Aerospace Science Technology*, *6*(5), 367-381.

[23] Li, J., Li, J., Chen, H., & Jing, N. (2014). A data transmission scheduling algorithm for rapid-response earth-observing operations. *Chinese Journal of Aeronautics*, *27*(2), 349-364.

[24] Li, Y., Wang, R., & Xu, M. (2014). Rescheduling of observing spacecraft using fuzzy neural network and ant colony algorithm. *Chinese Journal of Aeronautics*, *27*(3), 678-687.

[25] Liao, D., & Tang, Y. (2007). Imaging order scheduling of an earth observation satellite. *IEEE Transactions on Systems Man and Cybernetics Part C-Applications and Reviews*, *37*(5), 794-802.

[26] Lin, W., Liao, D., Liu, C., & Lee, Y. (2005). Daily Imaging Scheduling of an Earth Observation Satellite. *IEEE Transactions on Systems Man and Cybernetics Part A-Systems and Humans*, *35*(2), 213-223.

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

[28] Mancel, C., & Lopez, P. (2003). Complex optimization problems in space systems. In *Proceedings of the 13th international conference on automated planning and scheduling*.

[29] Mok, S. H., Jo, S., Bang, H., & Leeghim, H. (2018). Heuristic-Based Mission Planning for an Agile Earth Observation Satellite. *International Journal of Aeronautical and Space Sciences*, 1-11.

[30] Nag, S., Li, A.S.,& Merrick, J.H., (2018). Scheduling algorithms for rapid imaging using agile Cubesat constellations. *Advances in Space Research*, *61*(3), 891-913.

[31] Tangpattanakul, P., Jozefowiez, N., & Lopez, P. (2012) Multi-objective optimization for selecting and scheduling observations. In C.A. Coello, V. Cutello, K. Deb, S. Forrest, G. Nicosia, M. Pavone (Eds.), *Lecture Notes in Computer Science*. Springer.

[32] Tangpattanakul, P., Jozefowiez, N., & Lopez, P. (2015). A multi-objective local search heuristic for scheduling Earth observations taken by an agile satellite. *European Journal of Operational Research*, *245*(2), 542-554.

[33] Vanderbeck, F. (2000). On Dantzig-Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm. *Operations Research*, *48*(1), 111-128.

[34] Vasquez, M., & Hao, J. (2001). A logic-constrained knapsack formulation and a tabu algorithm for the daily photograph scheduling of an earth observation satellite. *Computational Optimization and Applications*, *20*(2), 137-157.

[35] Vasquez, M., & Hao, J. (2003). Upper bounds for the SPOT 5 daily photograph scheduling problem. *Journal of Combinatorial Optimization*, *7*(1), 87-103.

[36] Valicka, C. G., Garcia, D., Staid, A., Watson, J. P., Hackebeil, G., Rathinam, S., & Ntaimo, L. (2019). Mixed-integer programming models for optimal constellation scheduling given cloud cover uncertainty. *European Journal of Operational Research*, *275*(2), 431-445.

[37] Wang, J., Demeulemeester, E., & Qiu, D. (2016). A pure proactive scheduling algorithm for multiple earth observation satellites under uncertainties of clouds. *Computers & Operations Research*, *(*74), 1-13.

[38] Wang, J., Demeulemeester, E., Hu, X., Qiu, D., & Liu, J. (2018). Exact and Heuristic Scheduling Algorithms for Multiple Earth Observation Satellites Under Uncertainties of Clouds. *IEEE Systems Journal*.

[39] Wolfe, J., & Stephen, S.E. (2000). Three scheduling algorithms applied to the earth observing systems domain. *Management Science*, *46*(1), 148-168.

[40] Wu, G., Liu, J., Ma, M., & Qiu, D. (2013). A two-phase scheduling method with the consideration of task clustering for earth observing satellites. *Computers & Operations Research*, *40*(7), 1884-1894.

[41] Wu, G., Wang, H., Pedrycz, W., Li, H., & Wang, L. (2017). Satellite observation scheduling with a novel adaptive simulated annealing algorithm and a dynamic task clustering strategy. *Computers & Industrial Engineering*, 113, 576-588.

[42] Wu, K., Zhang, D., Chen, Z., Chen, J., & Shao, X. (2019). Multi-type multi-objective imaging scheduling method based on improved NSGA-III for satellite formation system. *Advances in Space Research*, *63*(8), 2551-2565.

[43] Xu, R., Chen, H., Liang, X., & Wang, H. (2016). Priority-based constructive algorithms for scheduling agile earth observation satellites with total priority maximization. *Expert Systems with Applications*, *51*, 195-206.

[44] Yu, Y., Wang, S., Wang, J., & Huang, M. (2019). A branch-and-price algorithm for the heterogeneous fleet green vehicle routing problem with time windows. *Transportation Research Part B: Methodological*, *122*, 511-527.

[45] Zhang, Z., Hu, F., & Zhang, N. (2018). Ant colony algorithm for satellite control resource scheduling problem. *Applied Intelligence*, 1-11.

[46] Zufferey, N., Amstutz, P., & Giaccari, P. (2008). Graph colouring approaches for a satellite range scheduling problem. *Journal of Scheduling*, *11*(4), 263-277.

[47] Manne, AS. (1958). Programming of economic lot sizes. *Management science*, *4*(2), 115-135.

[48] Degraeve, Z., & Jans, R. (2007). A new Dantzig-Wolfe reformulation and branch-and-price algorithm for the capacitated lot-sizing problem with setup times. *Operations research*, *55*(5), 909-920.

[49] Brown, N., Arguello, B., Nozick, L., & Xu, N. (2018). A Heuristic Approach to Satellite Range Scheduling With Bounds Using Lagrangian Relaxation. *IEEE Systems Journal*, *12*(4), 3828-3836.

[50] National Meteorological Information Center. http://data.cma.cn/site/index.html.