Exact and heuristic scheduling algorithms for multiple earth observation satellites under uncertainties of clouds

Jianjiang Wang[a,b], Erik Demeulemeester[b], Dishan Qiu[a], Xuejun Hu[b,c,*], Jin Liu[a]

[a] *Science and Technology on Information Systems Engineering Laboratory, 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*

Email: jianjiangwang@nudt.edu.cn, erik.demeulemeester@kuleuven.be, ds_qiu@sina.com, xuejun_hu@hnu.edu.cn, liujin229234@163.com

Abstract

Most earth observation satellites (EOSs) are equipped with optical sensors, which cannot see through clouds. Hence, many observations will be useless due to the presence of clouds. In this work, we study the scheduling problem of multiple EOSs under uncertainties of clouds. In order to improve the possibility of completing tasks, we take the scheduling of each task to multiple resources (orbits) into account and establish a novel non-linear mathematical model. To solve the problem efficiently, an exact algorithm based on enumeration is proposed, in which each subproblem is solved by path programming, and all the feasible solutions of subproblems are combined to solve the master problem. Furthermore, three heuristics are designed to solve the large-scale problems. From the experimental results on random samples, it is observed that the solutions of our model perform better than those of the previous studies. Besides, both our exact algorithm and a mixed-integer nonlinear programming solver - Couenne can solve our model optimally for small problems, but our algorithm is more efficient than Couenne. For large-scale problems, we reveal the strengths and weaknesses of the proposed heuristic algorithms while solving different instances of various sizes.

Keywords: earth observation satellites, uncertainties of clouds, nonlinear mixed-integer model, enumeration, heuristic

1. Introduction

Earth Observation Satellite (EOS) scheduling means to allocate the tasks submitted by users to satellites, making the schedule satisfy the operational constraints. Because of some special advantages, e.g. an expansive coverage area, long-term surveillance, a high frequency of repeated observations, accurate and effective information access and unlimited airspace borders, EOSs have been extensively employed in earth resources exploration, nature disaster surveillance, urban planning, crop monitoring, etc. Due to the explosively increased number of applications, the number of satellites, in spite of having increased quickly, is still too limited. Hence, it is nontrivial for EOS scheduling to achieve high observation effectiveness and efficiency.

Up to now, a great number of studies focusing on EOS scheduling have been proposed, in which the scheduling of non-agile EOSs was formulated and solved in different ways:

Mathematical programming: Without considering memory and energy constraints, Benoist et al. [4], Habet et al. [17, 18, 19] and Lemaître et al. [23] formulated the problem with mathematical programming models. In addition, mixed integer programming (MIP) models were constructed on the basis of a "flow variable" formulation [8, 9, 13, 14, 43], in which the precedence relations between tasks are well defined. Moreover, Lin et al. [27, 28, 29] and Marinelli et al. [30] proposed "time-indexed" formulations of EOS scheduling, and established integer programming models. Hall et al. [20] formulated the problem as a longest path problem with time windows, and set up an integer linear programming model.

Constraint satisfaction problem (CSP): Verfaillie et al. [38] formulated EOS scheduling as constraint satisfaction problems. Agnèse et al. [1], Bensana et al. [5] and Verfaillie et al. [39] proposed valued constraint satisfaction problem (VCSP) formulations for SPOT-5 satellite scheduling, without considering energy constraints.

Knapsack problem: Vasquez et al. [36, 37] formulated EOS scheduling as 0-1 knapsack problems that include large numbers of capacity, binary and ternary "logical" constraints. Specifically, the knapsack formulation is based on binary variables, each representing a photo camera: $x_i = (p, c)$ takes the value 1 if the photo p is taken by the camera c and 0

Table 1: Formulation methods

Categories	Content	references	general forms	pros	cons
Mathematical programming	• General mathematical programming; • "flow variable" formulation; • "time-indexed" formulation	[4, 8, 9, 13, 14, 17, 18, 19, 20, 23, 43]	• General mathematical programming: maxg_kG_k $\forall j \in [1,n]\ Y_{2j-1} + Y_{2j} \leq 1$ $\forall j \in [1,n]$ if $tw(j) > j$ then $(Y_{2j-1} = Y_{2tw(j)-1}$ and $Y_{2j} = Y_{2tw(j)})$ $\sum_{i\in[1,2n]} X_{0\rightarrow i} \leq 1 \sum_{i\in[1,2n]} X_{i\rightarrow 2n+1} \leq 1$ • "flow variable" formulation: $\varphi^c_{jk} = 1$ that denotes task j is the direct predecessor of task k for observation on camera c, and the objective function, constraints are defined, respectively; • "time-indexed" formulation: decision variables are defined as α_{it} and γ_{kit} in which $\alpha_{it} = 1$ if task i begins to be observed at time t and otherwise $\alpha_{it} = 0$, and γ_{kit} is a step function indicating the setup status of processing task k to task i at time t.	accurate; rigorous; directly solved by mature solvers and algorithms.	difficult to be obtained; vulnerable; any changes in parameters may invalidate the models
Constraint satisfaction problem	General CSP; Value CSP;	[1, 5, 38, 39]	$< X, D, C >$ in which $X = \{x_1, x_2, ..., x_n\}$ is the set of variables, $D = \{D(x_1), D(x_2), ..., D(x_n)\}$ is the set of domains and $D(x_i)$ represents the domain of variable x_i . Besides, $C = \{c_1, c_2, ..., c_e\}$ is the set of constraints.	fast solution process	not guaranteed to be optimal
Knapsack problem	0-1 knapsack problem; Window-Constrained Packing Problem	[36, 37, 44]	• 0-1 knapsack problem: $x_i = (p, c)$ takes the value 1 if the photo p is taken by the camera c and 0 otherwise. • Window-Constrained Packing Problem: n jobs $\{job_i \mid i = 1, ..., n\}$ and job_i must be done within its opportunity window $[ws_i, we_i]$. $x_i = 1$ means that job_i is scheduled.	easy to understand; easy to solve	fail to describe the complex constraints;
Graph-based formulation	directed acyclic graph (DAG); graph coloring problem	[12, 33, 48]	• directed acyclic graph: $G = (X, U)$: the candidate observations are denoted by the set of vertices X, and the precedence relations between observations are represented by the set of arcs U. For any pair $(j, k) \in X \times X$, $(j, k) \in U$ if and only if task k can be observed just after task j. • graph coloring problem: the set of jobs is represented as the set of vertices V and the collection of resources (satellites) as k (the number of colors). Assign a resource to a job while respecting the following constraint: a job can be associated with any resources as long as the job is not in conflict with other jobs in the timeline	easy to understand; easy to solve	not suitable for the complex scheduling problems

otherwise. Thus, a schedule corresponds to the non-zero components of a binary vector. In addition, Wolfe et al. [44] defined the problem as a "Window-Constrained Packing Problem" (WCPP).

Graph-based formulation: Gabrel et al. [12] adopted a directed acyclic graph (DAG) model to describe the satellite scheduling problem. In their formulation, the observations were denoted by vertices, and the precedences between observations were represented by arcs. Hence, the scheduling problem was formulated as the selection of a multiple criteria path in a graph. In addition, Sarkheyli et al. [33] and Zufferey et al. [48] modeled EOS scheduling as graph coloring problems.

The above formulation methods have been summarized in Table 1. Furthermore, the solution approaches for EOS scheduling can be classified into the following three categories, as is shown in Table 2.

Exact algorithms: Agnèse et al. [1] and Bensana et al. [5] proposed depth-first branch-and-bound algorithms for the scheduling of the SPOT-5 satellite. Also, Bensana et al. [5] and Verfaillie et al. [39] suggested Russian Doll search algorithms, which are based on branch-and-bound, but replace one search by n successive searches on nested subproblems, using the results of each search when solving larger subproblems, in order to improve the lower bound on the global valuation of any partial assignment. Besides, Gabrel et al. [12] and Hall et al. [20] developed dynamic programming methods to obtain the optimal solutions of non-agile EOS scheduling problems.

Metaheuristics: A large number of metaheuristics were proposed for non-agile EOS scheduling, which primarily contain tabu search algorithms [5, 36, 46, 48], evolution algorithms [2, 21, 24, 32, 34, 44, 46], ant colony algorithms [25, 40, 45], local

Table 2: Solution algorithms

Categories	Content	references	general forms	pros	cons
Exact algorithms	• branch-and-bound; • Russian Doll search; • dynamic programming	[1, 5, 12, 20, 39]	• branch-and-bound: divide the solution space into some subspaces (branching process), and eliminate the subspaces that do not contain the optimal solution (pruning process) • Russian Doll search: based on branch and bound, replace one search by n successive searches on nested subproblems • dynamic programming: formulate the scheduling problem as multistage decision problems, consider which task to be scheduled at each stage, and the task that cannot produce the optimal solution will be eliminated	optimal guarantee	consume too much time; consume too much memory; not available for large-scale problems; dependent on the problem modeling
Meta-heuristics	tabu search algorithms; evolution algorithms; ant colony algorithms; local search algorithms; simulated annealing algorithms	[2, 5, 15, 16, 21, 24, 25, 32, 34, 36, 38, 40, 44, 45, 46, 48]	stochastic searching in the solution space, first obtain an initial solution or multiple initial solutions (population), and then find some better solutions from neighborhood search. In addition, some rules are adopted to avoid premature end	universal applicability; faster than exact methods; controllable solution time	not guaranteed to be optimal; cannot evaluate the quality of solutions; slow convergence rate
Heuristics	greedy algorithms; constructive algorithms;	[1, 5, 7, 8, 10, 20, 30, 41, 42, 44, 47]	the solution is constructed based on some problem-specific rules (such as priority rule, shortest processing time, first come first service, etc.)	fast for solution; easy to design	not guaranteed to be optimal; bad performance in some cases; problem-specific, cannot be universally used;

search algorithms [35, 38] and simulated annealing algorithms [15, 16, 46].

Heuristics: Agnèse et al. [1] and Bensana et al. [5] proposed greedy algorithms to get feasible solutions for non-agile EOS scheduling problems. On the basis of some problem-specific rules, Bianchessi et al. [7, 10], Hall et al. [20], Wang et al. [41, 42], Wolfe et al. [44] and Xu et al. [47] developed constructive algorithms that can solve the problems efficiently, without guaranteeing the optimality of the solutions. Bianchessi et al. [8] and Marinelli et al. [30] adopted lagrangian relaxation heuristics to solve the problems, obtaining close-to-optimal solutions.

Many observations performed by EOSs are lost due to the presence of clouds, because most EOSs are equipped with optical sensors that cannot see through clouds [15, 16]. For example, around 80% of the observations with the currently operational optical SPOT satellites are useless due to the presence of clouds [3]. Hence, the impact of clouds is a nontrivial issue for EOS scheduling, which cannot be ignored. Unfortunately, to the best of our knowledge, only a few studies have considered the impact of clouds. Lin et al. [27, 28, 29] formulated the presence 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 large uncertainties of clouds, which are always changing over time [2, 6, 22] and cannot be forecasted exactly, so decision makers cannot get the deterministic information beforehand. Liao et al. [26] 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.

In deterministic multi-satellite scheduling, a task will be completed successfully once it has been scheduled. Hence, for each task, it is sufficient to schedule it to only one orbit, and there is no difference between scheduling it to one orbit or to multiple orbits. In contrast, in the scheduling of multiple EOSs under uncertainties, a scheduled task can fail to be completed at a certain probability. If a task can be allocated to multiple resources, the probability of completing this task will be higher. Unfortunately, in the previous studies of uncertain EOS scheduling, the characteristics of deterministic scheduling were simply and inadequately migrated, and each task was scheduled to only one resource, which is not appropriate for the scheduling of EOSs under uncertainties of clouds.

Differently from the previous studies, this paper suggests to schedule each task to multiple orbits and establishes a corresponding expectation model, for the purpose of increasing the probability of the successful observation for each task. Because of the complexities of the problem, the proposed model is neither linear nor quadratic, which brings many challenges for its solution. Both exact and heuristic algorithms are proposed to find optimal and feasible solutions, respectively. For the problems of small size, an exact algorithm based on enumeration is developed, in which each subproblem is solved using path programming and all the feasible solutions of subproblems are combined to solve the master problem. Compared to the mixed integer nonlinear programming solver - Couenne that can solve our problem optimally, our exact algorithm reduces the complexity of solving the model using a divide-and-conquer strategy. In addition, a number of efficient heuristic algorithms are designed to generate feasible solutions that are close-to-optimal for large-scale problems. Afterwards, an extensive simulation experiment is conducted to demonstrate that the solutions of our model perform better than those of previous studies. The experimental results indicate that our exact algorithm is much faster than Couenne for solving small to medium size problems. Finally, the feasibility of both the exact algorithm and the heuristics is verified, coupled with an evaluation of the quality and the performance of each solution.

The remainder of the paper is organized as follows. In the next section, we formulate the problem with a novel mathematical model, and propose an exact algorithm based on enumeration. Section 3 suggests a number of efficient heuristic algorithms for the large-scale problems. In Section 4, a real case study is provided. Subsequently, numerical computational results of our approaches are presented in Section 5. The last section offers conclusions and directions for future research.

2. The uncertain EOS Scheduling Problem

This study focuses on the scheduling of multiple EOSs under uncertainties of clouds, where the presence of clouds for observations is formulated as stochastic events. Essentially, the problem is a stochastic programming problem, and a novel mathematical model is constructed for this problem.

2.1. Problem description

In the previous studies, researchers usually formulate the satellites as the resources, and assume a task will have at most one observation window on each resource. However, most EOSs are nearly polar heliosynchronous satellites, and they can pass over a strip for multiple times if the scheduling horizon is long enough. Hence, the observation windows for a task on each satellite will not be unique [41, 42], which makes the problem difficult for modeling and solving. To handle the difficulties, we formulate the orbits of the satellites as the resources. Hence, there will be at most one observation window for each task on each resource.

Some notations of this study are summarized in Table 3. Let T be the set of tasks (strips) submitted by users and let O be the set of orbits within the scheduling horizon. Each task $i \in T$ is associated with 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 $b_{ik} = 0$. $[s_{ik}, f_{ik}]$ denotes the time duration for task i on orbit k. Specifically, s_{ik} represents the start time of task i on orbit k and f_{ik} represents the finish time.

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

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

where u_{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 time of sensor shutdown, attitude stability and startup of orbit k, respectively. Besides, σ_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.

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 uncertainties of clouds, we formulate the presence of clouds for observations as stochastic events, denoted by 0-1 stochastic 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 presence 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 presence of clouds, thus we can obtain $p\{\tilde{\lambda}_{ik} = 1\} = p_{ik}$ and $p\{\tilde{\lambda}_{ik} = 0\} = 1 - p_{ik}$.

Table 3: Notations

T	set of tasks, $T = \{1, ..., n\}$
i, j, j'	task index, $i, j, j' \in T \cup \{\text{s,t}\}$, 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, $k \in O$
$[s_{ik}, f_{ik}]$	time duration 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, $i \in T, k \in O$
u_{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, $i, j \in T, k \in O$
sd_k	time of sensor shutdown of orbit k, $k \in O$
as_k	time of attitude stability of orbit k, $k \in O$
su_k	time of startup of orbit k, $k \in O$
σ_k	slewing velocity of orbit k, $k \in O$
π_k	energy consumption for each unit slewing angle on orbit k, $k \in O$
$\tilde{\lambda}_{ik}$	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, $i \in T, k \in O$

2.2. Mathematical model

In this paper, we model the scheduling of EOSs with a directed acyclic graph (DAG) formulation. For each orbit k, a directed acyclic graph $G^k = (V^k, A^k)$ is first defined. With the task-on-node representation, the nodes in V^k represent the tasks that are available for orbit k, plus two special nodes $\{s, t\}$ representing the dummy starting and dummy terminating tasks. A^k is the set of arcs for orbit k, which is defined as follows:

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

In this formulation, for each orbit k, binary decision variables $x^k_{ij} \in \{0, 1\}, \forall i, j \in T, k \in O, (i, j) \in A^k, b_{ik}, b_{jk} = 1$ are used as flow variables that are equal to 1 if edges $(i, j) \in A^k$ are selected and 0 otherwise. Clearly, a path from the starting node s to the final node t that satisfies the memory and energy constraints represents a feasible schedule. A non-linear mathematical model of our scheduling problem is given below:

$$max \sum_{i \in T} \omega_i \cdot \{1 - \prod_{k \in O} (1 - p_{ik} \cdot \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} x^k_{ij})\} \tag{3}$$

subject to

$$\sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} x^k_{ij} = \sum_{\substack{j' \in T \cup \{s\} \\ j' \neq i}} x^k_{j'i}, \quad \forall i \in T, k \in O, (i, j) \in A^k, (j', i) \in A^k, b_{ik}, b_{jk}, b_{j'k} = 1 \tag{4}$$

$$\sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} x^k_{ij} \leq 1, \forall i \in T, k \in O, (i, j) \in A^k, b_{ik} = 1. \tag{5}$$

$$\sum_{i \in T} \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} x^k_{ij} (f_{ik} - s_{ik}) m_k \leq M_k, \forall k \in O, \quad (i, j) \in A^k, b_{ik} = 1, b_{jk} = 1. \tag{6}$$

$$\sum_{i \in T} \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} x^k_{ij} (f_{ik} - s_{ik}) e_k + \sum_{i \in T} \sum_{\substack{j \in T \\ j \neq i}} x^k_{ij} \rho^k_{ij} \leq E_k, \quad \forall k \in O, (i, j) \in A^k, b_{ik} = 1, b_{jk} = 1. \tag{7}$$

$$x^k_{ij} \in \{0, 1\}, \forall i, j \in T \cup \{s, t\}, k \in O, (i, j) \in A^k, \quad b_{ik} = 1, b_{jk} = 1. \tag{8}$$

The objective (3) is to maximize the expectation value of the profits of the executed tasks under uncertainties of clouds, in which $1 - \prod_{k \in O} (1 - p_{ik} \cdot \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} x^k_{ij})$ denotes the completion probability for task i when it is scheduled to multiple orbits. The set of constraints (4) are flow balance constraints, which ensure that the number of predecessors is equal to the number of successors for each task. Besides, constraints (4) also guarantee that the selection of edges constitutes a feasible path from the starting node s to the terminating node t. Constraints (5) enforce that each task can only be scheduled to the orbits that are available for it. Constraints (6) check that the memory consumption of the scheduled tasks cannot exceed the memory capacity for each orbit. Constraints (7) 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.

2.3. An exact solution algorithm

It has been known that the proposed model is neither linear nor quadratic, which makes it challenging to find an optimal solution. As far as we know, Couenne is able to solve the model and obtain optimal solutions for small problems. In addition to Couenne, this paper proposes another exact algorithm based on enumeration to get optimal solutions faster.

With respect to Couenne, a spatial branch-and-bound algorithm is implemented to solve non-convex mixed integer non-linear programming (MINLP) problems [49]. It should be noted that Couenne is a general solver, and it may lose the specific formulation information for some problems. With respect to the problem in our study, only the objective function is coupled, and the constraints are independent for each orbit. Therefore, based on the divide-and-conquer approach, we divide the problem into multiple subproblems, reducing the complexity of solving the model. Hence, the proposed algorithm is faster than Couenne in finding the optimal solution.

In detail, the enumeration algorithm can be described as follows:

Step 1. Solve each subproblem with a path programming algorithm, and obtain all non-dominated solutions for each orbit.

Step 2. Enumerate all feasible solutions for the master problem, which are corresponding to all combinations of feasible solutions for the subproblems.

Step 3. Compute the objective function value for each feasible solution, and select the solution with maximum objective value as the optimal solution.

3. Heuristic algorithms

Because of its large space complexity, the proposed exact algorithm will fail to handle large-scale problems in practice. Hence, heuristics are required to solve the large-scale problems efficiently. Although a lot of heuristic algorithms have been proposed for EOS scheduling, such as greedy algorithms [5, 44], heuristics based on some problem-specific rules [20] and some heuristic algorithms for emergency scheduling [42], all of them are designed for the deterministic scheduling problem where each task needs to be scheduled only once. Hence, the algorithms are not suitable for this work where each task requires being scheduled multiple times due to the uncertain impact of clouds.

In addition, the above heuristic algorithms are all deterministic and single-pass, and the quality of the derived solutions cannot be guaranteed. Hence, in this study, three stochastic and multi-pass heuristic algorithms are designed to obtain multiple unique feasible solutions among which the best one is chosen. The proposed algorithms can be grouped into two categories: column-based heuristics and a knapsack-based heuristic.

3.1. Column-based heuristics

The first heuristic algorithm **Heuristic 1** is based on all the non-dominated solutions (columns) for each subproblem (orbit), which can be obtained by path programming as described in the exact algorithm. Afterwards, the L best solutions are selected on each orbit, and the other solutions are cut off. Differently from the exact algorithm that enumerates all combinations of the columns to get all feasible solutions, **Heuristic 1** selects columns for each orbit from the L solutions and combines them heuristically to get a subset of combinations, which are corresponding to a subset of feasible solutions. Afterwards, the feasible solution in the subset with the maximum objective will be marked as the best solution.

Heuristic 1: Column-based heuristic selection

Step 1. Obtain the set of columns COL_k for each orbit k, $k \in O$, with path programming algorithm (see Section 2.3).

Step 2. Compute the profits of the columns with $prof_{kr} = \sum_{i \in T} \omega_i \cdot p_{ik} \cdot \pi_{irk}$, for each column $col_{kr} \in COL_k$, each orbit $k \in O$, in which $\pi_{irk} = 1$ if task i is scheduled to orbit k in column col_{kr}, otherwise $\pi_{irk} = 0$.

Step 3. For each orbit $k \in O$, choose a predefined number L of columns from COL_k with maximum profits to form a new and small column set COL_k, and delete the other columns.

Step 4. Generate a certain number Z of feasible solutions based on the heuristic combination of the columns. For each orbit $k \in O$, Select a column col_{kr} from COL_k with biased random sampling, with the profits of the columns being priority values, and the priority rule is to select the column with the highest priority value. Combine the obtained columns for each orbit to form a feasible solution sol_l, and put sol_l into the solution set SOL.

Step 5. Select the best solution from SOL, i.e., the solution with the maximum objective value.

```
Step 1'. Generate a predefined number of columns for each orbit
    For each orbit $k \in O$
        For $l = 1, \dots, L$
            $j \leftarrow t$ // $t$ is the terminate node of orbit $k$
            While $j \neq s$ // $s$ is the starting node
                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.
                $j \leftarrow i$
            End while
            Obtain the column $col_{kl}$, $COL_k \leftarrow COL_k \cup \{col_{kl}\}$
        End for
    End for
```

Heuristic 1 gets all columns for each orbit with path programming which will consume too much time. Besides, if there is not enough time to obtain all columns for each orbit, neither optimal nor feasible solutions will be produced. To overcome this drawback, **Heuristic 2** is developed, which considers obtaining a subset of columns for each orbit by using a backtracking approach. In detail, the procedure starts from the dummy terminating node t, and it selects a predecessor i using biased random sampling. Subsequently, the above procedure will be repeated until reaching the starting node s, which implies that a feasible solution has been produced. Finally, the above procedures will be repeated L times to get L feasible solutions. As a consequence, **Heuristic 2** is identical to **Heuristic 1** except that **Step 1** is replaced by **Step 1'**, and **Step 3** is removed.

3.2. Knapsack-based heuristic

Both **Heuristic 1** and **Heuristic 2** can be attributed to column-based heuristics, which combine the columns for each orbit to form feasible solutions. In contrast, **Heuristic 3** handles the problem as a knapsack problem and solves it based on task allocation and retraction. Before describing **Heuristic 3** in detail, some definitions and details are proposed below:

Definition 1. Task Requirement

The task requirement TR_i represents the priority to schedule task i:

$$TR_i = \frac{\omega_i \cdot [1 - \prod_{k \in O}(1 - p_{ik} b_{ik})]}{\sum\limits_{k \in O} b_{ik}} \tag{9}$$

Definition 2. Conflict

A conflict, say $Conf_{ik}$, is defined as the set of conflicting tasks that are scheduled to orbit k and that are violating the setup time constraints with task i.

Definition 3. Conflict Set

The conflict set $ConfSet_i$ of task i denotes the set of all distinct conflicts on all orbits.

Definition 4. Task Retraction Expense

The task retraction expense TRE_{ik} is defined as the decrease in expected profits if task i is retracted from orbit k.

$$TRE_{ik} = \omega_i \cdot [P_i(O_i) - P_i(O_i \setminus k)] \tag{10}$$

in which O_i is the set of orbits on which task i has been scheduled, and $P_i(O_i)$ represents the relevant completion probability of task i. It must be noted that $P_i(O_i) = 0$ if $O_i = \emptyset$.

```
Heuristic 3 Knapsack-based heuristic
Step 1. Initialize the current optimal solution $OptSol = \emptyset$, current optimal objective value $OptObj = 0$, solution
        index $l = 0$, predefined number of solutions $Z$.
Step 2. If $l < Z$, Go to Step 3, otherwise the algorithm ends.
Step 3. Initialize the task set $T$, the orbit set $O$, the set of scheduled tasks $Sche_k = \emptyset$ for each orbit $k$, $k \in O$, and
        the conflict set $ConfSet_i = \emptyset$ for each task $i$, $i \in T$.
Step 4. If $T = \emptyset$, go to Step 5; otherwise, go to step 6.
Step 5. Compute the objective value $Obj_l$ of the current solution $Sol_l$. If $Obj_l > OptObj$: $OptObj \leftarrow Obj_l$,
        $OptSol \leftarrow Sol_l$. Afterwards, $l \leftarrow l + 1$, go to Step 2
Step 6. Select task $i$ from $T$ with maximum task requirement $TR_i$, $T \leftarrow T \setminus i$.
Step 7. Schedule task $i$ on each feasible orbit
        For each orbit $k \in O$
          If there is no conflict of task $i$ on orbit $k$
            Schedule task $i$ to orbit $k$, $Sche_k \leftarrow Sche_k \cup \{i\}$;
            While memory and energy constraints are not all satisfied
              Select task $j$ from $Sche_k$ with regret based biased random sampling, with the task retraction expense
              $TRE_{jk}$ being the priority values, and the priority rule is to select the task with the smallest priority
              value.
              Retract task $j$ from orbit $k$, $Sche_k \leftarrow Sche_k \setminus j$, and update the relevant memory and energy.
            End while
          Else
            Obtain the conflict $Conf_{ik}$, $ConfSet_i \leftarrow ConfSet_i \cup \{Conf_{ik}\}$
          End if
        End for
Step 8. If task $i$ has been successfully scheduled to at least one orbit, go to Step 4, otherwise go to Step 9.
Step 9. Calculate the conflict expense $ConfE_{ik}$ for each conflict $Conf_{ik}$, $Conf_{ik} \in ConfSet_i$
Step 10. Select the conflict $Conf_{ik}$ from $ConfSet_i$ with regret based biased random sampling, with the priority
        values being $\omega_i p_{ik} - ConfE_{ik}$ for each conflict, and the priority rule is to select the conflict with the highest
        priority value.
Step 11. Retract all the tasks in conflict $Conf_{ik}$, then Schedule task $i$ to orbit $k$. If both memory and energy
        constraints are satisfied, which implies that task $i$ has been successfully scheduled, update the schedule
        $Sche_k$; otherwise, return to the previous schedule $Sche'_k$. Go to Step 4
```

Definition 5. Conflict Expense

The conflict expense $ConfE_{ik}$ of Conflict $Conf_{ik}$ is defined as the decrease in expected profits if the conflicting tasks in $Conf_{ik}$ is retracted:

$$ConfE_{ik} = \sum_{j \in Conf_{ik}} TRE_{jk} \tag{11}$$

Firstly, **Heuristic 3** selects the task i from the set T with maximum requirement, and schedules i to each orbit. For an orbit k, if there are no conflicting tasks, task i will be scheduled on k, and we retract some tasks based on regret-based biased random sampling to satisfy both the memory and energy constraints if necessary. If task i has been successfully scheduled on at least one orbit, the next task will be accessed; otherwise, a conflict $Conf_{ik}$ will be selected from the set $ConfSet_i$ with regret-based biased random sampling. Subsequently, the tasks in $Conf_{ik}$ will be retracted to accommodate task i if both the memory and energy constraints are satisfied. Then the next task will be selected until $T = \emptyset$, which implies that a feasible solution has been obtained. Finally, the above procedure will be repeated Z times to get Z feasible solutions, and the best solution will be selected.

4. Case study

In this section, a real-world case is provided to validate the feasibility of our methods. The observation requests that were submitted to the China Centre for Resource Satellite Data and Application (CCRSDA) are illustrated in Table 4. The relevant targets needed to be observed for disaster monitoring, and the optical observations required being executed between 8:00 and 16:00 on July 24, 2018.

Six earth observation satellites (GF-1, GF-2, ZY1-02C, ZY1-04, ZY3 and ZY3-02) that are under control of the CCRSDA were selected to execute the submitted requests, and the orbital parameters of the satellites are referred to the website of CCRSDA [51]. In addition, the relevant observation windows are illustrated in Table 5. The probabilities of successful observations are obtained from the cloud coverage of the observation target, and the cloud coverage is provided by the National

Table 4: Observation requests

No.	Region	Latitude	Longitude	Profit
0	Gaolan County, Gansu Province	$36^\circ 26' 47.194''$ N	$103^\circ 50' 47.953''$E	1
1	Shanke County, Heilongjiang Province	$48^\circ 41' 55.442''$N	$128^\circ 9' 6.596''$E	6
2	Beijing Tongzhou District	$39^\circ 52' 17.211''$N	$116^\circ 36' 58.646''$E	6
3	Huize County, Yunnan Province	$26^\circ 28' 39.034''$N	$103^\circ 27' 48.256''$E	3
4	Lanshan District, Shandong Province	$35^\circ 23' 39.27''$N	$119^\circ 0' 59.376''$E	8
5	The rear banner of Wulat, bayannur city, Inner Mongolia autonomous region	$41^\circ 34' 23.599''$N	$108^\circ 31' 58.994''$E	10
6	The middle banner of Wulat, bayannur city, Inner Mongolia autonomous region	$41^\circ 5' 24.797''$N	$107^\circ 4' 52.108''$E	1
7	Balikun kazak autonomous county, Hami city, Xinjiang uygur autonomous region	$43^\circ 36' 17.488''$N	$93^\circ 1' 22.805''$E	5
8	Yuepu lake county, Kashgar, Xinjiang uygur autonomous region	$39^\circ 14' 21.967''$N	$76^\circ 47' 3.746''$E	5
9	Shouguang, Shandong Province	$36^\circ 47' 56.400''$N	$118^\circ 41' 16.799''$E	8
10	Southern of Henan Province	$32^\circ 19' 12''$N	$114^\circ 36' 36''$E	8
11	Northern of Anhui Province	$32^\circ 31' 12''$N	$116^\circ 8' 24''$E	9
12	Southern of Anhui Province	$30^\circ 44' 24''$N	$117^\circ 10' 48''$E	4
13	Northern of Jiangxi Province	$29^\circ 19' 48''$N	$116^\circ 22' 48''$E	8

Meteorological Information Center [50]. In addition, for all the heuristics, the predefined number of obtained solutions Z is set to 1000. The number of columns L for each orbit is set to 20 for **Heuristic 1** and **Heuristic 2**.

The actual schedule obtained by the current system is illustrated as below:

$$\begin{gathered}
\text{Satellite 0: } s \to 9 \to 3 \to t \\
\text{Satellite 1: } s \to 13 \to 6 \to t \\
\text{Satellite 2: } s \to 5 \to 8 \to t \\
\text{Satellite 3: } s \to 1 \to 10 \to t \\
\text{Satellite 4: } s \to 11 \to 7 \to t \\
\text{Satellite 5: } s \to 12 \to 2 \to t
\end{gathered}$$

The optimal solution of our model obtained by both our exact algorithm and Couenne is shown as follows:

$$\begin{gathered}
\text{Satellite 0: } s \to 9 \to 3 \to 7 \to 8 \to t \\
\text{Satellite 1: } s \to 1 \to 13 \to 5 \to t \\
\text{Satellite 2: } s \to 1 \to 5 \to 8 \to t \\
\text{Satellite 3: } s \to 1 \to 10 \to 8 \to t \\
\text{Satellite 4: } s \to 11 \to 7 \to t \\
\text{Satellite 5: } s \to 13 \to 2 \to 7 \to t
\end{gathered}$$

The solutions of **Heuristic 1** and **Heuristic 2** are the same with the above optimal solution. The solution of **Heuristic 3** is described as below:

$$\begin{gathered}
\text{Satellite 0: } s \to 9 \to 3 \to 7 \to 8 \to t \\
\text{Satellite 1: } s \to 1 \to 13 \to 5 \to t \\
\text{Satellite 2: } s \to 1 \to 5 \to 8 \to t \\
\text{Satellite 3: } s \to 1 \to 10 \to 8 \to t \\
\text{Satellite 4: } s \to 12 \to 2 \to 7 \to t \\
\text{Satellite 5: } s \to 11 \to 7 \to t
\end{gathered}$$

With respect to the above solutions, 20 sample tests were conducted to compare the performances of the solutions. For each sample test, 10,000 samples were randomly produced according to the probabilities of successful observations that are described in Table 5. The minimum observation profit, average observation profit and maximum observation profit on the samples are listed in Table 6.

As shown in Table 6, although the maximum profits of the optimal solution, the solution of **Heuristic 1** and the solution of **Heuristic 2** on the samples are normally less than those of the current schedule, the solutions of them outperform the current schedule on average. In addition, the solution of **Heuristic 3** is compared with the current schedule, and it turns out that the minimum profits, the average profits and the maximum profits of **Heuristic 3** are higher.

Table 5: Time windows

Task No.	Satellite											
	GF-1		GF-2		ZY1-02C		ZY1-04		ZY3		ZY3-02	
	Wind	Prob	Wind	Prob	Wind	Prob	Wind	Prob	Wind	Prob	Wind	Prob
0	–	–	–	–	[14:51:31.307, 14:54:03.321]	0.36	[14:51:02.540, 14:53:33.382]	0.42	–	–	–	–
1	–	–	[13:04:36.851, 13:06:50.993]	0.32	[13:14:14.429, 13:16:45.943]	0.29	[13:13:52.603, 13:16:22.803]	0.35	–	–	–	–
3	[12:05:55.136, 12:08:13.094]	0.38	–	–	–	–	–	–	–	–	–	–
4	[10:30:57.692, 10:33:14.462]	0.42	–	–	–	–	–	–	–	–	–	–
5	–	–	[14:39:58.030, 14:42:10.814]	0.28	[14:52:38.005, 14:55:09.914]	0.29	[14:52:09.194, 14:54:39.696]	0.34	–	–	–	–
6	[14:39:54.780, 14:42:08.218]	0.42	[14:52:34.874, 14:55:05.936]	0.38	[14:52:06.059, 14:54:35.771]	0.36						
7	[12:10:54.700, 12:13:11.537]	0.33	–	–	–	–	–	–	[16:04:42.395, 16:06:16.420]	0.35	[16:04:42.395, 16:06:16.420]	0.35
8	[13:47:07.059, 13:49:27.218]	0.42	–	–	[16:32:56.462, 16:35:29.494]	0.44	[16:32:20.470, 16:34:52.385]	0.41	–	–	–	–
9	[10:31:20.561, 10:33:37.355]	0.56	–	–	–	–	–	–	–	–	–	–
10	–	–	[14:37:15.975, 14:39:28.623]	0.42	[14:49:49.201, 14:52:24.587]	0.38	[14:49:20.667, 14:51:54.488]	0.47	[14:27:06.915, 14:28:38.119]	0.41	[14:27:06.915, 14:28:38.119]	0.41
11	–	–	[14:37:14.271, 14:39:28.436]	0.32	–	–	–	–	[14:27:05.678, 14:28:36.882]	0.35	[14:27:05.678, 14:28:36.882]	0.35
12	–	–	[14:36:43.123, 14:38:57.959]	0.28	–	–	–	–	[14:26:35.645, 14:28:06.894]	0.27	[14:26:35.645, 14:28:06.894]	0.27
13	–	–	[14:36:23.737, 14:38:37.053]	0.38	–	–	–	–	[14:26:16.467, 14:27:47.388]	0.36	[14:26:16.467, 14:27:47.388]	0.36

Table 6: Results of sample test

Sample No.	Optimal Solution			Heuristic 1			Heuristic 2			Heuristic 3			Current Schedule		
	Min	Ave	Max	Min	Ave	Max	Min	Ave	Max	Min	Ave	Max	Min	Ave	Max
0	0	37.030	68	0	37.030	68	0	37.030	68	0	36.421	72	0	29.067	72
1	0	37.001	68	0	37.001	68	0	37.001	68	0	36.236	72	0	29.117	72
2	5	37.069	68	5	37.069	68	5	37.069	68	0	36.412	72	0	29.263	72
3	0	36.925	68	0	36.925	68	0	36.925	68	0	36.176	72	0	28.845	65
4	0	37.157	68	0	37.157	68	0	37.157	68	0	36.373	72	0	29.279	66
5	0	36.997	68	0	36.997	68	0	36.997	68	0	36.273	72	0	29.045	69
6	0	37.189	68	0	37.189	68	0	37.189	68	0	36.638	72	0	29.267	73
7	3	37.242	68	3	37.242	68	3	37.242	68	0	36.463	72	0	29.232	73
8	0	36.964	68	0	36.964	68	0	36.964	68	3	36.276	72	0	29.083	72
9	5	36.997	68	5	36.997	68	5	36.997	68	0	36.297	72	0	29.150	67
10	0	36.969	68	0	36.969	68	0	36.969	68	0	36.333	72	0	28.988	69
11	0	36.944	68	0	36.944	68	0	36.944	68	0	36.298	72	0	29.053	72
12	0	37.161	68	0	37.161	68	0	37.161	68	3	36.426	72	0	29.166	69
13	0	37.060	68	0	37.060	68	0	37.060	68	0	36.353	72	0	28.969	69
14	0	37.173	68	0	37.173	68	0	37.173	68	0	36.460	72	0	29.238	70
15	0	36.812	68	0	36.812	68	0	36.812	68	0	36.075	72	0	29.041	72
16	0	36.909	68	0	36.909	68	0	36.909	68	0	36.225	72	0	28.993	67
17	0	37.059	68	0	37.059	68	0	37.059	68	0	36.387	72	0	29.009	70
18	0	36.959	68	0	36.959	68	0	36.959	68	0	36.313	72	0	29.012	68
19	0	37.149	68	0	37.149	68	0	37.149	68	0	36.401	72	0	29.227	70

Table 7: Parameters of satellites

Satellite	Slewing velocity	Startup time	Shutdown time	Stability time	Memory /time	Energy /time	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

Table 8: Performance comparisons between the enumeration algorithm and Couenne

m	n	Enumeration Algorithm		Couenne	
		Obj	Time(s)	Obj	Time(s)
9	10	12.5441	<0.001	12.5441	0.080
	20	26.3332	<0.001	26.3332	0.240
	30	28.8549	<0.001	28.8549	1.520
	40	50.5927	0.343	50.5927	6.060
	50	55.5849	0.671	55.5849	16.900
	60	59.1567	0.358	59.1567	14.620
21	10	27.8513	<0.001	27.8513	0.530
	20	64.4809	<0.001	64.4809	3.430
	30	91.2543	5.492	91.2543	34.020

5. Computational results

For this section, a great number of problem instances were created in order to evaluate the effectiveness and efficiency of our proposed approaches. The computational tests have two components: on some small instances, the superiority of our proposed non-linear robust model is verified and the feasibility of both the exact algorithm and the heuristics is assessed; on the other hand, the performances of the heuristics are compared on some large problem instances.

In order to verify the effectiveness and efficiency of our algorithm, the tasks are randomly generated in the area: latitude 0°-60° and longitude 0°-150°. Without loss of generality, the profits of tasks are integers, uniformly distributed in the interval [1,10]. In correspondence with the literature [5, 11, 21, 31], three different satellites are considered in this paper. The parameters of the satellites are outlined in Table 7. In addition, the memory capacity and energy capacity for each orbit are randomly generated in the intervals [120,160] and [180,240], respectively. Considering the uncertainties of clouds, for each time duration of observation, the probability that there is no presence of clouds, i.e. the observation is successful, will be uniformly distributed in [0.2,1].

The algorithms were implemented in C++ and ran on a personal laptop equipped with an Intel(R) Core(TM) i5-2430M 2.40 GHz (2 processors) and 4 Gb RAM, with operating system Windows 7. Besides, the non-convex MINLP solver is Couenne-0.3.2.

5.1. Performance evaluation on the small problem instances

In this section, the scheduling horizon is first set to 6 hours, which is corresponding to 9 orbits. The numbers of tasks are 10, 20, 30, 40, 50 and 60, respectively. Then the number of orbits is increased to 21, and the numbers of tasks are set to be 10, 20 and 30, respectively. It must be noted that for 21 orbits, we can only test instances with fewer than 30 tasks due to the high memory requirements. For each parameter setting, 10 problem instances will be randomly generated. For all problem instances, Couenne, the exact algorithm and the heuristics were applied. With respect to the heuristics, the predefined number of feasible solutions Z is set to 1000, and the number of columns for each orbit L is 20.

5.1.1. Performance comparisons between Couenne and our exact algorithm

Firstly, the solution performances of our exact algorithm is compared with those of Couenne. The comparison results are illustrated in Table 8 in which column "m" denotes the number of orbits, and "n" indicates the number of tasks. Besides, columns "Obj" indicate the average objective values for the 10 problem instances, and columns "Time" represent the total solution time. As shown in Table 8, both Couenne and our exact algorithm can solve the problems to obtain optimal solutions, but our exact algorithm consumes less time.

Table 9: Performance evaluation on the samples

m	n	Exact Algorithm			Heuristic1			Heuristic2			Heuristic3			CPLEX		
		a	b	c	a	b	c	a	b	c	a	b	c	a	b	c
9	10	1.0	12.70	18.2	1.0	12.70	18.2	1.0	12.70	18.2	1.0	12.55	18.2	0.1	11.92	18.2
	20	2.8	26.69	36.8	2.8	26.69	36.8	2.8	26.69	36.8	1.9	25.95	36.8	0.9	24.43	36.8
	30	6.0	29.97	39.5	6.0	29.92	39.5	5.5	29.82	39.5	5.1	28.81	38.2	2.5	27.45	38.9
	40	15.7	54.35	75.2	15.7	54.31	75.2	15.4	53.04	74.1	11.4	49.92	72.1	9.7	48.35	73.5
	50	19.8	60.63	84.1	19.8	60.01	83.7	19.5	60.09	83.5	14.2	55.23	79.0	11.0	52.10	80.8
	60	19.3	64.52	93.4	20.2	64.40	93.4	19.8	63.01	91.3	14.7	58.59	87.8	10.7	54.75	91.3
21	10	4.9	27.97	36.4	4.9	27.97	36.4	4.9	27.97	36.4	4.5	27.86	36.4	1.5	24.97	36.4
	20	23.7	66.04	88.4	23.7	66.04	88.4	23.6	65.90	88.3	22.1	64.00	88.0	17.3	60.12	88.0
	30	46.4	94.47	121.8	44.9	93.83	120.5	44.9	92.22	119.7	41.6	90.39	118.6	33.0	84.13	121.2

Table 10: Performance evaluation on the small problems

m	n	Optimal solution	Heuristic1		Heuristic2		Heuristic3	
			Obj	Time(s)	Obj	Time(s)	Obj	Time(s)
9	10	12.544	**12.544**	<0.001	**12.544**	<0.001	**12.544**	0.002
	20	26.333	**26.333**	<0.001	26.215	<0.001	25.951	0.002
	30	28.855	28.847	<0.001	28.828	<0.001	28.809	0.004
	40	50.593	50.590	0.002	49.397	0.003	49.907	0.008
	50	55.585	55.249	0.002	53.816	<0.001	54.937	0.014
	60	59.157	58.827	<0.001	56.512	0.002	58.656	0.016
21	10	27.851	**27.851**	0.003	**27.851**	<0.001	**27.851**	0.008
	20	64.481	64.422	0.004	64.129	0.002	63.987	0.010
	30	91.254	90.532	0.004	88.803	0.002	90.076	0.030

5.1.2. Performance comparisons between Liao's model and our model

In order to verify the superiority of our model, the solutions of our model are compared with those of Liao & Tang's model [26]. Liao & Tang also formulated the presence of clouds as stochastic events with some probabilities, and considered scheduling a task to only one resource and performing it only once, which resembles the traditional deterministic scheduling. In addition, the objective function is described as below [26]:

$$\max_{\gamma} E[\sum_{k} \omega_k x_k + \sum_{i=0}^{I} \sum_{n=1}^{N} y_{in} \sum_{j=1, j\neq i}^{I} \gamma_{ijn}]. \tag{12}$$

The objective is two-fold: 1) maximize the profits of the complete tasks without uncertainties; 2) maximize the expected number of complete tasks under uncertainties of clouds. In this study, Liao & Tang's model is directly solved by CPLEX because it is a standard mixed integer programming model.

For the sake of comparison, the obtained solutions were tested on a large sample with the sample size being 100,000 for each instance. Table 9 shows the comparison results for each problem instance, in which column "m" shows the number of orbits, and "n" indicates the number of tasks. Besides, columns "a", "b" and "c" represent the mean values over 10 instances of respectively the minimum, the average and the maximum scheduling profits obtained over all scenarios in the sample. Because CPLEX is used to solve Liao & Tang's model that schedules each task to at most one orbit and therefore lacks robustness, the solutions of CPLEX are on average worse than those of our algorithms, which has been shown in Table 9. Hence, we can conclude that the proposed model is superior to Liao & Tang's model under uncertainties of clouds.

5.1.3. Performance evaluations of the heuristics

Table 10 describes the performances of the heuristics for solving the small instances, in which the definitions of columns "m", "n", "Obj" and "Time" are the same as those of Table 8, respectively. A bolded number denotes that the heuristic generates the optimal solutions for all problem instances in that set. From Table 10, it is observed that **Heuristic 1** can get better solutions than **Heuristic 2** and **Heuristic 3**.

5.2. Performance evaluation on the large problem instances

Due to its large space complexity, it is truly infeasible for the exact algorithm to solve the large-scale problems that are more practical. Hence, heuristics will be applied more extensively in practice. In this section, in order to evaluate the performance of the proposed heuristics, a number of large-scale problem instances were randomly created. In detail, the numbers of tasks

Table 11: Performance comparisons of the heuristics for 21 orbits

Time Limits	Number of tasks	Heuristic 1		Heuristic 2		Heuristic 3	
		Obj	#F	Obj	#F	Obj	#F
1	120	251.586	10	232.257	10	**264.269**	10
	160	**309.153**	10	300.236	10	291.534	10
	200	**375.375**	8	364.202	10	301.453	10
10	120	256.141	10	239.850	10	**264.568**	10
	160	**362.915**	10	336.405	10	297.419	10
	200	**449.676**	10	416.366	10	305.165	10
60	120	259.896	10	241.476	10	**266.497**	10
	160	**370.867**	10	343.158	10	300.905	10
	200	**463.934**	10	435.598	10	308.597	10

Table 12: Performance comparisons of the heuristics for 42 orbits

Time Limits	Number of tasks	Heuristic 1		Heuristic 2		Heuristic 3	
		Obj	#F	Obj	#F	Obj	#F
1	120	403.608	10	387.431	10	**424.069**	10
	160	476.767	8	464.958	10	**480.738**	10
	200	-	0	**611.007**	10	590.293	10
10	120	431.473	10	409.055	10	**432.770**	10
	160	**552.304**	10	521.883	10	500.973	10
	200	**692.670**	10	660.999	10	606.546	10
60	120	**440.953**	10	416.056	10	434.317	10
	160	**564.523**	10	530.659	10	502.300	10
	200	**714.795**	10	676.523	10	610.697	10

are set to 120, 160 and 200, respectively, and the numbers of orbits are 21 and 42, respectively. Similarly to the former experiments, 10 problem instances were randomly created for each parameter setting. Besides, differently from the former experiments, the number of solutions is not limited beforehand, but a time limit is imposed for each heuristic, and thus the numbers of solutions considered depend on the solution time. The time limits are set to 1, 10 and 60 seconds, respectively.

In order to make the performances of the algorithms as best as possible, the column number for each orbit of both **Heuristic 1** and **Heuristic 2** has been reasonably set to 100 based on preliminary test results. The performances of different heuristics with different time limits are listed in Tables 11 and 12, where columns "Obj" are the average objective values of the solvable instances, and "#Fea" are the numbers of problem instances that can be solved to get feasible solutions. For columns "Obj", the bold numbers imply the largest average value for this parameter setting, without considering the number of instances that are solved successfully.

It is illustrated in Table 11 that for the problems of 21 orbits, when the number of tasks is smaller, namely 120 tasks, **Heuristic 3** performs somewhat better than the other algorithms. However, if the number of tasks increases to 160 or 200, **Heuristic 1** will be the best option for most cases. In addition, the performances of **Heuristic 3** are clearly worse than those of the other heuristics for the larger problems with the number of tasks being 160 or 200. Besides, if the time limit is small and there are more tasks, **Heuristic 1** fails to get feasible solutions for some instances. That is because **Heuristic 1** is based on all columns for each orbit that will be obtained by path programming, of which the solution time will increase exponentially with the number of tasks. Thus, if the path programming algorithm cannot get all feasible columns for each orbit within the time limit, **Heuristic 1** cannot get any feasible solutions. Hence, **Heuristic 1** is not available for large-scale problems when the time limits are very small.

Table 12 shows the comparison results for the problems of 42 orbits, from which we can also conclude that for fewer tasks or a small time limit, **Heuristic 3** is the best algorithm. However, for more tasks (160 or 200) and larger time limits (10 or 60 seconds), **Heuristic 1** will be the best option. In addition, similarly to the previous experiments, for small time limits and large-scale problems (1 second for 160 or 200 tasks), **Heuristic 1** will fail to get feasible solutions for some instances. Comparing **Heuristic 2** and **Heuristic 3**, for fewer tasks (120), **Heuristic 3** performs better because it can get higher scheduling profits, and for more tasks (160 and 200), **Heuristic 2** performs better with only one exception (1 second, 160 tasks). From the above analysis, we can conclude that if the solution time is sufficient, **Heuristic 1** shall be adopted to get very good solutions. Besides, if the solution time is very limited, for problems with more tasks **Heuristic 2** should be used, whereas for problems with fewer tasks **Heuristic 3** is a better choice.

6. Conclusions and future work

This paper formulated the presence of clouds as stochastic events, and then investigated the scheduling of multiple EOSs. Due to the uncertainties of clouds, a task that is scheduled to multiple resources will be completed at a higher probability than if scheduled to one resource only. Hence, we took into account scheduling each task to multiple resources and formulated the problem with a novel mathematical model. First of all, an exact algorithm was suggested to solve the problem optimally, in which each subproblem is solved by path programming, and all feasible solutions are combined to solve the master problem. Due to its large space complexity, the exact algorithm mostly fails to solve large-scale problems that are more practical. To overcome this drawback, three heuristic algorithms were designed to solve the large-scale problems.

From the simulation experiments, the superiority of our robust model was verified compared with the traditional model that allocates each task to only one orbit. In addition, it was proved that for small-size problems, our exact algorithm is faster than Couenne for solving the problem to optimality. Besides, the heuristics can obtain feasible solutions that are very close to the optimal ones for small problems. Finally, for large-scale problems, the performances of the different heuristics were evaluated. The comparison results reveal that for not too many tasks or small time limits, **Heuristic 3** performs best, whereas **Heuristic 1** will be the best for larger problems and larger time limits.

In the future, we will consider the scheduling of agile EOSs under uncertainties. Different from the non-agile satellites in this study, the agile satellites do not only have the maneuverability of slewing, but also the maneuverability of pitching, along with the orbit. Hence, the satellite will have a longer time window for observation. Consequently, the scheduling will have more freedom, and managers need not only to allocate the tasks to the orbits, but also to decide the starting and finishing times. Furthermore, the sequence of task executions will also not be fixed, as for any two tasks i and j, it is now possible to start the execution of task i before that of task j, or the execution of task j before that of task i, which obviously will make the problem more complicated. Besides, to make our work more practical, future research will focus on the combined scheduling of observations and data downloads.

Acknowledgments

We appreciate Shusong Huang of the China Centre for Resource Satellite Data and Application for providing us the real observation requests of the case study. This research was also supported by the National Natural Science Foundation of China under Grants No. 71701067, and No. 71801218.

References

[1] Agnése, J.C., & Bensana, E. (1995). Exact and approximate methods for the daily management of an earth observation satellite. In *proceedings of the fifth ESA workshop on aritificial intelligence and knowledge based systems for space.*

[2] Barbulescu, L., Watson, J.P., Whitley, L.D., & Howe, A.E. (2004). Scheduling space-ground communications for the air force satellite control network. *Journal of Scheduling*, *7*(1), 7-34.

[3] 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.

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

[5] Bensana, E., Verfaillie, G., Agnese, J.C., Bataille, N., & Blumestein, D. (1996). Exact and inexact methods for the daily management of an earth observation satellite. In *Proceedings of the international symposium on space mission operations and ground data systems* (pp. 507-514).

[6] Bensana, E., Verfaillie, G., Michelon-Edery, C., & Bataille, N. (1999). Dealing with uncertainty when managing an earth observation satellite. In *Proceedings of the 5th international symposium on artificial intelligence, robotics and automation in space* (pp. 205-210).

[7] Bianchessi, N., Piuri, V., Righini, G., Roveri, M., Laneve, G., & Zigrino, A. (2004). An optimization approach to the planning of earth observing satellites. In *Proceedings of the 4th international workshop on planning and scheduling for space.*

[8] 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.*

[9] 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.

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

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

[12] 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.

[13] 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.

[14] Gabrel, V. (2006). Strengthened 0-1 linear formulation for the daily satellite mission planning. *Journal of Combinatorial Optimization*. *11*(3), 341-346.

[15] Globus, A., Crawford, J., Lohn, J., & Morris, R. (2002). Scheduling earth observing fleets using evolutionary algorithms: problem description and approach. In *Proceeddings of the 3rd international NASA workshop on planning and scheduling for space*.

[16] 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.

[17] Habet, D., & Vasquez, M. (2004). Solving the selecting and scheduling satellite photographs problem with a consistent neighborhood heuristic. In *16th IEEE International Conference on Tools with Artificial Intelligence(ICTAI 2004)*(pp. 302-309). IEEE.

[18] Habet, D. (2009). Tabu search to solve real-life combinatorial optimization problems: a case of study. In *Foundations of Computational Intelligence Volume 3* (pp. 129-151). Springer Berlin Heidelberg.

[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] Hall, N. G., & Magazine, M. J. (1994). Maximizing the value of a space mission. European journal of operational research, 78(2), 224-241.

[21] Han, S. M., Beak, S. W., Cho, K. R., Lee, D. W., & Kim, H. D. (2008). Satellite mission scheduling using genetic algorithm. In *SICE Annual Conference* (pp. 1226-1230). IEEE.

[22] 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*.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] Lin, W., & Liao, D. (2004). A tabu search algorithm for satellite imaging scheduling. In *Prodeeding of the IEEE international conference on systems, man and cybernetics*(pp. 1601-1606).

[28] Lin, W., & Chang, S. (2005a). Hybrid algorithms for satellite imaging scheduling. In *Proceedings of the IEEE international conference on systems, man and cybernetics* (pp. 2518-2523).

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

[30] Marinelli, F., Salvatore, N., Rossi, F., & Smriglio, S. (2011). A lagrangian heuristic for satellite range scheduling with resource constraints. *Computers & Operations Research*, *38*(11), 1572-1583.

[31] Pemberton, J.C., & Greenwald, L.G. (2002). On the need for dynamic scheduling of imaging satellites. In *Proceeding of the American society for photographing and remote sensing (ASPRS'02).*

[32] Salman, A. A., Ahmad, I., & Omran, M. G. (2015). A metaheuristic algorithm to solve satellite broadcast scheduling problem. *Information Sciences*, *322*, 72-91.

[33] Sarkheyli, A., Vaghei, B.G., & Bagheri, A. (2010). New tabu search heuristic in scheduling earth observation satellites. In *Proceedings of the 2nd international conference on software technology and engineering.*

[34] Soma, P., Venkateswarlu, S., Santhalakshmi, S., Bagchi, T., & Kumar, S. (2004). Multi-satellite scheduling using genetic algorithms. In *Proceedings of the 8th international conference on space operations.*

[35] 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.

[36] 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.

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

[38] Verfaillie, G., & Schiex, T. (1994). Solution reuse in dynamic constraint satisfaction problems. In *Proceeding of the twelfth national conference on artificial intelligence (AAAI'94)*(pp. 307-312).

[39] Verfaillie, G., Lemaitre, M., & Schiex, T. (1996). Russian doll search for solving constraint optimization problems. In *Proceedings of the 13th national conference on artificial intelligence* (pp. 181-187).

[40] Wang, H., Xu, M., Wang, R., & Li, Y. (2009). Scheduling earth observing satellites with hybrid ant colony optimization algorithm. In *Proceeding of the international conference on artificial intelligence and computational intelligence (AICI'09).*

[41] Wang, J., Zhu, X., Qiu, D., & Yang, L.T. (2014). Dynamic scheduling for emergency tasks on distributed imaging satellites with task merging. *IEEE Transactions on Parallel and Distributed System*, *25*(9), 2275-2285.

[42] Wang, J., Zhu, X., Yang, L.T., Zhu, J., & Ma, M. (2015). Towards dynamic real-time scheduling for multiple earth observation satellites. *Journal of Computer and System Sciences*, *81*(1), 110-124.

[43] 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.

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

[45] 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.

[46] Xhafa, F., Herrero, X., Barolli, A., & Takizawa, M. (2014). A Comparison Study on Meta-Heuristics for Ground Station Scheduling Problem. In *Proceedings of 17th International Conference on Network-Based Information Systems (NBiS)*(pp. 172-179). IEEE.

[47] 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.

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

[49] Belotti P, Lee J, Liberti L, et al. (2009). Branching and bounds tightening techniques for non-convex MINLP. *Optimization Methods & Software*, *24*(4-5):597-634.

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

[51] China Centre for Resource Satellite Data and Application. http://www.cresda.com/EN/.