KU LEUVEN
FACULTY OF ECONOMICS AND BUSINESS

Proactive scheduling algorithms for multiple earth observation satellites under uncertainties of clouds

Wang J, Demeulemeester E, Qiu D.

KBI_1515

Noname manuscript No.
(will be inserted by the editor)

Proactive scheduling algorithms for multiple earth observation satellites under uncertainties of clouds

Jianjiang Wang · Erik Demeulemeester · Dishan Qiu

Received: date / Accepted: date

Abstract This paper investigates the scheduling of multiple earth observation satellites (EOSs) under uncertainties of clouds. Firstly, we formulate the presence of clouds as stochastic events, transforming the problem into a stochastic programming problem. Based on different perspectives, we model the problem mathematically using both an expectation model and a chance constrained programming (CCP) model. Afterwards, for the first time, we employ a Dantzig-Wolfe decomposition and a column generation technique for the uncertain scheduling of EOSs. With respect to the expectation model, we devise a branch-and-price algorithm to solve the model optimally and efficiently. On the other hand, we first reformulate the CCP model as a mixed integer programming (MIP) model using sample approximation. Subsequently, considering the difficulties and the infeasibility of the branch-and-price algorithm for this MIP model, we suggest a column generation based heuristic algorithm to get "good" feasible solutions. By numerous simulation experiments, we verify the effectiveness and test the performance of our proposed formulations and approaches.

Jianjiang Wang · Dishan Qiu
Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha, China
E-mail: jianjiangwang.nudt@gmail.com

Jianjiang Wang · Erik Demeulemeester
Research Center for Operations Management, Department of Decision Sciences and Information Management, Faculty of Economics and Business, KU Leuven, Belgium

Erik Demeulemeester
E-mail: erik.demeulemeester@kuleuven.be

Dishan Qiu
E-mail: ds_qiu@sina.com

Keywords earth observation satellites · uncertainties of clouds · expectation model · chance constraint programming · branch-and-price · sample approximation · column generation heuristic

1 Introduction

Earth observation satellites are the platforms equipped with sensors that orbit the earth to take photographs of special areas at the request of users [10, 16]. EOSs can take photographs, while moving along their orbits, which is shown in Fig. 1. After capturing the photographs, the acquired data will be stored in the onboard limited memory and transferred to a ground station when the satellites are in the feasible transferring range. Most of 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.

In this work, the satellite management process is taken into account while multiple EOSs are operated to satisfy users' requests. The requests require the selection, allocation and scheduling in the ground station according to some operational constraints of the satellites before the derived sequence is transmitted.

Because of some unique 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. With the development of space science and technology, the number of satellites

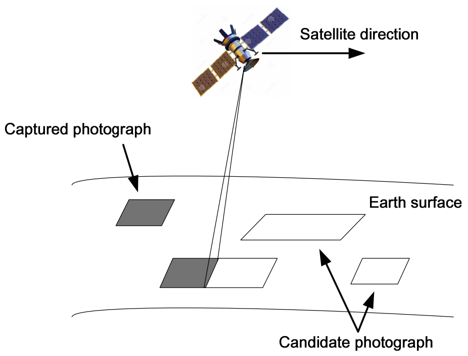

Fig. 1 The satellite captures the photographs [48]

increases continuously. However, satellites are still limited in comparison with the explosively increased number of applications. Hence, scheduling is a significative issue to satisfy more requests and obtain a high observation efficiency.

Although a large number of studies concerning EOS scheduling have been proposed, unfortunately, to the best of our knowledge, the previous studies considering the impact of clouds are extremely limited (see details in Section 2). However, EOS observations are significantly affected by the presence of clouds, since most EOSs are equipped with optical sensors that cannot see through clouds [23,24]. For instance, around 80% of the observations with the currently operational optical SPOT satellites are useless due to the presence of clouds [3]. Besides, the presence and status of clouds are normally random, which cannot be forecasted deterministically. The uncertainties of clouds bring more difficulties for EOS scheduling. Hence, clouds are a nontrivial issue, which requires more focus.

In this study, considering the uncertainties of clouds, we formulate the presence of clouds for observations as stochastic events. From different perspectives, we propose both an expectation model and a CCP model to formulate the scheduling problem of multiple EOSs under uncertainties. With regard to the expectation model that is in fact an integer programming (IP) model, we decompose it into a set-packing master problem and some subproblems using Dantzig-Wolfe decomposition. Moreover, we design a branch-and-price algorithm to solve the model. On the other hand, for the CCP model, we firstly transform it to a MIP model using sample approximation. Subsequently, we also decompose the MIP model into a master problem and some subproblems, and suggest a column generation heuristic (CGH) algorithm to get "good" integer feasible solutions. By numerous simulation experiments, we prove that the branch-and-price algorithm can solve the expectation model optimally and efficiently, and the CGH can solve the CCP model to get feasible solutions that are close-to-optimal in a short time.

The remainder of this paper is organized as follows. In the next section we reveal the previous work. Subsequently, Section 3 describes the problem in detail, and formulates the problem with an expectation model and a CCP model, respectively. In Section 4, we present a branch-and-price algorithm to solve the expectation model. Furthermore, Section 5 proposes a CGH algorithm to solve the CCP model. Numerical results of our approaches are presented in Section 6. The last section offers conclusions and directions for future research.

2 Previous work

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

Mathematical programming: Benoist et al. [5], Habet et al. [25], Lemaître et al. [28] and Tangpattanakul et al. [48] developed general mathematical programming models for EOS scheduling. Liao et al. [31], Lin et al. [32,33] and Marinelli et al. [39] proposed the time-indexed formulation of EOS scheduling, and established integer programming models. In addition, integer programming models are also constructed on the basis of a "flow variable" formulation [9,10,21,22].

Constraint satisfaction problem: Lemaître et al. [27] formulated EOS scheduling as a constraint satisfaction problem. Bensana et al. [6] and Verfaillie et al. [52] proposed valued constraint satisfaction problem (VCSP) formulations for SPOT-5 satellite scheduling.

Knapsack problem: Vasquez et al. [50,51] and Wolfe et al. [58] formulated EOS scheduling as 0-1 knapsack problems.

Graph-based formulation: Gabrel et al. adopted a directed acyclic graph model was adopted to describe the satellite scheduling problem [20]. Besides, Sarkheyli et al. [45] and Zufferey et al. [60] modeled EOS scheduling as graph coloring problems.

Alternatively, Frank et al. [19] and Pralet et al. [41] adopted the Constraint-Base Interval (CBI) language to describe the problem.

In addition, the solution approaches for EOS scheduling can be classified into the following categories.

Exact algorithms: Bensana et al. [6] proposed a depth-first branch and bound algorithm for SPOT-5 satellite scheduling. Also, Benoist et al. [5], Bensana et al. [6] and Verfaillie et al. [52] 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, to improve the lower bound on the global valuation of any partial assignment. Besides, Gabrel et al. [20] and Lemaître et al. [28] developed dynamic programming methods to get the optimal solutions of EOS scheduling problems.

Metaheuristics: A large number of metaheuristics were proposed for EOS scheduling, which primarily contain tabu search algorithms [6,10,16,26,33,50,60], genetic algorithms [2,29,46,47,58], ant colony algorithms [30,59], local search algorithms [27,28,48] and simulated annealing algorithms [23,24,26].

Heuristics: Bensana et al. [6] and Lemaître et al. [28] proposed greedy algorithms to get feasible solutions for EOS scheduling problems. On the basis of heuristic rules, Bianchessi et al. [8,11], Karapetyan et al. [26], Wang et al. [53,54] and Wolfe et al. [58] developed constructive algorithms that can solve the problem efficiently, without guaranteeing the optimality of the solutions. Bianchessi et al. [9], Lin et al. [33] and Marinelli et al. [39] adopted lagrangian relaxation heuristics to solve the problems, obtaining close-to-optimal solutions.

Among all the previous studies, only a few have considered the impact of clouds. Lin et al. [32,33] 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 a lot of uncertainties of clouds, which are always changing over time and are impossible to be forecasted exactly [3,7,27]. Thus decision makers cannot get the deterministic information of clouds before scheduling. Liao et al. [31] 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 [3], the online scheduling of a Pleiades satellite that is equipped with a cloud detection instrument is considered, and the decisions are 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 programmes or for getting bounds for integer programmes. Currently, column generation and branch-and-price that is a combination of column generation and branch-and-bound have been successfully used in many fields [12–14,17,40,42,44]. 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 [21,38] and to evaluate the feasible solutions derived from some heuristics [10]. To the best of our knowledge, only Wang and Reinelt [57] has taken the branch-and-price algorithm into account to get the optimal solutions for some small instances of EOS scheduling.

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, $k \in O$
$[ws_{ik}, we_{ik}]$	time window 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^k_{ij}	setup time between task i and task j on orbit k, $i, j \in T, k \in O$
ρ^k_{ij}	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, $i \in T, k \in O$

3 The uncertain EOS scheduling problem

In this study we focus on the scheduling of multiple EOSs under uncertainties of clouds, in which the presence of clouds for observations is formulated as stochastic events. Essentially, the problem is a stochastic programming problem, and we construct both a mathematical expectation model and a CCP model to formulate the problem.

3.1 Problem description

In EOS scheduling, users normally submit two types of requests: (1) a spot, i.e., a circle with limited dimension, or (2) a polygon which may cover a wide geographical area. Due to its large size, a polygon usually is failed to be observed in a single orbit and therefore partitioned into multiple rectangular strips [10,16,48,53]. In order to facilitate the description, a spot can be seen as a single strip. Hence, the tasks in this work are corresponding to the strips that require being photographed.

In this work, we consider 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 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 time window 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, which 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 time windows 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 $we_{ik} - ws_{ik}$.

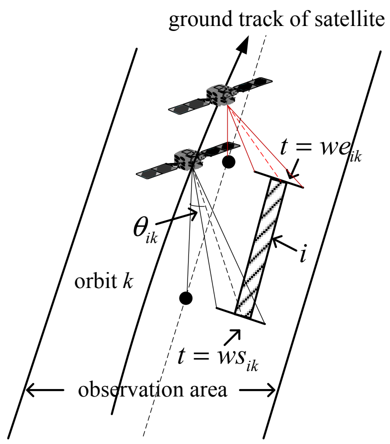

Fig. 2 Time window for observation

After observing a task, the satellite requires a sequence of transformation operations to observe the next one, which are 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 [55]:

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

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:

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

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 $p\{\tilde{\lambda}_{ik} = 0\} = 1 - p_{ik}$.

At present, two different ways for formulating the stochastic programming have been proposed to suit the different purposes of management [34,35]. The first way for stochastic programming is the expectation model, which optimizes the expected values of the criteria subject to some expected constraints. The second, chance constrained programming, was pioneered by Charnes and Cooper [15] as a means of handling uncertainty by specifying a confidence level at which it is desired that the stochastic constraint holds [35]. In this paper, we formulate the stochastic programming using both ways and construct the relevant mathematical models.

3.2 Expectation model

In this study, we formulate the EOS scheduling problem with flow variables, and establish a mathematical expectation model, which maximizes the expected profits of successful observations. 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 $T = \{1, ..., n\}$ is the set of real tasks and 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 $x_{ij}^k = 0$. The mathematical model is given below:

$$max \sum_{i\in T} \sum_{\substack{j \in T\cup\{t\} \\ j\neq i}} \sum_{k\in O} \omega_i \cdot p_{ik} \cdot x_{ij}^k \tag{1}$$

subject to

$$\sum_{\substack{j \in T\cup\{t\} \\ j\neq i}} \sum_{k\in O} x_{ij}^k \leq 1, \forall i \in T \tag{2}$$

$$\sum_{\substack{j \in T\cup\{t\} \\ j\neq i}} x_{ij}^k = \sum_{\substack{j \in T\cup\{s\} \\ j\neq i}} x_{ji}^k,\ \forall i \in T, k \in O \tag{3}$$

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

$$x_{ij}^k(ws_{jk} - we_{ik} - st_{ij}^k) \geq 0, \forall i, j \in T, k \in O \tag{5}$$

$$\sum_{i\in T} \sum_{\substack{j \in T\cup\{t\} \\ j\neq i}} x_{ij}^k(we_{ik} - ws_{ik})m_k \leq M_k, \forall k \in O \tag{6}$$

$$\sum_{i\in T} \sum_{\substack{j \in T\cup\{t\} \\ j\neq i}} x_{ij}^k(we_{ik} - ws_{ik})e_k + \sum_{i\in T} \sum_{\substack{j \in T \\ j\neq i}} x_{ij}^k \rho_{ij}^k \leq E_k, \forall k \in O \tag{7}$$

$$x_{ij}^k \in \{0,1\}, \forall i, j \in T\cup\{s,t\}, k \in O \tag{8}$$

The objective (1) is to maximize the expected value of the profits of the executed tasks under uncertainties of clouds. The set of constraints (2) guarantees that each task will be observed at most once. Constraints (3) are flow balance constraints that force the number of predecessors to be equal to the number of successors for each task. Constraints (4) 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 (5). 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.

3.3 Chance constrained programming model

Next to the expectation model, we also construct the chance constrained programming model for the scheduling of multiple EOSs under uncertainties of clouds. From the above problem description and the previous expectation model, we can arrive at the conclusion that the constraints of our problem are deterministic, but the scheduling objective, namely maximizing the profits of the executed tasks, is stochastic. In [34], Liu has proven that the chance constrained programming is also available for stochastic objectives.

Let $(1-\alpha)$ denote the predefined confidence level. The objective of the chance constrained programming model is:

$$max \bar{f}, \tag{9}$$

in which $\bar{f}$ is constrained by the following chance constraint:

$$P\{\sum_{i\in T} \sum_{\substack{j \in T\cup\{t\} \\ j\neq i}} \sum_{k\in O} \omega_i \cdot \tilde{\lambda}_{ik} \cdot x_{ij}^k \geq \bar{f}\} \geq 1-\alpha \tag{10}$$

The chance constraint (10) states that the probability that the profits of executed tasks under uncertainties of clouds will be at least $\bar{f}$ is larger than or equal to the confidence level $1-\alpha$. Besides, the remaining constraints, such as unicity constraints, setup time constraints, memory constraints, etc, are identical to those of the expectation model. Therefore, the CCP model is formulated as: $max \bar{f}$ subject to: (2)-(8),(10). Compared with the expectation model, the solution complexity of the CCP model is indeed larger due to the chance constraint.

4 Solution approach for the expectation model

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

4.1 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, $r \in R_k$, $k \in O$.

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

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 uncertainties 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

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

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

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

The objective (11) is to maximize the expected value of the profits of executed tasks. Constraints (12) are corresponding to constraints (2), 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 4.3).

It should be noted that the above set packing model decreases the number of constraints compared with the original expectation model, which seems to be faster for solving. However, the number of feasible solutions for each orbit grows exponentially with the problem size, which results in an exponential growth in the computational time. 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, as described in the next section.

4.2 Column generation

Despite the large number of feasible solutions for each orbit, it is possible to solve the LP relaxation using 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 re-optimized. 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 a dynamic programming algorithm (see Section 4.4.1). 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. In addition, because of the block diagonal structure, the column with the most positive reduced cost for each orbit can be identified and added separately.

4.3 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 for the specified orbit with the most positive reduced cost to be added to the current pool of active columns in RMP. Hence, the objective of the subproblem for orbit k, $k \in O$ is outlined as below:

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

Note that the index k can be removed from all decision variables and parameters for each subproblem, since each subproblem is solved separately. Therefore, the above objective function can be rewritten as:

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

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:

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

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.

$[ws_i, we_i]$: time window 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 that task i will be successfully executed, $i \in T$.

Decision variables. The decision variables used in the subproblem 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:

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

subject to

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

$$\sum_{\substack{j\in T\cup\{t\}\\ j\neq i}} \alpha_{ij} \le b_i, \forall i \in T \quad (20)$$

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

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

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

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

Constraints (19)-(24) are corresponding to constraints (3)-(8) 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. Subsequently, we solve the subproblems with a forward-checking dynamic programming algorithm (see Section 4.4.2).

4.4 Dynamic programming

In this work, dynamic programming based on some labels (see [20]) is proposed significantly, which cannot only provide some initial columns for solving the RMP, but also solve the subproblems to price out some promising columns. In order to facilitate the description, 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 in the remaining text. 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 dummy terminating tasks. Each node in V represents an available task and all nodes are numbered in the chronological order of the time windows. 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 constraints between tasks i and j are satisfied.

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 can both obtain the initial feasible solutions and solve the subproblems to add the new columns to the RMP.

4.4.1 Initial feasible solutions

In this study, in order to generate some initial feasible solutions for each orbit, we first assign each node i of the directed acyclic graph a weight ω_i', $\omega_i' = \omega_i p_i$. Afterwards, we originally proposed to enumerate all feasible solutions for each orbit, but unfortunately, for some large-scale problems, this is infeasible due to the limitation on the computer memory. Hence, as an alternative, we enumerate a great number $ColNum$ of feasible solutions using a dynamic programming algorithm. Ultimately, among the $ColNum$ solutions, a predefined number L of elements of maximum weight are selected to be used in the column generation algorithm.

For each node j, we store a set of labels $P(j)$ that represent all paths from the starting node s to j. A label in $P(j)$, corresponding to a path $p = \{s, i_1, \ldots, i_k, j\}$, is denoted by $l_p = [i_k, \uparrow l_{p'}, \Omega_p, M_p', E_p']$, where i_k is the immediate predecessor of j in path p, $\uparrow l_{p'}$ is a pointer on the label referring to the path $p' = \{s, i_1, \ldots, i_k\}$ in $P(i_k)$, Ω_p is the weight of path p, i.e., the sum of weights of the nodes in path p, and M_p', E_p' are the memory consumption and energy consumption of path p, respectively. Besides, $N(j)$ is defined as the number of labels in $P(j)$. To respect the memory limitation, $N(j)$ must be less than or equal to $ColNum$ for each node j. The dynamic programming algorithm is described in **Algorithm 1**, in which $\Gamma^{-1}(j)$ is the set of all predecessors of j. At each iteration, we determine the label set for a node j, $j \leftarrow 1, \ldots, n, t$ (remember here that t represents the dummy terminating task). For each subpath in each predecessor i of j, we add the subpath with the node j and the relevant arc (i, j). If both the memory and energy constraints are satisfied, we will add the label of the path to $P(j)$, and then access the next subpath.

By this algorithm, for each node j (including the terminating node t), at most a number $ColNum$ of paths from the starting node s to j will be stored in the label set $P(j)$. Meanwhile, each feasible path stored in the label set $P(t)$ is corresponding to a feasible solution. Then, a predefined number L of feasible solutions of maximum weight will be selected 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.

Algorithm 1 Dynamic programming

$P(s) \leftarrow \{[null, null, 0, 0, 0]\}$
$P(j) \leftarrow \emptyset, N(j) \leftarrow 0$ $(j \leftarrow 1, \ldots, n, t)$
for $j \leftarrow 1, \ldots, n, t$ **do**
  for all $i \in \Gamma^{-1}(j)$ **do**
    for all $l_p \in P(i)$ **do**
      if memory and energy constraints are satisfied and $N(j) < ColNum$ **then**
        $P(j) \leftarrow P(j) \cup \{[i, \uparrow l_p, \Omega_p + \omega_j', M_p' + m(we_j - ws_j), E_p' + e(we_j - ws_j) + \rho_{ij}]\}$
        $N(j) \leftarrow N(j) + 1$
      end if
    end for
  end for
end for

4.4.2 Solution for the subproblems

For each subproblem, we want to find the absent feasible solution with the most positive reduced cost, and add it to the RMP to improve the objective. Hence, based on the 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 [20], the subproblem can be solved using a forward-checking dynamic programming algorithm. Before describing the algorithm, we first introduce some definitions.

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 inequality holds.

Definition 2: Dominated path and nondominated path. From the starting node s to node j, path p, $l_p \in P(j)$ is a dominated path if $\exists l_q \in P(j)$ such that path p is dominated by path q, otherwise path p is a non-dominated path.

Definition 3: Optimal path. For the paths from s to t, optimal path p is the path with maximum Ω_p, which is corresponding to the optimal solution for the subproblem.

The forward-checking dynamic programming algorithm is outlined in **Algorithm 2**. At each iteration, we determine the label set $P(j)$ for $j = 1, 2, \ldots, n, t$ in two stages. In the first stage, we define for all predecessors i of node j the set $P_i(j)$ of labels associated with efficient subpaths from s to i plus the arc (i, j), with both the memory constraints and the energy constraints being satisfied. In the second stage, we merge the sets $P_i(j)$ for all predecessors i, $i \in \Gamma^{-1}(j)$ while removing all dominated paths in order to obtain $P(j)$. Dominance tests are applied only between paths belonging to different sets $P_i(j)$. At the end of the algorithm, $P(t)$ describes all feasible non-dominated paths from s to t, among which the optimal path corresponding to the optimal solution is selected.

Algorithm 2 Forward-checking dynamic programming

$P(s) \leftarrow \{[null, null, 0, 0, 0]\}$
$P(j) \leftarrow \emptyset \ (j \leftarrow 1, \dots, n, t)$
$P_i(j) \leftarrow \emptyset; (i \in \Gamma^{-1}(j), j \leftarrow 1, \dots, n, t)$
for $j \leftarrow 1, \dots, n, t$ **do**
  {First stage}
  for all $i \in \Gamma^{-1}(j)$ **do**
    for all $l_p \in P(i)$ **do**
      if memory and energy constraints are satisfied **then**
        $P_i(j) \leftarrow P_i(j) \cup \{[i, \uparrow l_p, \Omega_p + \omega'_j, M'_p + m(we_j - ws_j), E'_p + e(we_j - ws_j) + \rho_{ij}]\}$
      end if
    end for
  end for
  {Second stage}
  for all $i \in \Gamma^{-1}(j)$ **do**
    for all $l_p \in P_i(j)$ **do**
      for all $l_{p'} \in P(j)$ **do**
        if P dominated P' then $P(j) \leftarrow P(j) \backslash l_{p'}$
        if P' dominated P then $P_i(j) \leftarrow P_i(j) \backslash l_p$
      end for
    end for
    $P(j) \leftarrow P(j) \cup P_i(j)$
  end for
end for

4.5 Branch-and-bound

From the column generation algorithm, we can solve the LP relaxation of the set packing model to get the optimal solution, which is an upper bound to the optimal solution of the original problem. However, column generation cannot guarantee the integrality. In many cases, the solutions of column generation are not integer solutions. Hence, in order to get the optimal integer solution of the original problem, a branch-and-price algorithm, which is a combination of column generation and branch-and-bound, is required to solve the problem. In the branch-and-price algorithm, column generation is employed at each node of the branch-and-bound tree, then if the solution from column generation is not integer, branching will be employed.

Branching is an important issue in the branch-and-price algorithm, which is different from the typical 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 [4,49]. Currently, there have been some effective branching strategies proposed by researchers. One of the strategies is Ryan-Foster branching, and the other way is to branch on the variables of the original problem.

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 $\sum_{j \in T \cup \{t\}, j \neq i} \alpha_{ijrk} \neq \sum_{j \in T \cup \{t\}, j \neq i} \alpha_{ijsk}$ for task $i, i \in T$, we will branch on task i, i.e., set $\sum_{j \in T \cup \{t\}, j \neq i} x_{ij}^k = 0$ or $\sum_{j \in T \cup \{t\}, j \neq i} x_{ij}^k = 1$. 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 $\sum_{j \in T \cup \{t\}, j \neq i} x_{ij}^k$ is set to 1, the corresponding weight ω'_i is set to $+\infty$ for orbit k; otherwise if $\sum_{j \in T \cup \{t\}, j \neq i} x_{ij}^k$ is set to 0, ω'_i is set to $-\infty$ for orbit k. The second advantage is the fact that this branching strategy yields a balanced branch-and-bound tree.

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

5 Solution approach for the CCP model

Because of some difficulties, the CCP model is intractable to solve directly [36,37]. In order to solve the CCP model, we have transferred it to a mixed integer programming model by sample approximation in the previous research [55].

5.1 Sample approximation

A sample W is a set of scenarios of the random vector $w(\tilde{\lambda}_{ik}), i \in T, k \in O$, such that $W = \{w_1, ..., w_{|W|}\}$, in which $|W|$ is the sample size. The basic idea of the reformulation introduced in [43] is to solve the problem and get a solution, which is infeasible for at most $\lfloor |W| \cdot \epsilon \rfloor$ scenarios. Thus the solution will be feasible for the

Algorithm 3 BRANCH-AND-PRICE

```
Initialize a predefined number of columns with the maximum weights using Algorithm 1.
GlobalUpperBound ← +∞, GlobalLowerBound ← −∞, OptSolObj ← 0
Initialize an empty stack S for storing the nodes of the branch-and-bound tree.
Initialize the RootNode and S ← push(S, RootNode).
while (S ≠ ∅) do
   CurrentNode ← Pop(S);
   LP_opt_found ← FALSE;
   while (LP_opt_found = FALSE) do
      LP_opt_found ← TRUE;
      CurrentNode.LocalUpperBound ← SOLVE-MASTER-LP(); /*solve the linear programming relaxation of the master
      problem*/
      for (k = 1, . . . , m) do
         RC_k ← FIND-NEW-COLUMN(k); /*solve the subproblem k with Algorithm 2 to get the maximum reduced
         cost of the absent columns*/
         if (RC_k > 0) then
            add the new column to the restricted master problem;
            LP_opt_found ← FALSE
         end if
      end for
   end while
   if CurrentNode = RootNode then
      GlobalUpperBound = CurrentNode.LocalUpperBound;
   end if
   if CurrentNode.LocalUpperBound ≤ GlobalLowerBound then
      Continue; /*In this situation, we cannot find a better solution in the subtree of the current node, and prune*/
   else
      if (integer solution) then
         if (CurrentNode.LocalUpperBound = GlobalUpperBound) then
            OptSolObj ← CurrentNode.LocalUpperBound;
            register the optimal solution;
            algorithm ends;
         end if
         if (CurrentNode.LocalUpperBound < GlobalUpperBound and CurrentNode.LocalUpperBound >
         GlobalLowerBound) then
            GlobalLowerBound ← CurrentNode.LocalUpperBound;
            OptSolObj ← GlobalLowerBound;
            register the solution as the current optimal solution found;
         end if
      end if
      if (fractional solution) then
         branch to get LeftChildNode and RightChildNode;
         S ← Push(S, RightChildNode),S ← Push(S, LeftChildNode);
      end if
   end if
end while
```

sample approximation problem with a confidence level at least $(1-\epsilon)$ that is refereed to the sample confidence level. Normally, we set $1-\epsilon > 1-\alpha$. Afterwards, we reformulate the CCP model as below:

Let y_l for each scenario $w_l, w_l \in W$ be binary variables, $y_l = 0$ if the obtained solution must be feasible for scenario w_l and $y_l = 1$ otherwise.

Chance constraint (10) can be replaced by the following constraints:

$$\sum_{i \in T} \sum_{\substack{j \in T \cup \{t\} \\ j \neq i}} \sum_{k \in O} \omega_i \cdot \lambda_{ik}^l \cdot x_{ij}^k \geq -y_l M + \bar{f}, \forall w_l \in W \tag{25}$$

$$\sum_{w_l \in W} y_l \leq \lfloor |W| \cdot \epsilon \rfloor \tag{26}$$

$$y_l \in \{0,1\}, \forall w_l \in W \tag{27}$$

Constraints (25) state that the profits of the observations have to be larger than or equal to $\bar{f}$ for scenario w_l if $y_l = 0$, in which λ_{ik}^l is the value of stochastic variable $\tilde{\lambda}_{ik}$ under scenario w_l, and M is assumed to be a sufficiently large number, which can be set to the sum of the profits of all tasks. Constraint (26) imposes that the number of scenarios for which the solution is not necessarily feasible, which means the profits of the observations are not necessary to be larger than or equal to $\bar{f}$, will be at most $\lfloor |W| \cdot \epsilon \rfloor$. Thus, the sample approximation formulation of the original CCP problem is: maximize $\bar{f}$ subject to (2)-(8), (25)-(27).

5.2 Set packing model

Similarly with the expectation model, we also reformulate the sample approximation formulation as a master problem and some subproblems using Dantzig-Wolfe decomposition. Note that, the master problem is also a set packing problem, which only involves the unicity constraints and the sample related constraints.

To facilitate the reformulation, we add the following notations to those in Section 4.1:

c^l_{kr}: the profit of column r of orbit k under scenario w_l, $r \in R_k$, $k \in O$, $w_l \in W$.

The set packing model of the CCP problem can be formulated as follows:

$$\max \bar{f} \tag{28}$$

subject to

$$\sum_{k\in O}\sum_{r\in R_k}\sum_{\substack{j\in T\cup\{t\}\\ j\neq i}} \alpha_{ijrk} z_{kr} \leq 1, \forall i \in T \tag{29}$$

$$\sum_{r\in R_k} z_{kr} = 1, \forall k \in O \tag{30}$$

$$\bar{f} - y_l M - \sum_{k\in O}\sum_{r\in R_k} c^l_{kr} z_{kr} \leq 0, \forall w_l \in W \tag{31}$$

$$\sum_{w_l\in W} y_l \leq \lfloor |W| \cdot \epsilon \rfloor \tag{32}$$

$$z_{kr}, y_l \in \{0,1\}, \forall r \in R_k, k \in O, w_l \in W \tag{33}$$

The objective is to maximize the value of the real variable $\bar{f}$ that is constrained by the chance constraints. Constraints (29) are corresponding to constraints (2), which ensure that each task will be executed at most once. Constraints (30) are the convexity constraints representing that a feasible solution should be selected for each orbit. Constraints (31) state that the profits of the observations have to be larger than or equal to $\bar{f}$ for scenario w_l if $y_l = 0$. Constraint (32) is identical to constraint (26).

5.3 Column generation

Similarly with the expectation model, the above set packing model for chance constrained programming can also be solved by column generation to obtain the optimal solution of the LP relaxation. For the first iteration of the column generation, the set of initial columns R'_k, $R'_k \subseteq R_k$ for each orbit k can also be obtained by **Algorithm 1**, except that the weight for each node i will be modified as $\omega'_i = (\sum_{w_l\in W} \omega_i \lambda^l_i)/|W|$. For each 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 (29),
- δ_k: dual variables corresponding to constraints (30),
- ϕ_l: dual variables corresponding to constraints (31).

In this study, it is not necessary to consider the dual variable corresponding to constraint (32) since this constraint does not involve the z_{kr} variables and therefore the associated dual variable does not impact the reduced costs of the z_{kr} variables. On the basis of the dual variables, we can model and solve the subproblem for each orbit to add the column with the most positive reduced cost.

5.4 Subproblem

In order to find the absent column with the most positive reduced cost, the objective of subproblem k, $k \in O$ is defined as:

$$\max_{r\in R^k}\{\sum_{w_l\in W} c^l_{kr}\phi_l - \sum_{i\in T}\sum_{\substack{j\in T\cup\{t\}\\ j\neq i}} \alpha_{ijrk}\mu_i - \delta_k\} \tag{34}$$

Because each subproblem can be solved separately, we can remove the index k without provoking ambiguity. In addition, noting that $c^l_r = \sum_{i\in T}\sum_{\substack{j\in T\cup\{t\}\\ j\neq i}} \omega_i \lambda^l_i \alpha_{ijr}$, and we can formulate the objective as:

$$\max \sum_{i\in T}\sum_{\substack{j\in T\cup\{t\}\\ j\neq i}} (\sum_{w_l\in W} \omega_i \lambda^l_i \phi_l - \mu_i) \cdot \alpha_{ij} - \delta \tag{35}$$

Hence, with the notations similar to Section 4.3, we can formulate the subproblem as: maximize (35) subject to (19)-(24).

The above subproblem can also be solved using **Algorithm 2** with the modification that the weight for each node i should be $\sum_{w_l\in W} \omega_i \lambda^l_i \phi_l - \mu_i$.

Table 2 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

5.5 Column generation heuristic

As described previously, the optimal solutions obtained from the column generation algorithm normally are not integer feasible solutions. Intuitively, we also intended to use the branch-and-price algorithm to get the optimal integer solutions of the original problem. However, after experimental testing, we discovered that it is difficult and inappropriate to solve the chance constrained programming using the branch-and-price algorithm. The first reason is that in the RMP of the CCP model, not only z_{ik}, $i \in T, k \in O$, but also $y_l, w_l \in W$ are integer variables. Among the integer variables, branching on z_{ik}, $i \in T, k \in O$ that are associated with columns can be easily handled by the branching strategies: branching on the original variables and modifications of the weights of the relevant nodes (see Section 4.5). However, for the sample associated variables $y_l, w_l \in W$, branching will destroy the structure of the master problem, which will make the problem extremely difficult to be solved. Secondly, from the sample approximation formulation of the CCP model, we found that if we solve the LP relaxation of the set packing model (28)-(32), most (even all) variables $y_l, w_l \in W$ will be fractional, and one branching can only set one variable to be integer. Due to the large size of the sample, numerous branches will be required, which will result in the branch-and-price algorithm being much too time consuming.

Fortunately, some column generation heuristic algorithms can solve the complicated combinational optimization problems efficiently and provide a lower bound close to the optimal solution [1,18]. Hence, in order to get the integer feasible solutions of the CCP problem, we propose to solve the problem using a column generation based heuristic algorithm. It solved the integer programming model of the restricted master problem to find an integer solution based on the existing columns when we have solved the LP relaxation optimally. Clearly, the obtained integer feasible solution cannot be guaranteed to be optimal. However, we can get a "good" integer feasible solution in a short time even for large-scale problems.

The pseudocode of the column generation heuristic algorithm is outlined in **Algorithm 4**.

6 Computational results

For this section, we created a great number of problem instances in order to evaluate the effectiveness and efficiency of our proposed approaches. By the simulations, the performances of both the branch-and-price algorithm and the column generation heuristic algorithm are evaluated.

In order to verify the effectiveness and efficiency of our algorithms, 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 this work, 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 for each orbit are randomly generated in the intervals [200,240] and [240,320], respectively. Considering the uncertainties of clouds, for each time window of observation, the probability that there is no presence 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 12.3 API 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.

6.1 Performance evaluation of the branch-and-price algorithm

In this experiment, the number of tasks ranges from 20 to 180 with an increment of 20. 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. Hence, we will have 180 instances in total.

Before evaluating the performance of our branch-and-price algorithm, we need to set the parameters L and $ColNum$ reasonably in order to make the performance of the algorithm as best as possible. Table 3 shows the parameter testing results, in which only the solution times are given due to the fact that all in-

Algorithm 4 Column generation heuristic

```
Initialize a predefined number of columns with the maximum weights using Algorithm 1.
LP_opt_found ← FALSE;
while (LP_opt_found = FALSE) do
    LP_opt_found ← TRUE;
    CurrentNode.LocalUpperBound ← SOLVE-MASTER-LP(); /*solve the linear programming relaxation of the master problem*/
    for (k = 1, . . . , m) do
        RC_k ← FIND-NEW-COLUMN(k); /*solve the subproblem k with Algorithm 2 to get the maximum reduced cost of the absent columns*/
        if (RC_k > 0) then
            add the new column to the restricted master;
            LP_opt_found ← FALSE
        end if
    end for
end while
FeaSolObj ← SOLVE-MASTER(); /*solve the master problem based on the existing columns to obtain a "good" integer feasible solution*/
```

Table 3 Average solution times for different parameter settings of the branch-and-price algorithm

Parameter L	Parameter $ColNum$			
	1000	2000	3000	4000
5	4.447	4.871	**4.347**	6.661
10	4.481	5.278	5.062	6.667
15	5.130	6.064	5.263	8.707
20	5.776	5.909	6.777	8.939
25	6.810	6.042	7.461	10.368
30	6.054	7.190	6.305	9.483
35	6.293	9.225	7.240	11.061
40	7.692	9.520	8.005	9.861

stances are solved optimally with the same objective function values. Table 3 reveals that the branch-and-price algorithm will perform best with the parameter setting "$ColNum$ = 3000,L = 5". Hence, in the following experiments, we will adopt this setting for the branch-and-price algorithm.

To verify the superiority of the branch-and-price algorithm, we first compare its performance with that of CPLEX. In addition, there is also another intuitive approach to solve the expectation model. Using the Dantzig-Wolfe decomposition, we can construct the set packing model and some subproblems (see Section 4), and then we solve the set packing model directly to get integer solutions based on all columns or the maximum number (limited by computer memory) of columns that can be obtained by **Algorithm 1** for each orbit. In the subsequent text, we call this approach Dantzig-Wolfe decomposition based heuristic (DWDH) algorithm, and we will also compare the performance of our branch-and-price algorithm with that of the DWDH algorithm.

Table 4 shows the comparison results, in which columns "Obj" contain the average of the schedule objective values for the 10 instances, and columns "Time" contain the average values of the solution times. Besides, columns "(opt,fea)" show the number of instances that are solved optimally and the number of instances for which we can only get feasible solutions, respectively. Furthermore, more details for each problem instance are outlined in Tables 6 and 7 of the Appendix, in which the nonoptimal solutions are denoted in underlined numbers. According to the results in Table 4, we can conclude that both the proposed branch-and-price algorithm and CPLEX can derive the optimal solutions for all instances. Hence, the scheduling objective values are equivalent all the time. However, the branch-and-price algorithm is much faster to obtain the optimal solutions, which is more efficient compared with CPLEX. Additionally, it can be observed that the DWDH algorithm solves faster than our branch-and-price algorithm, especially for the larger problems (from 60 to 180 tasks). This is because the DWDH algorithm only solves the set packing model once to get the integer solution, without the iteration procedure for generating columns. Moreover, the DWDH can get all optimal solutions for small size problems (form 20 to 100 tasks), because there are not too many columns for each orbit, and we can easily enumerate all columns and solve the set packing problems optimally. However, with the number of tasks increasing, the number of columns for each orbit increases, and we cannot enumerate all columns for some instances due to the limitation of computer memory. Hence, for some instances, we cannot get the optimal solutions, as shown in Table 4.

6.2 Performance evaluation of the column generation heuristic algorithm

In this section, in order to evaluate the performance of the proposed column generation heuristic algorithm for

Table 4 Scheduling objective, solving time and the number of instances of (optimal, feasible) solutions

Number of orbits	Number of tasks	Branch-and-price			CPLEX			DWDH		
		Obj	Time	(opt,fea)	Obj	Time	(opt,fea)	Obj	Time	(opt,fea)
21	20	64.650	0.051	(10,0)	64.650	0.278	(10,0)	64.650	0.291	(10,0)
	40	134.740	0.121	(10,0)	134.740	0.624	(10,0)	134.740	0.315	(10,0)
	60	195.553	0.361	(10,0)	195.553	1.032	(10,0)	195.553	0.318	(10,0)
	80	265.357	1.012	(10,0)	265.357	1.614	(10,0)	265.357	0.297	(10,0)
	100	306.544	1.205	(10,0)	306.544	2.676	(10,0)	306.544	0.353	(10,0)
	120	370.447	3.438	(10,0)	370.447	4.170	(10,0)	370.285	0.711	(9,1)
	140	415.264	2.570	(10,0)	415.264	5.364	(10,0)	415.206	1.255	(9,1)
	160	451.481	4.421	(10,0)	451.481	9.592	(10,0)	450.509	2.047	(8,2)
	180	504.276	6.533	(10,0)	504.276	21.519	(10,0)	501.727	4.178	(3,7)
42	20	85.057	0.057	(10,0)	85.057	0.354	(10,0)	85.057	0.282	(10,0)
	40	179.776	0.259	(10,0)	179.776	0.818	(10,0)	179.776	0.267	(10,0)
	60	261.264	0.568	(10,0)	261.264	1.419	(10,0)	261.264	0.270	(10,0)
	80	353.237	1.003	(10,0)	353.237	2.393	(10,0)	353.237	0.416	(10,0)
	100	434.372	2.060	(10,0)	434.372	3.681	(10,0)	434.372	1.092	(10,0)
	120	519.401	2.848	(10,0)	519.401	6.213	(10,0)	519.401	1.746	(10,0)
	140	595.663	8.690	(10,0)	595.663	10.340	(10,0)	595.368	4.563	(7,3)
	160	695.135	9.085	(10,0)	695.135	14.727	(10,0)	692.669	7.790	(5,5)
	180	753.849	12.501	(10,0)	753.849	59.098	(10,0)	744.476	10.655	(0,10)

the CCP problem, we compare the solutions of CGH with those of the DWDH algorithm (see Section 6.1), the branch-and-cut algorithm in [55] and CPLEX. In this experiment, the number of tasks is set to 20, 40, 60, 80 and 100, and the number of orbits is set to 21 and 42, respectively. For each parameter setting, we also create 10 problem instances randomly. Additionally, the sample confidence level $1-\epsilon$ is set to 0.95, and the sample size $|W|$ is set to 100. Besides, the sample for each instance is randomly generated using Monte-Carlo simulation. According to the parameter testing results in Section 6.1, we will adopt the parameter setting "$ColNum = 3000, L = 5$" for the CGH algorithm.

Table 5 reveals the evaluation results, in which columns "M" and "N" represent the number of orbits and the number of tasks, respectively. Besides, columns "Obj", "Time" and "(opt,fea)" are identical to those in Table 4, and columns "GAP %" describe the average gap between the objective function values of the feasible solutions and those of the optimal solutions. Furthermore, with respect to the CGH algorithm, we can at least get feasible solutions for all instances. However, for the instances that are not included in (opt,fea), we cannot figure out whether the solutions are optimal or nonoptimal because we cannot get the optimal solutions. In contrast, with regard to the other algorithms, for the instances that are not included in (opt,fea), we cannot get feasible solutions due to out-of-memory problems. Note that, for most large-scale instances, the DWDH algorithm, the branch-and-cut algorithm and CPLEX fail to solve the instances due to out-of-memory problems. Therefore, we solve the instances of 80 tasks for 42 orbits with the CGH algorithm only and we do not solve the instances of 100 tasks for 42 orbits. Furthermore, more details for each problem instance are outlined in Tables 8 and 9 of the Appendix, in which the nonoptimal solutions are denoted in underlined numbers.

From Tables 5, we can conclude that the CGH can get "good" feasible solutions in reasonable times for most instances. Additionally, although CPLEX and the branch-and-cut algorithm can get the optimal solutions, the solution times will increase explosively. Especially, for some large-scale problems, no feasible solutions can be obtained due to the limit of computer memory. Fortunately, using our proposed column generation heuristic algorithm, we can solve the problems very efficiently, and at least the feasible solutions can be obtained. Furthermore, for the small or medium size instances, the DWDH gets better solutions that are mostly the optimal solutions, compared with the CGH algorithm. However, for the large-scale instances, the DWDH algorithm performs worse, as it will cost much more time to get the optimal solutions than the branch-and-cut algorithm and CPLEX. Even worse, for some larger problem instances (100 tasks for 21 orbits and 60 tasks for 42 orbits), the DWDH algorithm cannot get feasible solutions due to the limited memory. For the comparisons between CPLEX and the branch-and-cut algorithm, for the small or medium size instances (less than 80 tasks for 21 orbits), the solution times of the branch-and-cut algorithm are less than those of CPLEX. On the contrary, for the large-scale instances (100 tasks for 21 orbits and all the instances for 42 orbits), the branch-and-cut algorithm will be more time consuming.

Table 5 Scheduling objective, solving time and the number of instances of (optimal, feasible) solutions

M	N	CGH				DWDH				CPLEX			Branch-and-cut		
		Obj	Time	(opt,fea)	GAP %	Obj	Time	(opt,fea)	GAP %	Obj	Time	(opt,fea)	Obj	Time	(opt,fea)
21	20	53	1.717	(10,0)	0.00	53	1.631	(10,0)	0.00	53	1.557	(10,0)	53	1.406	(10,0)
	40	116.9	5.681	(7,3)	0.25	117.2	20.769	(10,0)	0.00	117.2	16.551	(10,0)	117.2	9.536	(10,0)
	60	171.4	10.418	(6,4)	1.45	173.8	107.145	(10,0)	0.00	173.8	413.407	(10,0)	173.8	75.992	(10,0)
	80	233.5	15.614	(0,10)	2.88	240	339.581	(9,1)	0.09	240.2	431.491	(10,0)	240.2	396.614	(10,0)
	100	266	14.375	(0,10)	3.33	262.7	486.118	(2,5)	5.70	275	545.930	(10,0)	275	1962.470	(10,0)
42	20	73.6	29.756	(10,0)	0.00	73.6	87.809	(10,0)	0.00	73.6	42.180	(10,0)	73.6	64.013	(10,0)
	40	162.7	195.981	(2,8)	1.47	165.1	1200.381	(10,0)	0.00	165.1	243.092	(10,0)	165.1	370.754	(10,0)
	60	241.5	2416.464	(0,8)	1.73	243.25	847.652	(4,0)	0.00	248.38	2052.772	(8,0)	248.38	2437.500	(8,0)
	80	306.3	3096.078												

7 Conclusions and future work

In this paper, considering the uncertainties of clouds, we formulated the presence of clouds for observations as stochastic events, and then investigated the scheduling of multiple EOSs. Based on different viewpoints, we proposed both an expectation model and a chance constrained programming model to formulate the problem. With respect to the expectation model, we devise an exact algorithm based on branch-and-price to solve the model optimally and efficiently. On the other hand, because of the difficulties and the infeasibility of the branch-and-price algorithm for solving the CCP model, a column generation based heuristic algorithm is designed to obtain "good" feasible solutions efficiently. To the best of our knowledge, this is the first study that suggests exact and heuristic algorithms based on column generation to solve the scheduling of multiple EOSs under uncertainties of clouds. Finally, by a great number of simulation experiments, we proved that: 1) the branch-and-price algorithm can solve the expectation model optimally for all generated instances and this faster than CPLEX; 2) for the CCP model, the column generation heuristic algorithm can get close-to-optimal solutions for all instances efficiently.

In the future, the first extension of our research is to consider the scheduling of agile satellites 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 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. In addition, for a unique window, the impact of clouds for different parts will be different, which will make the problem more complicated. Moreover, besides the proactive scheduling, we will also consider developing reactive scheduling algorithms for EOSs. Currently, most of the new generations of EOSs will be equipped with cloud detection instruments [3] that can detect the status of clouds for specified areas before observation. Hence, the scheduling decisions can be made according to the detection results on board. However, the time interval between detection and observation will be very short, thus the decision should be made very quickly. Hence, it will be important to design highly efficient reactive scheduling algorithms. Further, we will also consider incorporating some more sophisticated techniques such as dual stabilization and dynamic constraint aggregation to speed up the convergence of the column generation process.

Acknowledgments

This research has been supported by the China Scholarship Council. We would like to acknowledge the China Scholarship Council for the financial support and the Research Center for Operations Management of the KU Leuven for providing a visiting research period to Jianjiang Wang. This research has also been supported by the National Natural Science Foundation of China under Grants No. 61104180, and No. 71271213.

References

1. Baldacci, R., Mingozzi, A., & Roberti, R. (2012). Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints. *European Journal of Operational Research*, *218*(1), 1-6.
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. Belien, J. (2006). Exact and heuristic methodologies for scheduling in hospitals: problems, formulations and algorithms. PhD thesis, Katholieke Universiteit Leuven, Belgium.
5. 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.
6. 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).
7. 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).
8. 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.*
9. 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.*
10. 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.
11. Bianchessi, N., & Righini, G. (2008). Planning and scheduling algorithms for the COSMO-SkyMed constellation. *Aerospace Science Technology*, *12*(7), 535-544.

12. 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.
13. 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.
14. 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.
15. Charnes, A., & Cooper, W.W. (1959). Chance-constrained programming. *Management Science*, *6*(1), 73-79.
16. Cordeau, J., & Laporte, G. (2005). Maximizing the value of an earth observation satellite orbit. *Journal of the Operational Research Society*, *56*(8), 962-968.
17. 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.
18. 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.
19. Frank, J., Jónsson, A., Morris, R., & Smith, D.E. (2001). Planning and scheduling for fleets of earth observing satellites. In *Proceedings of the 6th international symposium on artificial intelligence, robotics, automation and space.*
20. 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.
21. 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.
22. Gabrel, V. (2006). Strengthened 0-1 linear formulation for the daily satellite mission planning. *Journal of Combinatorial Optimization. 11*(3), 341-346.
23. Globus, A., Crawford, J., Lohn, J., & Morris, R. (2002). Scheduling earth observing fleets using evolutionary algorithms: problem description and approach. In *Proceedings of the 3rd international NASA workshop on planning and scheduling for space.*
24. 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.
25. 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.
26. 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.
27. 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.*
28. 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.
29. 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.
30. 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.
31. 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.
32. Lin, W., & Chang, S. (2005). Hybrid algorithms for satellite imaging scheduling. In *Proceedings of the IEEE international conference on systems, man and cybernetics* (pp. 2518-2523).
33. 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.
34. Liu, B. (1999). *Uncertain Programming.* New York: Wiley.
35. Liu, B., & Liu, B. (2002). *Theory and practice of uncertain programming.* Heidelberg: Physica-verlag.
36. Luedtke, J., & Ahmed, S. (2008). A sample approximation approach for optimization with probabilistic constraints. *SIAM Journal on Optimization*, *19*(2), 674-699.
37. Luedtke, J., Ahmed, S., & Nemhauser, G.L. (2010). An integer programming approach for linear programs with probabilistic constraints. *Mathematical Programming*, *122*(2), 247-272.
38. Mancel, C., & Lopez, P. (2003). Complex optimization problems in space systems. In *Proceedings of the 13th international conference on automated planning and scheduling.*
39. 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.
40. Montoya, C., Morineau, O.B., Pinson, E., & Rivreau, D. (2014). Branch-and-price approach for the multi-skill project scheduling problem. *Optimation Letters*, *8*(5), 1721-1734.
41. Pralet, C., Verfaillie, G., Maillard, A., et al. (2014). Satellite data download management with uncertainty about the generated volumes. In *Proceedings of the 24th international conference on automated planning and scheduling.*
42. Robenek, T., Umang, N., Bierlaire, M., & Ropke, S. (2014). A branch-and-price algorithm to solve the integrated berth allocation and yard assignment problem in bulk ports. *European Journal of Operational Research*, *235*(2), 399-411.
43. Ruszczyński, A. (2002). Probabilistic programming with discrete distributions and precedence constrained knapsack polyhedra. *Mathematical Programming*, *93*(2), 195-215.
44. Savelsbergh, M. (1997). A branch-and-price algorithm for the generalized assignment problem. *Operations Research*, *45*(6), 831-841.
45. 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*.
46. 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*.
47. 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.
48. 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*, doi: 10.1016/j.ejor.2015.03.011
49. 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.
50. 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.
51. Vasquez, M., & Hao, J. (2003). Upper bounds for the SPOT 5 daily photograph scheduling problem. *Journal of Combinatorial Optimization*, *7*(1), 87-103.
52. 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).
53. 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.
54. 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.
55. Wang, J., Demeulemeester, E., & Qiu, D. (2014). A pure proactive scheduling algorithm for multiple earth observation satellites under uncertainties of clouds. *Computers Operations Research*, submitted.
56. Wang, J., Demeulemeester, E., Qiu, D., & Liu, J. (2015). Exact and inexact scheduling algorithms for multiple earth observation satellites under uncertainties of clouds. *European Journal of Operational Research*, submitted.
57. Wang, P., & Reinelt, G. (2011). Solving the earth observing satellite constellation scheduling problem by branch-and-price. *In Operations Research Proceedings 2010* (pp. 491-496).
58. Wolfe, J., & Stephen, S.E. (2000). Three scheduling algorithms applied to the earth observing systems domain. *Management Science*, *46*(1), 148-168.
59. 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.
60. Zufferey, N., Amstutz, P., & Giaccari, P. (2008). Graph colouring approaches for a satellite range scheduling problem. *Journal of Scheduling*, *11*(4), 263-277.

Appendix

Table 6: Branch-and-price: scheduling objectives and solution times for 21 orbits

Number of tasks	Instance No.	Branch-and-price Obj	Branch-and-price Time	CPLEX Obj	CPLEX Time	DWDH Obj	DWDH Time
20	0	85.189	0.032	85.189	0.273	85.189	0.228
	1	52.442	0.061	52.442	0.319	52.442	0.346
	2	54.731	0.053	54.731	0.219	54.731	0.125
	3	67.887	0.048	67.887	0.246	67.887	0.348
	4	71.161	0.040	71.161	0.324	71.161	0.369
	5	93.917	0.046	93.917	0.224	93.917	0.127
	6	55.590	0.060	55.590	0.341	55.590	0.403
	7	53.703	0.053	53.703	0.246	53.703	0.272
	8	58.466	0.056	58.466	0.285	58.466	0.366
	9	53.413	0.056	53.413	0.298	53.413	0.328
	Average	64.650	0.051	64.650	0.278	64.650	0.291
40	0	137.882	0.057	137.882	0.807	137.882	0.257
	1	147.874	0.060	147.874	0.564	147.874	0.361
	2	142.777	0.369	142.777	0.676	142.777	0.321
	3	115.471	0.049	115.471	0.621	115.471	0.414
	4	134.805	0.055	134.805	0.573	134.805	0.345
	5	147.542	0.076	147.542	0.583	147.542	0.260
	6	126.355	0.146	126.355	0.630	126.355	0.172
	7	128.986	0.086	128.986	0.534	128.986	0.282
	8	118.005	0.046	118.005	0.639	118.005	0.373
	9	147.706	0.268	147.706	0.610	147.706	0.360
	Average	134.740	0.121	134.740	0.624	134.740	0.315
60	0	196.728	0.270	196.728	0.919	196.728	0.241
	1	221.495	0.327	221.495	0.873	221.495	0.338
	2	163.606	0.198	163.606	0.900	163.606	0.308
	3	185.217	0.324	185.217	1.023	185.217	0.323
	4	190.113	0.247	190.113	0.837	190.113	0.292
	5	173.637	0.541	173.637	0.972	173.637	0.364
	6	175.067	0.051	175.067	0.872	175.067	0.179
	7	185.011	0.776	185.011	1.436	185.011	0.516
	8	211.800	0.080	211.800	1.095	211.800	0.374
	9	252.853	0.795	252.853	1.388	252.853	0.242
	Average	195.553	0.361	195.553	1.032	195.553	0.318
80	0	278.899	0.677	278.899	1.531	278.899	0.252
	1	299.610	3.502	299.610	1.968	299.610	0.291
	2	276.498	0.825	276.498	1.755	276.498	0.208
	3	249.193	0.599	249.193	1.311	249.193	0.254
	4	236.116	1.813	236.116	1.339	236.116	0.317
	5	251.623	0.718	251.623	1.762	251.623	0.369
	6	241.594	0.601	241.594	1.559	241.594	0.482
	7	300.114	0.617	300.114	1.513	300.114	0.210
	8	241.515	0.365	241.515	1.511	241.515	0.326
	9	278.403	0.400	278.403	1.895	278.403	0.262
	Average	265.357	1.012	265.357	1.614	265.357	0.297
100	0	344.398	2.314	344.398	2.756	344.398	0.426
	1	315.877	1.218	315.877	3.175	315.877	0.679
	2	298.680	0.926	298.680	2.790	298.680	0.369
	3	306.887	1.151	306.887	2.394	306.887	0.229
	4	281.197	1.164	281.197	1.738	281.197	0.220
	5	305.543	2.008	305.543	2.827	305.543	0.338
	6	282.026	1.185	282.026	2.698	282.026	0.266
	7	352.937	1.002	352.937	2.904	352.937	0.298
	8	299.303	0.479	299.303	3.136	299.303	0.316
	9	278.596	0.604	278.596	2.343	278.596	0.391
	Average	306.544	1.205	306.544	2.676	306.544	0.353
120	0	330.243	0.571	330.243	3.399	330.243	0.466
	1	418.603	16.584	418.603	5.607	<u>416.983</u>	2.656
	2	366.373	0.846	366.373	3.055	366.373	0.378
	3	395.652	1.631	395.652	3.519	395.652	0.907
	4	394.852	1.181	394.852	3.772	394.852	0.366
	5	355.232	1.376	355.232	4.508	355.232	0.429
	6	357.197	6.318	357.197	4.648	357.197	0.531
	7	385.708	1.012	385.708	4.223	385.708	0.264

	8	337.870	3.776	337.870	4.321	337.870	0.778
	9	362.738	1.082	362.738	4.643	362.738	0.333
	Average	370.447	3.438	370.447	4.170	370.285	0.711
140	0	424.157	0.890	424.157	4.566	424.157	0.859
	1	452.395	2.361	452.395	5.005	452.395	2.310
	2	441.600	2.388	441.600	5.175	441.600	1.681
	3	413.634	8.057	413.634	7.616	413.634	1.822
	4	379.223	1.005	379.223	4.798	379.223	1.100
	5	445.239	2.849	445.239	5.373	445.239	1.089
	6	391.512	1.114	391.512	5.224	391.512	0.965
	7	387.088	1.163	387.088	5.551	387.088	0.301
	8	413.825	2.656	413.825	4.538	413.825	1.027
	9	403.971	3.215	403.971	5.795	403.390	1.400
	Average	415.264	2.570	415.264	5.364	415.206	1.255
160	0	464.522	1.072	464.522	5.707	464.522	1.208
	1	441.268	20.525	441.268	10.856	438.172	2.013
	2	469.092	3.063	469.092	10.028	469.092	1.889
	3	467.692	4.397	467.692	7.757	467.692	1.895
	4	415.887	3.232	415.887	8.939	415.887	1.591
	5	431.997	4.339	431.997	7.764	431.997	1.107
	6	447.246	1.541	447.246	9.422	447.246	0.960
	7	455.412	1.332	455.412	11.034	455.412	2.156
	8	464.610	3.016	464.610	18.806	457.992	6.081
	9	457.079	1.691	457.079	5.608	457.079	1.571
	Average	451.481	4.421	451.481	9.592	450.509	2.047
180	0	495.112	6.480	495.112	18.028	492.447	4.570
	1	528.673	1.050	528.673	7.741	527.499	2.823
	2	533.262	2.909	533.262	7.514	522.832	11.092
	3	498.317	1.443	498.317	15.226	497.944	3.401
	4	511.670	10.077	511.670	20.535	511.670	4.711
	5	512.192	7.567	512.192	96.956	508.054	2.793
	6	500.597	27.907	500.597	12.596	500.597	3.167
	7	484.206	2.736	484.206	13.008	478.806	4.680
	8	473.988	1.366	473.988	14.290	472.676	2.465
	9	504.744	3.796	504.744	9.298	504.744	2.075
	Average	504.276	6.533	504.276	21.519	501.727	4.178

Table 7: Branch-and-price: scheduling objectives and solution times for 42 orbits

Number of tasks	Instance No.	Branch-and-price		CPLEX		DWDH	
		Obj	Time	Obj	Time	Obj	Time
20	0	82.141	0.039	82.141	0.408	82.141	0.374
	1	93.639	0.049	93.639	0.346	93.639	0.307
	2	94.820	0.059	94.820	0.380	94.820	0.284
	3	86.304	0.064	86.304	0.287	86.304	0.248
	4	73.268	0.053	73.268	0.321	73.268	0.313
	5	85.736	0.069	85.736	0.378	85.736	0.234
	6	91.975	0.054	91.975	0.370	91.975	0.348
	7	81.353	0.064	81.353	0.344	81.353	0.207
	8	75.932	0.056	75.932	0.360	75.932	0.248
	9	85.403	0.065	85.403	0.345	85.403	0.256
	Average	85.057	0.057	85.057	0.354	85.057	0.282
40	0	179.332	0.209	179.332	0.700	179.332	0.179
	1	185.616	0.055	185.616	0.769	185.616	0.274
	2	181.413	0.543	181.413	0.897	181.413	0.246
	3	165.315	0.220	165.315	0.926	165.315	0.231
	4	167.618	0.355	167.618	0.757	167.618	0.305
	5	170.068	0.108	170.068	0.812	170.068	0.264
	6	192.601	0.556	192.601	0.773	192.601	0.336
	7	198.124	0.413	198.124	0.749	198.124	0.389
	8	183.284	0.072	183.284	0.942	183.284	0.258
	9	174.388	0.054	174.388	0.858	174.388	0.191
	Average	179.776	0.259	179.776	0.818	179.776	0.267
	0	235.424	0.702	235.424	1.368	235.424	0.229
	1	261.037	0.484	261.037	1.425	261.037	0.264
	2	269.794	1.026	269.794	1.479	269.794	0.277
	3	243.804	0.230	243.804	1.380	243.804	0.279
	4	264.553	0.555	264.553	1.309	264.553	0.220
60							

	5	255.099	0.462	255.099	1.501	255.099	0.216
	6	274.514	0.277	274.514	1.388	274.514	0.233
	7	251.616	0.372	251.616	1.547	251.616	0.346
	8	276.225	0.401	276.225	1.328	276.225	0.345
	9	280.573	1.174	280.573	1.466	280.573	0.286
	Average	261.264	0.568	261.264	1.419	261.264	0.270
80	0	384.777	0.846	384.777	2.302	384.777	0.505
	1	324.193	0.878	324.193	2.604	324.193	0.421
	2	345.339	1.370	345.339	2.325	345.339	0.469
	3	361.666	0.909	361.666	2.450	361.666	0.389
	4	324.252	1.159	324.252	2.342	324.252	0.295
	5	350.838	1.246	350.838	2.242	350.838	0.381
	6	345.012	0.768	345.012	2.353	345.012	0.481
	7	365.825	0.951	365.825	2.522	365.825	0.313
	8	373.299	1.132	373.299	2.404	373.299	0.416
	9	357.168	0.771	357.168	2.386	357.168	0.488
	Average	353.237	1.003	353.237	2.393	353.237	0.416
100	0	429.873	1.739	429.873	3.264	429.873	0.911
	1	459.140	1.974	459.140	3.290	459.140	0.753
	2	433.824	1.409	433.824	3.599	433.824	0.890
	3	462.789	1.580	462.789	3.084	462.789	0.860
	4	458.231	2.491	458.231	3.373	458.231	0.863
	5	393.861	1.450	393.861	5.083	393.861	0.908
	6	430.496	2.406	430.496	3.392	430.496	1.209
	7	447.369	3.340	447.369	4.370	447.369	2.454
	8	397.893	1.957	397.893	3.824	397.893	1.061
	9	430.244	2.254	430.244	3.532	430.244	1.012
	Average	434.372	2.060	434.372	3.681	434.372	1.092
120	0	512.406	2.493	512.406	4.645	512.406	2.555
	1	467.842	2.768	467.842	6.166	467.842	1.573
	2	525.099	2.751	525.099	6.146	525.099	2.361
	3	491.345	2.374	491.345	7.221	491.345	1.143
	4	523.082	3.007	523.082	7.104	523.082	1.989
	5	485.355	2.677	485.355	8.371	485.355	1.731
	6	557.310	3.212	557.310	5.699	557.310	1.148
	7	561.606	2.760	561.606	5.029	561.606	1.464
	8	563.867	3.157	563.867	6.078	563.867	1.792
	9	506.096	3.276	506.096	5.674	506.096	1.708
	Average	519.401	2.848	519.401	6.213	519.401	1.746
140	0	584.946	3.162	584.946	7.551	584.946	4.375
	1	592.659	3.229	592.659	11.298	592.659	3.874
	2	579.876	7.997	579.876	8.244	579.852	3.560
	3	587.649	5.138	587.649	8.832	587.649	3.596
	4	622.100	11.798	622.100	11.971	622.100	5.500
	5	608.550	6.053	608.550	9.533	608.550	8.208
	6	624.674	18.905	624.674	9.955	624.120	4.643
	7	589.878	16.334	589.878	11.521	589.878	4.575
	8	556.978	11.009	556.978	13.228	554.605	4.683
	9	609.320	3.275	609.320	11.264	609.320	2.612
	Average	595.663	8.690	595.663	10.340	595.368	4.563
160	0	677.001	5.276	677.001	15.768	677.001	7.335
	1	721.612	11.136	721.612	16.479	721.612	7.341
	2	698.823	6.269	698.823	10.586	697.885	7.004
	3	717.109	8.556	717.109	16.551	717.109	12.686
	4	707.563	8.950	707.563	20.118	706.207	9.233
	5	643.955	20.183	643.955	11.795	643.955	10.321
	6	687.957	11.196	687.957	15.423	673.578	5.492
	7	707.804	5.550	707.804	11.608	707.804	5.425
	8	696.164	5.027	696.164	12.655	690.526	7.632
	9	693.362	8.706	693.362	16.291	691.009	5.432
	Average	695.135	9.085	695.135	14.727	692.669	7.790
180	0	771.124	26.368	771.124	351.992	766.928	15.632
	1	738.813	4.303	738.813	31.596	736.901	9.305
	2	810.086	19.715	810.086	23.149	800.439	10.136
	3	762.752	11.281	762.752	22.356	747.790	8.689
	4	770.516	11.810	770.516	36.185	765.229	13.991
	5	752.441	17.867	752.441	22.538	739.037	12.253
	6	728.828	7.763	728.828	20.165	719.788	8.897
	7	649.328	6.117	649.328	14.811	636.985	7.168
	8	788.548	16.100	788.548	35.338	773.126	12.614
	9	766.049	3.681	766.049	32.849	758.532	7.865

	Average	753.849	12.501	753.849	59.098	744.476	10.655

Table 8: CGH: scheduling objectives and solution times for 21 orbits

Number of tasks	Instance No.	CGH			DWDH			CPLEX		Branch-and-cut	
		Obj	Time	GAP %	Obj	Time	GAP %	Obj	Time	Obj	Time
20	0	73	1.991	0.00	73	1.906	0.00	73	2.419	73	2.167
	1	43	0.505	0.00	43	0.373	0.00	43	0.588	43	0.573
	2	41	0.641	0.00	41	0.556	0.00	41	0.684	41	0.495
	3	57	1.461	0.00	57	1.325	0.00	57	1.247	57	0.902
	4	59	0.740	0.00	59	0.656	0.00	59	0.995	59	0.643
	5	76	0.594	0.00	76	0.535	0.00	76	0.563	76	0.631
	6	46	5.880	0.00	46	5.814	0.00	46	4.891	46	4.724
	7	45	1.877	0.00	45	1.860	0.00	45	1.405	45	1.558
	8	48	1.761	0.00	48	1.679	0.00	48	1.283	48	1.430
	9	42	1.715	0.00	42	1.604	0.00	42	1.490	42	0.932
	Average	53	1.717	0.00	53	1.631	0.00	53	1.557	53	1.406
40	0	120	5.872	0.00	120	7.943	0.00	120	7.546	120	4.794
	1	132	6.074	0.00	132	7.258	0.00	132	12.853	132	16.227
	2	128	18.383	0.00	128	113.550	0.00	128	69.841	128	40.753
	3	98	2.867	0.00	98	4.258	0.00	98	3.866	98	1.555
	4	114	1.732	0.00	114	1.567	0.00	114	1.552	114	1.669
	5	126	3.973	0.79	127	5.950	0.00	127	6.848	127	5.091
	6	106	7.041	0.93	107	46.767	0.00	107	40.790	107	15.875
	7	112	2.284	0.00	112	3.744	0.00	112	4.145	112	1.468
	8	104	3.403	0.00	104	4.926	0.00	104	5.199	104	2.380
	9	129	5.177	0.77	130	11.731	0.00	130	12.868	130	5.546
	Average	116.9	5.681	0.25	117.2	20.769	0.00	117.2	16.551	117.2	9.536
60	0	180	13.195	0.00	180	37.635	0.00	180	43.127	180	216.074
	1	194	5.204	2.02	198	21.703	0.00	198	21.676	198	26.309
	2	145	5.302	2.68	149	17.842	0.00	149	10.365	149	3.986
	3	165	7.977	0.00	165	27.258	0.00	165	27.822	165	15.593
	4	170	15.801	0.00	170	27.257	0.00	170	28.080	170	40.577
	5	150	26.711	5.66	159	79.636	0.00	159	68.780	159	147.520
	6	157	10.596	0.00	157	39.101	0.00	157	41.042	157	25.575
	7	162	8.747	4.14	169	789.748	0.00	169	3855.450	169	253.252
	8	190	5.175	0.00	190	10.572	0.00	190	15.362	190	14.438
	9	201	5.476	0.00	201	20.697	0.00	201	22.364	201	16.594
	Average	171.4	10.418	1.45	173.8	107.145	0.00	173.8	413.407	173.8	75.992
80	0	257	30.193	0.39	258	314.013	0.00	258	458.391	258	652.555
	1	243	15.934	1.62	247	125.579	0.00	247	228.871	247	149.294
	2	254	24.889	1.55	258	622.660	0.00	258	1006.310	258	187.044
	3	219	8.097	3.95	228	43.290	0.00	228	34.741	228	9.032
	4	198	9.266	8.33	216	145.485	0.00	216	167.248	216	61.464
	5	216	4.618	4.42	224	106.361	0.88	226	136.157	226	26.583
	6	216	13.894	0.92	218	77.992	0.00	218	55.704	218	139.679
	7	271	8.573	1.45	275	56.521	0.00	275	51.180	275	50.964
	8	216	26.152	2.26	221	249.142	0.00	221	185.598	221	371.529
	9	245	14.525	3.92	255	1654.770	0.00	255	1990.710	255	2318.000
	Average	233.5	15.614	2.88	240	339.581	0.09	240.2	431.491	240.2	396.614
100	0	317	22.788	1.55	297	302.120	7.76	322	440.876	322	753.468
	1	283	8.391	2.41	-	-	-	290	1049.260	290	664.961
	2	272	15.591	0.73	272	92.144	0.73	274	253.277	274	422.815
	3	265	6.504	5.69	281	309.532	0.00	281	205.237	281	25.737
	4	253	16.388	1.56	233	1153.790	9.34	257	658.817	257	1156.200
	5	268	8.478	3.25	262	291.990	5.42	277	280.450	277	276.502
	6	248	19.936	3.88	215	1095.160	16.67	258	823.305	258	1300.600
	7	252	14.595	1.18	-	-	-	255	1280.720	255	14594.300
	8	272	18.460	2.51	279	158.091	0.00	279	227.410	279	166.140
	9	230	12.620	10.51	-	-	-	257	239.944	257	263.980
	Average	266	14.375	3.33	262.714	486.118	5.70	275	545.930	275	1962.470

Table 9: CGH: scheduling objectives and solution times for 42 orbits

Number of tasks	Instance No.	CGH			DWDH			CPLEX		Branch-and-cut	
		Obj	Time	GAP %	Obj	Time	GAP %	Obj	Time	Obj	Time
20	0	74	12.131	0.00	74	22.428	0.00	74	14.489	74	21.991
	1	81	4.145	0.00	81	4.016	0.00	81	4.226	81	4.843
	2	83	9.360	0.00	83	9.318	0.00	83	9.025	83	12.846
	3	73	4.181	0.00	73	4.011	0.00	73	5.487	73	9.523
	4	63	4.323	0.00	63	4.203	0.00	63	3.698	63	4.446
	5	75	17.286	0.00	75	19.252	0.00	75	18.918	75	67.237
	6	80	223.835	0.00	80	795.389	0.00	80	344.195	80	481.045
	7	66	5.254	0.00	66	5.157	0.00	66	6.438	66	9.081
	8	65	5.996	0.00	65	4.875	0.00	65	6.859	65	12.291
	9	76	11.050	0.00	76	9.437	0.00	76	8.467	76	16.824
	Average	73.6	29.756	0.00	73.6	87.809	0.00	73.6	42.180	73.6	64.013
40	0	167	71.339	0.00	167	863.867	0.00	167	179.806	167	446.879
	1	170	22.120	0.58	171	35.880	0.00	171	21.200	171	28.252
	2	161	453.900	1.23	163	1856.870	0.00	163	511.442	163	635.959
	3	146	17.912	3.31	151	40.826	0.00	151	28.043	151	20.940
	4	151	182.232	2.58	155	206.825	0.00	155	100.401	155	135.448
	5	157	101.285	0.63	158	83.307	0.00	158	62.440	158	41.600
	6	173	106.300	1.70	176	2208.530	0.00	176	713.506	176	588.397
	7	181	17.699	2.16	185	401.512	0.00	185	167.803	185	226.633
	8	159	329.845	2.45	163	229.994	0.00	163	99.504	163	399.091
	9	162	657.175	0.00	162	6076.200	0.00	162	546.776	162	1184.340
	Average	162.7	195.981	1.47	165.1	1200.381	0.00	165.1	243.092	165.1	370.754
60	0	217	21931.900	-	-	-	-	-	-	-	-
	1	242	39.949	1.22	245	234.491	0.00	245	57.817	245	140.723
	2	237	202.123	5.20	-	-	-	250	1036.100	250	616.951
	3	232	495.250	1.69	236	1089.110	0.00	236	683.180	236	486.577
	4	244	42.932	3.17	252	126.548	0.00	252	206.201	252	123.849
	5	238	49.873	0.83	240	1940.460	0.00	240	786.109	240	2320.240
	6	254	147.829	-	-	-	-	-	-	-	-
	7	232	571.249	2.93	-	-	-	239	2602.840	239	3196.570
	8	257	560.283	0.39	-	-	-	258	3614.520	258	2962.360
	9	262	123.248	1.87	-	-	-	267	7435.410	267	9652.730
	Average	241.5	2416.464	1.73	243.25	847.652	0.00	248.375	2052.772	248.375	2437.500
80	0	354	1160.630								
	1	273	32.233								
	2	290	373.435								
	3	309	15065.100								
	4	287	622.822								
	5	288	388.124								
	6	303	12898.600								
	7	325	154.644								
	8	324	101.593								
	9	310	163.596								
	Average	306.3	3096.078								

FACULTY OF ECONOMICS AND BUSINESS
Naamsestraat 69 bus 3500
3000 LEUVEN, BELGIË
tel. + 32 16 32 66 12
fax + 32 16 32 67 91
info@econ.kuleuven.be
www.econ.kuleuven.be