Solving the Agile Earth Observation Satellite Scheduling Problem with Time-Dependent Transition Times

Guansheng Peng, Guopeng Song, Yongming He, Jing Yu, Shang Xiang, Lining Xing and Pieter Vansteenwegen

Abstract—The scheduling of Agile Earth Observation Satellites is to select a subset of candidate targets each associated with a profit during their visible time windows in order to maximize the collected profits, under some operational constraints. For each pair of two consecutive observations, a transition time is required to perform a rotating movement of the camera, depending on the start times of the two observations. This time-dependency significantly increases the complexity of the scheduling problem. To solve this problem efficiently, we model the time-dependent transition time and prove that it satisfies the First-In-First-Out rule and the triangle inequalities rule. On this basis, we develop a novel hybrid heuristic, called “Greedy Randomized Iterated Local Search” (GRILS). A specific insert operator including a fast feasibility check and an assignment procedure are specifically designed to address the operational constraints of the scheduling. Extensive experiments on the single satellite instances and multi-satellite instances demonstrate that our algorithm outperforms the state-of-the-art algorithms with respect to solution quality and computation time.

Index Terms—time-dependent transition time, satellite scheduling, GRILS heuristic.

I. Introduction

THE mission of an Earth Observation Satellite (EOS) is to photograph targets on the surface of the Earth, according to the requests from users. EOSs play an increasingly crucial role in many applications, such as resource exploration, disaster alerts, climate change analysis and so on [1]. Due to the huge cost of satellites, the scheduling of EOSs is of great importance for the effectiveness of the observation systems.

The conventional EOS only has one degree of freedom (roll), and is therefore called a “non-agile” satellite. As can be seen in Fig. 1(a), it can only observe the target when it flies exactly over the target, which is also called the “nadir point”. The Observation Window (OW) refers to the duration when the event of taking images (called an “observation”) is processed. The Visible Time Window

(VTW) represents the time window in which it is possible to observe the corresponding target. For the non-agile EOS, the length of the OW of any target is equal to the length of its VTW. Therefore, the only decision of scheduling an EOS is the selection of the images [2]. In contrast, the new generation of EOSs, Agile Earth Observation Satellites (AEOS) are mobile on three axes (roll, pitch and yaw), thus allowing observations before or after its upright pass (the nadir point) [3]. As can be seen in Fig. 1(b), the VTW of the given target is much longer than its observation duration. In this case, the start time for each observation should be determined, and thus the solution space is much larger than that of EOS scheduling. This agility increases the number of targets that can be observed during a period, but also bring a significant challenge for scheduling.

Fig. 1. Comparison of (a) non-agile EOS and (b) AEOS when observing two targets.

Scheduling AEOSs is typically an oversubscribed resource scheduling problem where only a subset of targets can be scheduled due to some operational constraints [4]. The objective of this problem is to maximize the total profits associated with the observations of candidate targets, while satisfying some practical constraints. Given a long scheduling horizon, an AEOS may circle the earth many times and thus can observe the same target during different orbits. An orbit is defined as the time period that the satellite flies through the sunshine area while circling the earth one time. Therefore, each target may have

This study is supported by the National Natural Science Foundation of China (61773120, 61873328), and the National Natural Science Fund for Distinguished Young Scholars of China (61525304). It is also supported by the Foundation for the Author of National Excellent Doctoral Dissertation of China (2014-92). (Corresponding author: Lining Xing)

P. Vansteenwegen is with the KU Leuven Mobility Research Center - CIB, Leuven, 3001, Belgium. The Other authors are with the College of Systems Engineering, National University of Defense Technology, Changsha 410073, China (email: pengguansheng_nudt@163.com; xinglining@gmail.com).

multiple VTWs, each located in one orbit, but at most one of its VTWs can be selected for observation. This is called the "multi-orbit uniqueness constraint". Fig. 2 shows an example to illustrate this constraint. In this example, the satellite moves during orbit 1, 2 and 3 successively. Different orbits are separated by a long period of shadow in which the observations cannot be processed. Due to the limitation of the rotation capability of the satellite's camera on the roll axes, target 1 can be observed during orbit 1 and 2, but it is invisible in orbit 3. Similarly, target 2 can be observed during orbit 2 and 3. For each target, at most one of its VTWs can be chosen. This constraint makes the scheduling during different orbits strongly coupled. Therefore, the multi-orbit scheduling problem is more difficult than the single orbit scheduling. In addition, for each pair of consecutive observations, a transition time is required to rotate the camera from the previous observation to the next observation. The length of the transition time is proportional to the angular changes on the three axes, and thus it depends on the start times of the two observations. This is called the "time-dependent" transition time constraint. Despite that a variety of previous studies have considered the time-dependent transition time [5], [6], the inherent properties of the time-dependency are still not clear, as well as its impacts on the scheduling.

Fig. 2. Multiple time windows constraints.

By reviewing literature, we found that the time-dependent transition time is close to the time-dependent travel time in the context of routing problem, such as the Time-Dependent Vehicle Routing Problem (TDVRP) and the Time-Dependent Orienteering Problem (TDOP) [7], [8]. This time-dependency originates from many practical scenarios where the travel time between a pair of vertices may vary at different time slots during a day due to traffic congestion. A number of works have been conducted to analyze the inherent properties of the time-dependency of the travel time: the First-In-First-Out (FIFO) rule and the triangle inequalities rule. Inspired with these works, we model the time-dependent transition time to verify whether it satisfies these two rules. On this basis, a novel heuristic algorithm coupled with a specific insert operator is proposed to address the AEOS scheduling with the

visible time window constraints. To deal with the multi-orbit uniqueness constraint, we embed an assignment procedure into the insert operator in order to allocate the targets with multiple observations to the appropriate orbit.

The main contributions of this paper are twofold: (1) We are the first to model the time-dependent transition time according to its properties: the FIFO rule and the triangle inequalities rule. We prove that in practice the time-dependent transition time always follows these rules; (2) To tackle the problem, we present an efficient heuristic combined with a fast feasibility check method and an *assignment* procedure to address the operational constraints. Extensive experiments are conducted to compare our algorithm with a commercial solver and the state-of-the-art heuristics for single-satellite instances and multi-satellite instances.

The remainder of this paper is organized as follows. Section II provides a review of the previous studies in the satellite scheduling field. Section III describes the mathematical modeling of the AEOS scheduling with time-dependent transition time. Solution methodology is presented in Section IV. Section V conducts the numerical experiments and demonstrates the performance of our algorithm. Finally, Section VI concludes our work and present the future research.

II. Literature Review

Due to the high practical value and complexity, the AEOS scheduling problem has been extensively studied the last decades. A limited number of exact algorithms have been proposed to tackle the simplified version of the problem in which the time-dependency of the transition time is ignored. Bianchessi et al. [9] present a column generation technique to solve the linear relaxation of the problem to obtain the upper bound, but the time-dependent transition time is not modeled. Wang et al. [10] build a mixed integer programming model for the AEOS scheduling problem in which three observation chances are chosen from each VTW. As a result, the transition times between the observation chances of different VTWs can be pre-computed and thus the time-dependency is ignored. Chen et al. [11] also build a MIP model for multiple satellite scheduling. However, they use an upper bound to replace the actual transition time empirically. Furthermore, these exact algorithms can only solve very small instances, which cannot satisfy practical needs.

A variety of heuristics and meta-heuristics has been proposed for the AEOS scheduling problem. Lemaître et al. [3] are the first to comprehensively introduce the overall AEOS problem. They present four heuristic algorithms to solve a simplified version of the problem. Cordeau and Laporte [12] develop a tabu search algorithm for the single orbit scheduling. They relax the time window constraints in their method, allowing infeasible solutions during the search. Habet et al. [13] solve a multi-objective model of the AEOS scheduling where the primary objective is to maximize the rewards of observations, and the secondary

objective is to minimize the total transition time. Unfortunately, the aforementioned works only consider scheduling during a single orbit. In practice, satellites have multiple observation chances during a long period so that the multi-orbit scheduling is more realistic.

For multi-orbit scheduling, Wolfe and Sorensen [14] propose a priority dispatch approach, a look ahead approach and a genetic approach. Dilkina et al. [4] propose a hybrid method of permutation-based search and constraint propagation. Lin et al. [15] present a Lagrangian relaxation method and several linear search techniques for the daily management of their satellite system. All of these works, however, use a constant value to represent the actual transition time and ignore its time-dependency.

Recently, Liu et al. [5] address the AEOS scheduling with time-dependent transition times. An adaptive large neighborhood search (ALNS) algorithm is developed to tackle the single satellite scheduling. He et al. [6] extended this approach to the cases of multiple satellites with an adaptive task assignment based ALNS (A-ALNS) heuristic. In both of their works, a group of local moves are specifically designed to enhance the intensification and diversification of the search, and an insertion operator based on the “time slack” is presented, considering the time-dependent transition time. The authors claimed that their ALNS can work well on large-size instances. However, we prove that the “time slack” can only be used when the transition time follows the FIFO rule which will be explained in detail in Section III-B1. In a recent work [1], a “minimal transition time” is introduced to replace the actual transition time based on the condition of the FIFO rule. On this basis, they present an Iterative Local Search algorithm to address the scheduling problem with the time-dependent profits, which is different with our scheduling problem with constant profits. This work, however, does not provide the proof of the FIFO rule for the time-dependent transition time.

In this work, we verify that the time-dependent transition time always satisfies the FIFO rule, given any realistic settings of satellite parameters. Furthermore, we prove that the time-dependent transition time still satisfies the triangle inequalities which mean that observing target B directly from target A always finishes earlier than observing B from target A through target C, under the condition of the special piecewise function of calculating the transition times. These results motivate us to design an effective heuristic for the scheduling problem. We develop a hybrid heuristic based on the “minimal transition time” [1] to address the time-dependent transition time, and embed an *assignment* procedure to tackle the multi-orbit uniqueness constraint. The presented algorithm is compared with the state-of-the-art algorithm, the ALNS algorithm [5], on single satellite instances. Due to the high similarity between the multi-orbit scheduling and the multi-satellite scheduling, the presented heuristic is also tested on the multi-satellite instances and compared with the A-ALNS algorithm as well. We use the same semi-agile satellite AS-01 as in [5] and [6] whose look angles

are fixed during the observation.

In the field of satellite scheduling, the scheduling of the Tracking and Data Relay Satellites (TDRS) with Intersatellite Link Antennas (ILA) is a very similar problem [16], [17]. The rotation of the ILAs also requires a transition time which depends on the service start times of two consecutive tasks. However, the calculations of the transition time and the change of the attitude angles are different, which may lead to a large difference of the model as well as the solution approach.

III. Problem description

In this section, we present the multiple AEOSs scheduling problem and its mathematical programming model. A group of candidate targets T associated with their geographic positions, service duration d_i and profits p_i are provided by users. Due to the limitation of on-board resources and the visibility constraints, only a subset of targets can be scheduled and observed by a group of satellites during a specific horizon (normally a day). Given the satellite parameters (the orbital parameters and the rotation capacity parameters of the camera) and the geographic positions of targets, a visibility analysis can be carried out to generate the input data of the scheduling, i.e., the VTWs of each visible target associated with their visible orbits. During each VTW, the start time and end time, and the look angles per second on the three axes (roll, pitch and yaw) are given. The objective of this scheduling problem is to determine the observations of a subset of targets during their VTWs, while satisfying the multi-orbit uniqueness constraints and transition time constraints. Notations of parameters in this study are summarized in Table I. A virtual source target s and a sink target e are added so that the schedule of each orbit must start from s and end at e . We define a binary parameter b_i^k to indicate the visibility of target i during orbit k . In order to facilitate the description of the model and the methodology, we ignore the superscript k when describing the scheduling during orbit k if this does not provide any ambiguity. For example, the observation i^k of target i is denoted as i if orbit k is not specified in the description.

A. Assumption

This study focuses on solving the scheduling problem with time-dependent transition time. Thus, some non-significant assumptions can be made in order to simplify the whole problem:

- 1) The satellite has sufficient on-board power and memory for each orbit, and thus the download scheduling of image data are not considered;
- 2) This study only considers spot targets which can be observed with one pass. The polygon targets can be divided into multiple strips which can be regarded as multiple independent spot targets.
- 3) We study the semi-agile satellite AS-01 whose look angles are unchanged during the observation. Hence, for

TABLE I
Notations

T	set of possible targets
i, j	target index, $i, j \in T \cup \{s, e\}$, in which s, e are virtual source target and sink target
i^k	observation of target i during orbit k
O	set of orbits during the scheduling horizon
VTW_i^k	the visible time window of target i during orbit k
$[st_i^k, et_i^k]$	the visible time window for target i during orbit $k \in O$
b_i^k	the binary parameter which equals 1 only if target i is visible for orbit k , otherwise $b_i^k = 0$;
p_i	profit of target i
d_i	duration of observing target i
γ, π, ψ	roll, pitch and yaw angles
Δg	total change of the look angles for a transition of the camera
$trans(\Delta g)$	transition time with respect to the total change of the look angles
$trans_{ij}^k(t_i^k, t_j^k)$	transition time between target i and j with respect to their start times during orbit k
$mintrans_{ij}^k(t_i)$	minimal transition time between target i and j while observing target i at moment t_i during orbit k

each observation, we use the look angles of its start time to calculate the transition time towards the next observation.

B. Modeling of Transition Time

As mentioned above, the time interval between two successive observations should be longer than the transition time which enables the satellite to change its look angles. The transition time intuitively depends on the look angles, and thus depends on the start times of the two observations. In a recent works [5], the authors considers the time-dependent transition time, which is calculated by the following piecewise linear function:

$$trans(\Delta g) = \begin{cases} 11.66, & \Delta g \leq 10 \\ a_1 + \Delta g/v_1, & 10 < \Delta g \leq 30 \\ a_2 + \Delta g/v_2, & 30 < \Delta g \leq 60 \\ a_3 + \Delta g/v_3, & 60 < \Delta g \leq 90 \\ a_4 + \Delta g/v_4, & \Delta g > 90 \end{cases}, \quad (1)$$

where v_1, v_2, v_3 and v_4 are four different angular velocities with respect to different values of the change of look angles, corresponding to the rotation capacity parameters of the satellite's camera. a_1, a_2, a_3 and a_4 are the constant terms corresponding to the angular velocities. For the satellite AS-01 we used, these values are $v_1 = 1.5^\circ/s, a_1 = 5, v_2 = 2^\circ/s, a_2 = 10, v_3 = 2.5^\circ/s, a_3 = 16, v_4 = 3^\circ/s, a_4 = 22$. Note that the setting of these parameters must satisfy that the transition time function is continuous and monotonically non-decreasing. Δg represents the total change of the look angles of the transition movement. It is calculated by $\Delta g = |\Delta \gamma| + |\Delta \pi| + |\Delta \psi|$, i.e., the sum of the change of look angles on three axes (roll, pitch and yaw). Thus, given two consecutive observation i and j and their start times t_i, t_j , the corresponding transition time can also be expressed as $trans_{ij}(t_i, t_j)$. In their methods, lots of computational time have been

spent on calculating the actual transition times between each pair of consecutive observations to guarantee the feasibility of the solutions, while no attention has been paid to the modeling of the transition time. In this work, we discuss the time-dependent transition time from two perspectives: the FIFO rule and the triangle inequalities. On this basis, we present a concept of “minimal transition time” to replace the actual transition time. An efficient pre-processing method is presented in Section IV-B to pre-compute the minimal transition times.

1) FIFO rule: When the time-dependency of transition times is considered, it is crucial to discuss whether the transition time for each pair of observations satisfies the FIFO rule. In the context of routing problem, the FIFO property of travel times guarantees that if a vehicle leaves vertex i for vertex j at a given time, leaving vertex i at a later time leads to a later arrival at vertex j . In our problem, the FIFO rule of transition times should be discussed for two cases: 1) fixing the start time of the next observation, varying the start time of the previous observation (see Fig. 3(a)); 2) fixing the start time of the previous observation, varying the start time of the next observation (see Fig. 3(b)). We first discuss the former case.

Proposition 3.1: Given the start time t_j of the next observation j , $trans_{ij}(t_i, t_j)$ satisfies the FIFO rule if and only if $\frac{dtrans_{ij}(t_i, t_j)}{dt_i} \geq -1$.

Proof. ($\rightarrow$ necessity)

If $trans_{ij}(t_i, t_j)$ satisfies the FIFO rule, then

$$t_i + trans_{ij}(t_i, t_j) \leq t_i + \Delta t + trans_{ij}(t_i + \Delta t, t_j), \Delta t \geq 0, \quad (2)$$

$$\frac{trans_{ij}(t_i + \Delta t, t_j) - trans_{ij}(t_i, t_j)}{\Delta t} \geq -1, \quad (3)$$

$$\lim_{\Delta t \rightarrow 0} \frac{trans_{ij}(t_i + \Delta t, t_j) - trans_{ij}(t_i, t_j)}{\Delta t} \geq \lim_{\Delta t \rightarrow 0} -1, \quad (4)$$

$$\frac{dtrans_{ij}(t_i, t_j)}{dt_i} \geq -1. \quad (5)$$

($\leftarrow$ sufficiency)

The look angles continuously vary over time in the real system, despite that the look angles in the input data are recorded in a discrete form. It is given that the calculations of the transition time (equation (1)) and the total change of the look angles are continuous functions. From the mean value theorem, we know that at least one point t_i^* in $[t_i, t_i + \Delta t]$ such that

$$\frac{dtrans_{ij}(t_i, t_j)}{dt_i} = \frac{trans_{ij}(t_i + \Delta t, t_j) - trans_{ij}(t_i, t_j)}{\Delta t}, \quad (6)$$

If $\frac{dtrans_{ij}(t_i^*, t_j)}{dt_i^*} \geq -1$ for all t_i^* in $[t_i, t_i + \Delta t]$, then

$$\frac{trans_{ij}(t_i + \Delta t, t_j) - trans_{ij}(t_i, t_j)}{\Delta t} \geq -1, \quad (7)$$

$$t_i + trans_{ij}(t_i, t_j) \leq t_i + \Delta t + trans_{ij}(t_i + \Delta t, t_j), \quad (8)$$

which indicates that the FIFO rule is satisfied $\square$

According to the aforementioned proposition, we need to verify $\frac{dtrans_{ij}(t_i, t_j)}{dt_i} \geq -1$. Suppose $\kappa = \frac{dtrans_{ij}(t_i, t_j)}{dt_i}$, which represents that given t_j , the variation of the transition time per unit time at t_i . For the semi-agile satellite AS-01 we considered, the roll angle is fixed during the observation, and the yaw angle is close to zero. We obtain:

$$|\kappa| = \lim_{\Delta t \rightarrow 0} \frac{|g_i(t_i + \Delta t) - g_i(t_i)|}{v} \approx \lim_{\Delta t \rightarrow 0} \frac{|\pi_i(t_i + \Delta t) - \pi_i(t_i)|}{v} \quad (9)$$

where $g_i(t_i)$ and $\pi_i(t_i)$ represent the instantaneous look angles on three axes and the instantaneous pitch angle at t_i , respectively. v is the instantaneous angular velocity of the camera. As the input of the scheduling, the look angles during each VTW are given in a discrete form (per second). Equation (9) can be transferred to:

$$|\kappa| \approx \frac{|\pi_i(t_i + 1) - \pi_i(t_i)|}{v} \leq \frac{|\pi_i(t_i + 1) - \pi_i(t_i)|}{v_{min}} = |\kappa_{upper}|, \quad (10)$$

where v_{min} represents the minimum angular velocity of camera among $\{v_1, v_2, v_3, v_4\}$ and $|\kappa_{upper}|$ is the upper bound of $|\kappa|$. In practice, the value of κ_{upper} depends on the settings of satellite parameters and the geographic positions of targets. However, we found that given any realistic settings of satellite parameters, the value of $|\kappa_{upper}|$ is always much less than one, based on the calculation from the instance data. It implies that the time-dependent transition time follows the FIFO rule for the former case. In other words, postponing the start time of the previous observation cannot lead to an earlier arrival at the next observation. Likewise, for the latter case, given the start time of the previous observation, if t_j satisfies the transition time constraint, $(t_j + \Delta t)$ also satisfies this constraint. A similar conclusion is also illustrated in a previous work [18], but the detailed proof is not given. Based on the FIFO rule, a concept of “minimal transition time” is presented to simplify the computation of the actual transition times, which will be described in the subsequent subsection. An efficient insertion procedure with fast feasibility check can be utilized in the proposed heuristic algorithm.

Fig. 3. Examples of transition times for the FIFO property.

2) Triangle inequalities rule: Another crucial influence of time-dependent transition time is the triangle inequalities, which means that the transition time between i and j is shorter than transition time from i to j through a “detour” observation m , when fixing the start times of observation i and j . Generally, if the angular velocity is constant, the triangle inequalities are always satisfied since

$\Delta g_{ij} \leq (\Delta g_{im} + \Delta g_{mj})$, where Δg_{ij} are the change of look angles from i to j . However, in practice, the angular velocity varies depending on the absolute value of the change of look angles, as can be seen in equation (1). The utilization of different angular velocities relative to different changes of look angles stems from the design of the maneuvering system that a larger change of look angles can be transited by a faster maneuvering engine in order to reduce the transition time. Hence, the larger Δg is, the higher the corresponding angular velocity is. Equation (1) is a continuous and concave function, and $v_1 \leq v_2 \leq v_3 \leq v_4$ is always satisfied.

Ignoring the service time of the detour observation m , the triangle inequality is satisfied only if the following inequality is satisfied:

$$trans(\Delta g_{im}) + trans(\Delta g_{mj}) \geq trans(\Delta g_{ij}) \quad (11)$$

where Δg_{ij} is the difference of the look angles of observation i and j , and v_{ij} represents the angular velocity corresponding to $|\Delta g_{ij}|$.

Given that $\Delta g_{ij} = \Delta g_{im} + \Delta g_{mj}$, $|\Delta g_{ij}| \leq |\Delta g_{im}| + |\Delta g_{mj}|$. There are two cases:

(1) If $|\Delta g_{ij}| > |\Delta g_{mj}|$, then $v_{im}, v_{mj} \leq v_{ij}$. Substituting it to equation (1), inequality (11) is satisfied due to the property of concave function (1).

(2) If $|\Delta g_{ij}| \leq |\Delta g_{mj}|$, given that equation (1) is a monotone non-decreasing function, $trans(\Delta g_{mj}) \geq trans(\Delta g_{ij})$.

We have proven that the time-dependent transition time respects the triangle inequalities rule for the given transition time function (equation (1)). This property indicates that inserting an additional observation between two scheduled observations will not shorten the total transition time. In other words, the increment of the total transition time for an insertion can be regarded as a cost, and removing any observations from a sequence of scheduled observations can shorten the total transition time.

3) Minimal transition time: Based on the analysis of the time-dependent transition time, the “minimal transition time” [1] can be used to replace the actual transition time. The minimal transition time allows the next observation to happen as early as possible when the start time of the previous observation is given. Fig. 4 shows an example of two consecutive observations i and j where the start time of observation i is fixed and the different colored rectangles of j represent its different start times. Since the time-dependent transition time follows the FIFO rule, given the start time of observation i , the earliest start time es_j of the observation j can always be determined. Observing j after es_j always satisfies the transition time constraint, while the observation of j before es_j is impossible. The minimal transition time $mintrans_{ij}(t_i)$ is the minimal required transition time corresponding to es_j . Similarly, given the start time of observation j , the latest start time ls_i of observation i can also be determined. To accelerate the algorithm, the minimal transition times between each pair of VTWs at each moment can be pre-calculated.

Fig. 4. The change of the transition time between observation i and observation j with different observation start times of observation j .

C. Integer Programming Model

In the model of the AEOS scheduling, three types of decision variables are defined as follows:

x_{ij}^k : binary variable equal to 1 if target j is scheduled immediately after target i , and 0 otherwise.

y_i^k : binary variable equal to 1 if target i is selected during orbit k , and 0 otherwise.

t_i^k : integer variable representing the start time of observing target i during orbit k .

During a VTW, the look angles per second are given in a discrete form. Thus, the decision variable t_i^k is also defined to specify which moment (second) should be chosen. Based on the statements mentioned above, the mathematical formulation can be expressed as follows:

$$\text{Maximize } \sum_{i \in T} \sum_{k \in O} p_i \cdot y_i^k \quad (12)$$

$$\sum_{k \in O} y_i^k \leq 1, \forall i \in T \quad (13)$$

$$\sum_{\substack{j \in T \cup \{e\} \\ j \neq i}} x_{ij}^k = \sum_{\substack{j \in T \cup \{s\} \\ j \neq i}} x_{ji}^k = y_i^k, \forall i \in T, k \in O \quad (14)$$

$$\sum_{j \in T \cup \{e\}} x_{sj}^k = 1, \forall k \in O \quad (15)$$

$$\sum_{j \in T \cup \{s\}} x_{je}^k = 1, \forall k \in O \quad (16)$$

$$t_i^k + d_i + \mintrans_{ij}^k(t_i^k) - t_j^k \leq M(1 - x_{ij}^k), \quad \forall i, j \in T, k \in O \quad (17)$$

$$st_i^k y_i^k \leq t_i^k \leq et_i^k y_i^k, \forall i \in T, k \in O \quad (18)$$

$$y_i^k \leq b_i^k, \forall i \in T, k \in O \quad (19)$$

$$x_{ij}^k \in \{0, 1\}, y_i^k \in \{0, 1\}, \forall i, j \in T \cup \{s, e\}, k \in O. \quad (20)$$

The objective function (12) maximizes the total profit of the scheduled targets. Constraints (13) are the multi-orbit uniqueness constraints. Constraints (14) are the flow balance constraints and connect variable x_{ij}^k and y_i^k . Constraints (15) and (16) express that the solution

starts from the virtual source target s and ends at the virtual sink target e . Constraints (17) and (18) indicate the transition time constraints and the visible time window constraints. Constraints (19) stipulate that each target can only be observed during its visible orbit. Constraints (20) state the domains of the decision variables.

IV. Solution Approach

A. General outline

The AEOS scheduling problem is proved to be NP-hard [3]. Still, a heuristic algorithm is promising to find the near-optimal solutions in an acceptable time. To solve the problem efficiently, a Greedy Randomized Iterated Local Search (GRILS) heuristic is presented to tackle the problem. The GRILS is a hybridization of the Greedy Randomized Adaptive Search Procedure (GRASP) and the Iterated Local Search (ILS) [19]. The ILS is a classic meta-heuristic that embeds local search and perturbation heuristics within an iterative process. It has been implemented successfully to deal with some conventional combinatorial optimization problem [20], [21]. The GRASP performs a number of iterations, each consisting of a constructive method followed by local search procedures. The hybrid version of these two approaches can efficiently solve the Multi-Constraint Team Orienteering Problem with Multiple Time Windows (MCTOPMTW) [19]. We employ this hybrid heuristic for our problem but a few adaptations are made. Specifically, our algorithm does not construct a new solution at each iteration from scratch. Instead, the best found solution of the previous iteration is used for the next iteration, and at each iteration, local search moves are successively performed on the solution. Algorithm 1 shows the framework of our GRILS algorithm.

The algorithm starts from a pre-processing procedure which pre-computes the earliest start times and latest start times for each pair of VTWs and for each moment. This procedure significantly reduces the computational time during the search. Then, the algorithm goes into two layers of loops. The inner loop corresponds to an ILS heuristic while the outer loop controls the randomness of the local moves in the inner loop. In the inner loop, two local search operators are defined: the INSERT operator and the SHAKE operator. The INSERT operator inserts unscheduled targets into the current solution until there are no more feasible insertions. This operator is governed by a greediness parameter (*Greed*) which controls the randomness of selecting insertions from their candidate list. The larger *Greed* is, the less randomness the INSERT operator has. For example, if there are 10 feasible insertions for the current solution S_c and the insertions are ordered based on non-decreasing profits, only the top $\lceil (1 - \text{Greed}) \cdot 10 \rceil$ insertions are considered. The SHAKE operator removes a certain number of scheduled observations from the current solution in order to escape from the local optimum. After the execution of the INSERT operator, if the newly generated solution is better than the best found solution, then the new solution will be recorded as the best found

solution. The inner loop terminates if no improvement can be found within a certain number of consecutive iterations. In the outer loop, parameter *Greed* varies from *StartGreed* to *StartGreed* – *GreedyRange* with a step of *GreedDecrease*. At each iteration of the outer loop, the ILS heuristic is performed on the incumbent best found solution.

Algorithm 1 Framework of Greedy Randomized Iterated Local Search

```

 $S_b \leftarrow \emptyset$ ; //Best found solution  $S_b$ 
 $S_c \leftarrow \emptyset$ ; //Current solution  $S_c$ 
 $Iteration \leftarrow 0$ ;
for  $Greed = StartGreed$ ;  $Greed > StartGreed - GreedyRange$ ;  $Greed = GreedDecrease$  do
   $S_c \leftarrow S_b$ ;
  while  $Iteration < NumofIterNoImp$  do
     $S_c \leftarrow INSERT(S_c, Greed)$ ;
    if  $S_c$  is better than  $S_b$  then
       $S_b \leftarrow S_c$ ;
       $Iteration \leftarrow 0$ ;
    else
       $Iteration \leftarrow Iteration + 1$ ;
    end if
     $S_c \leftarrow SHAKE(S_c)$ ;
  end while
end for
return  $S_b$ ;

```

B. Pre-processing

For a pair of VTWs, given the start time of the previous observation, the earliest start time of the next observation and the corresponding minimal transition time can be pre-calculated by using a dichotomy algorithm [1]. Notice that this algorithm can only be used under the condition that the FIFO rule is satisfied. More specifically, given the start time t_i of the previous observation i , the procedure checks whether the start time st_j of j satisfies the transition time constraint, i.e., using the look angles data corresponding to the associated moments and calculating the transition time with equation (1). If the constraint is satisfied, st_j is the earliest start time. If the end time et_j dissatisfies the constraint, then no feasible start time during VTW_j exists after observing target i at moment t_i . Otherwise, the algorithm checks the feasibility of the midpoint of the time interval $[st_j, et_j]$ and divides the time interval until a feasible earliest start time is found.

Due to a potential large number of calling the dichotomy algorithm during the iterative search, we pre-calculate the earliest start times (the latest start times) for each pair of VTWs during the same orbit and for each start time of the previous (next) VTW. By doing so, the minimal transition times can be directly obtained without any calculation during the optimization and thus the computational time will be reduced. Furthermore, due to the FIFO rule, the earliest start time of VTW_j corresponding to $(t_i + 1)$

can be easily found by checking the moments following *EarliestStartTime* $_{ij}(t_i)$. Thus, the dichotomy algorithm is called only once for each pair of VTWs during the same orbit. From the backward direction, the latest start time of the previous observation can be calculated in a similar way, denoted as *LatestStartTime* $_{ij}(t_j)$.

Another improvement of reducing the computation time is that for each VTW_i^k during orbit k , we calculate an “unreachable” earliest time vector ue_i^k . $ue_i^k(j)$ indicates the earliest start time when leaving VTW_i^k cannot reach VTW_j^k . Similarly, the “unreachable” latest time vector ul_i^k can also be calculated. Intuitively, Observation j^k can be inserted after (before) observation i^k only if $t_i < ue_i^k(j)$ ($t_i > ul_i^k(j)$). In this way, the number of unnecessary insertion attempts is significantly reduced.

C. INSERT operator

An INSERT operator is defined to intensify the search of the solution space. It selects a subset of unscheduled targets and schedules them in the current solutions while maintaining the feasibility of the solution. A solution is comprised of several sequence of scheduled observations with each sequence representing the schedule of an orbit. Given a sequence, each pair of consecutive observations must satisfy the visible time window constraints. When inserting an observation into a certain position in the sequence, the scheduled observations succeeding the position may have to be rescheduled or postponed to ensure the feasibility of the solution. And thus their transition times need to be updated to check whether the visible time window constraints are violated. This full feasibility check would require lots of computational time because a large number of insertion attempts are infeasible when considering the time window constraints. In our heuristic, we present an INSERT operator with a fast feasibility check method, inspired by the work of Vansteenwegen et al. [22] for the Orienteering Problem with Time Windows (OPTW). The time-dependent transition time is considered in the fast feasibility check method. Notice that this insertion method with fast feasibility check can work only when the FIFO rule is satisfied. To address the multi-orbit uniqueness constraint, we design an *Assignment()* procedure and integrate it to the INSERT operator.

Specifically, constraints (13) are ignored during the execution of the INSERT operator, that is, the targets which have been scheduled during one orbit can still be possibly scheduled during another orbit. After insertions during each orbit, the algorithm checks if any targets have been scheduled more than once. For the targets with duplicated observations, only one observation can be retained and the other will be removed from the solution. This procedure is denoted by *Assignment()*. According to the preliminary experiments, a Flexible Assignment (FA) strategy is presented and can considerably improve the solution quality. The FA strategy allocates the target to the orbit with the least number of VTWs. The intuitive idea behind this strategy is that targets should be preferentially scheduled to the orbit with least number of VTWs,

so that the orbits with large number of VTWs are able to accommodate the observations of other targets. After the assignment procedure, only one VTW (observation) is retained for each scheduled target, and the other VTWs associated with the same target are removed from the solution and marked as “frozen”. The “frozen” VTWs will not be considered for next insertion unless they are “unfrozen” once the corresponding targets are removed from the solution by the SHAKE operator.

Fig. 5 shows a visual overview of the INSERT operator with fast feasibility check. Given a scheduled observation sequence $(j-1), j, (j+1), (j+2)$ during a certain orbit (superscript k is ignored), we record for observation j the earliest start time es_j which is calculated by $es_j = EarliestStartTime_{(j-1)j}(es_{(j-1)})$. The earliest start time of the first observation $(j-1)$ in the sequence is set to its window start time $st_{(j-1)}$. The latest start time of each observation can also be calculated in the similar way, as can be seen in Fig. 5. Clearly, the valid time interval $[es_i, ls_i]$ for observation i defines the range of its feasible start times. If the start time of any observation is determined, the start times of other observations can easily be determined by using a backtracking algorithm. In Fig. 5, the algorithm tries to insert observation i between observation j and $(j+1)$. Its earliest start time and latest start time are calculated from its preceding observation and its succeeding observation. This insertion is feasible only if its earliest start time is earlier than its latest start time. This insert method does not require the recalculation of transition times for other observations in the sequence, and thus significantly reduce the computational time. Notice that after an insertion is executed, the earliest start times (the latest start times) of observations succeeding (preceding) the insert position needed to be updated.

Fig. 5. The insertion of a new observation i with fast feasibility check.

The pseudo code of the INSERT operator is presented in Algorithm 2. Firstly, for each orbit k in S_c , the algorithm finds all the unscheduled VTWs during that orbit and their feasible insert positions with the fast feasibility check. For each unscheduled target i , its feasible insertions which corresponds to different insert positions during orbit k

are stored in L_i^k . Afterwards, for each L_i^k , we only retain the insertion with the lowest cost and insert it to the candidate list L^k for orbit k . The cost of an insertion is the increment of the actual transition times calculated by the following equation:

$$cost_i = trans_{ji}(es_j, es_i) + trans_{i(j+1)}(es_i, es'_{j+1}) - trans_{j(j+1)}(es_j, es_{j+1}), \quad (21)$$

where es'_{j+1} is the updated earliest start time of $(j+1)$. Secondly, all the insertions in L^k are ordered by non-decreasing profit, and only the first $\lfloor (1 - Greed) \cdot |L^k| \rfloor + 1$ insertions are preserved. By doing so, the degree of randomness can be controlled by the parameter *Greed*. The smaller *Greed* is, the more likely the targets with lower profit can be inserted. Then, we select an insertion in L^k based on Roulette Wheel, and process the insertion on the current solution S_c , and then updating the earliest start times and the latest start times. Lastly, after executing the insertions during each orbit, in the current solution, there may exist some targets scheduled more than once during different orbits, which violates the multi-orbit uniqueness constraint. To ensure the feasibility, an *Assignment()* procedure is defined to assign the targets with duplicated observations to only one orbit. The INSERT operator terminates if no more feasible insertions exist.

D. SHAKE operator

The SHAKE operator is applied to S_c in order to escape from local optimum and better explore the solution space. It is derived from the work of Vansteenwegen et al. [22] for OPTW, but some modifications are made. With this operator, one or more scheduled observations will be removed for each orbit. Two integers are defined: S_d indicates the place in the sequence to start the removing process and R_d indicates how many consecutive observations are removed at a time. Since different sequences corresponding to different orbits may have different numbers of observations, the values of these two integers for different orbits are different. To diversify the search, the sequence with a large number of observations usually requires a large value of R_d .

Both S_d and R_d are initially set to one. After each SHAKE operator, S_d is increased by R_d and R_d is increased by one for the next SHAKE operator. If S_d^k is greater than the size of VTWs for orbit k , this number is subtracted to determine the new position. If R_d^k equals to this size, R_d^k is reset to one. This SHAKE operator has proven to be an excellent technique to address some well-known problems discussed by Gendreau et al. [23]. This is because almost every observation is removed at least once during the entire procedure.

V. Experimental results

Three comparative experiments are conducted to evaluate the performance of our algorithm. First, we compare our results with the optimal solutions solved by the CPLEX solver; Second, we compare our algorithm with

Algorithm 2 *INSERT*($S_c, Greed$)

```

 $IsNoMoreInsert \leftarrow false;$ 
while  $IsNoMoreInsert$  do
   $IsNoMoreInsert \leftarrow true;$ 
  for each orbit  $k \in O$  do
    if Unscheduled and unfrozen observations exist
      then
         $IsNoMoreInsert \leftarrow false;$ 
      end if
    for each unscheduled and unfrozen  $VTW_i^k$  during
      orbit  $k$  do
      Assume the scheduled observation sequence
         $\{1^k, \dots, n^k\}$  in the current solution  $S_c$  during
        orbit  $k$ .
      for each insert position  $ip$  in  $\{1^k, \dots, n^k\}$  do
        Assume position  $ip$  is between observation  $j^k$ 
          and  $(j+1)^k$ ;
        if  $ue_j^k(i) > es_j^k$  and  $ul_{(j+1)}^k(i) > ls_{(j+1)}^k$  then
           $es_i^k \leftarrow EarliestStartTime_{ji}(es_j^k);$ 
           $ls_i^k \leftarrow LatestStartTime_{(j+1)i}(ls_{j+1}^k);$ 
          if  $es_i^k \leq ls_i^k$  then
            insert the insertion into the candidate
              list  $L_i^k$ ;
          end if
        end if
      end for
      Select the insertion with the lowest cost in  $L_i^k$ 
        and insert it into the candidate list  $L^k$  of orbit
         $k$ ;
    end for
    Order the insertions in  $L^k$  by non-increasing profit
      and remove the last  $\lceil Greed|L^k| \rceil - 1$  insertions;
    Choose an insertion in  $L^k$  with roulette wheel
      selection based on their profits and process the
      insertion on  $S_c$ ;
    Update the earliest start times (latest start times)
      of the scheduled observations preceding (succeed-
        ing) the insert position;
  end for
  Assignment( $S_c$ );
end while

```

the state-of-the-art algorithms, that is ALNS algorithm [5] for single satellite instances, and A-ALNS algorithm [6] for multi-satellite instances; Third, a sensitivity analysis is made in order to evaluate the choice of the algorithm parameters.

Preliminary experiments on a random subset of instances demonstrate good results for $StartGreed = 1$, $GreedRange = 0.2$, $GreedDecrease = 0.02$. The maximum number of iterations with no improvement is set to 300. These parameter settings are used for all computational experiments. The sensitivity analysis of these parameter settings is carried out in Section V-D. The algorithm is implemented in C Sharp 1.6. All experiments are tested on an Intel Core 2 processor with 2.5 GHz

and 8 GB of RAM memory. The consumed time of the pre-processing procedure is included in the presented CPU times. The single satellite instances are available at <https://www.mech.kuleuven.be/en/cib/op#section-32>. The multi-satellite instances are presented in the work of He et al. [6].

A. Test instances

Since we consider the same agile satellite AS-01 as in Liu et al. [5], we compare the performance of our algorithm with their ALNS algorithm on their test instances for single satellite scheduling. The targets for observation are randomly generated with a uniform distribution in Chinese region (3°N-53°N and 74°E-133°E). Liu et al. [5] also present worldwide instances but our preliminary results show that in both the ALNS algorithm and our algorithm, almost all the targets can be scheduled in the worldwide instances, which cannot reflect the difference. Thus, the comparison on worldwide instances is not shown. The profit and the observation duration of each target are generated in a uniform distribution with the range of [1,10] and [15,30] (in seconds). In addition to the single satellite instances used in Liu et al. [5], the multi-satellite instances [6] are tested by our algorithm since the multiple satellite scheduling can be classified into multi-orbit scheduling. For detailed parameters of the satellites in the instances, we refer to the work by Liu et al. [5] and He et al. [6]. The single satellite instances are available at <https://www.mech.kuleuven.be/en/cib/op#section-32>.

B. Comparison with optimal solutions on small instances

The first experiment is a straightforward comparison between the results of the GRILS and the optimal solution solved by the commercial solver CPLEX (version 12.8.0) on small instances. The model in Section III-C cannot be directly solved by CPLEX due to the non-linear transition time constraints. However, as the input of the problem, the look angles data in each VTW are given in seconds (discrete form). Thus, the accessibility among different VTWs during the same orbit with different start times can be pre-computed and the transition time constraints are linearized. And yet, since each VTW has around 300 seconds, only small instances can be solved by CPLEX. In order to reduce the computational time and the memory usage of solving the CPLEX model, we define a time step T_{step} to evenly discretize the VTWs and specify the resolution. T_{step} is set to 100 seconds in this experiment.

Table II shows the comparative results. The first column represents the name of instances, denoted by “ID_ $|T|$ ” where $|T|$ is the number of targets. N_t and P_t respectively represent the number of visible VTWs of that instance and the total profits available. For CPLEX, the optimal values of the instances and the computational time are recorded in column 4 and 5. The gap between the best found solution and the optimal solution, and the average computation time t_{cost} (in seconds) of our GRILS algorithm are recorded in column 6 and 7, respectively. In these small

instances, each target has a visible time window in two orbits. From this comparison, the rather small value of gap illustrates that our GRILS algorithm obtains high-quality results and has very short computational times on these small instances.

TABLE II

Comparison between the CPLEX solver and the GRILS algorithm for small instances.

Instance	N_t	P_t	CPLEX		GRILS	
			$P_{optimal}$	Cpu(s)	Gap(%)	Cpu(s)
1_15	30	73	53	224.91	0.00	0.10
2_9	18	50	47	1.41	0.00	0.09
3_15	30	69	51	27.06	0.00	0.11
4_14	28	69	58	44.16	2.07	0.08

C. Comparison with state-of-the-art algorithms on large instances

In this experiment, both the single satellite instances and the multi-satellite instances are tested to evaluate the performance of our algorithm. For the single satellite instances, the ALNS algorithm [5] is compared with our algorithm, while its improved version for multi-satellite instances is the A-ALNS algorithm [6]. The A-ALNS is developed based on the ALNS algorithm, and an adaptive task assignment method is presented to allocate targets to their visible satellites and then schedule targets on their specified satellites by the ALNS algorithm. Tables III and IV show the comparative results, based on five independent runs. We keep the parameter setting of ALNS in [5] and compare the best parameter configuration of our ILS algorithm. Column P_s shows the scheduled profit of the algorithms. Column P_s/P_t is the ratio of the scheduled profits compared to the total target profit. Column “Targets” represents the number of targets in the instance. The “Outper” column shows how many percent our ILS outperforms ALNS in terms of scheduled profits.

TABLE III

Comparison on single satellite instances with Chinese distribution.

Targets	P_t	ALNS		GRILS		Outper(%)
		$P_s/P_t(\%)$	Cpu(s)	$P_s/P_t(\%)$	Cpu(s)	
100	571	91.13	22.06	99.75	7.99	9.45
200	1053	60.32	89.69	88.37	21.37	46.51
300	1474	43.45	175.65	72.41	38.13	66.63
400	2065	31.68	303.85	60.93	62.75	92.36
500	2683	26.42	420.61	53.45	94.02	102.28
600	3231	25.17	504.08	48.16	126.48	91.32
Aver		46.36		70.51		52.09

Table III shows the results for the Chinese area instances. The scheduled profits of our GRILS are on average 52% higher than those of the ALNS algorithm. Furthermore, the “Outperform” percentage grows largely as the size of the instance increases, which indicates that our algorithm performs better than the ALNS algorithm on large instances. For the instances with more than 400 targets, the collected profits of our algorithm are

almost twice those of ALNS algorithm. In terms of the computational time, our algorithm is many times faster than the ALNS algorithm. The 600-target instance only requires 43.4 seconds while this size of instance has reached the maximum capacity of a single AEOS satellite in practice for a daily scheduling.

TABLE IV

Comparison on multi-satellite instances with Chinese distribution.

Sate	Targets	A-ALNS		GRILS		Outper(%)
		$P_s/P_t(\%)$	Cpu(s)	$P_s/P_t(\%)$	Cpu(s)	
2	100-500	68.29	406.12	78.63	8.49	15.14
2	600-1000	30.09	1133.03	51.60	34.15	71.50
3	100-500	76.98	172.47	93.92	18.41	22.01
3	600-1000	37.49	763.92	73.18	58.56	95.19
4	100-500	88.57	172.08	99.06	27.10	11.85
4	600-1000	49.72	640.25	84.14	74.05	69.23
5	100-500	92.34	128.25	99.11	32.85	7.33
5	600-1000	55.54	589.94	86.59	86.89	55.91
6	100-500	95.87	87.79	99.80	39.64	4.10
6	600-1000	62.79	433.86	90.58	101.99	44.26
Aver		65.77		85.66		30.25

Table IV reports the comparison between the A-ALNS algorithm and our GRILS algorithm on multi-satellite instances with Chinese distribution. Instances are divided into two group: 100-500 and 600-1000 which indicate middle-size instances and large-size instances. The results are averaged in each group. It appears that our GRILS algorithm significantly outperforms the A-ALNS algorithm. Specifically, the ratio of collected profits of our GRILS is on average 30% higher than that of A-ALNS algorithm, while our GRILS consumes fewer computational time. This advantage can be explained from two aspects: On the one hand, our algorithm performs better than its underlying single satellite scheduling algorithm (namely ALNS) due to the well-designed INSERT operator; On the other hand, the flexible strategy which allocates the targets with duplicated observations to specific orbits can effectively improve the solution quality.

D. Sensitivity Analysis

In this section, we discuss the impact of the different parameters on the performance of the presented algorithm. Tables V, VI and VII individually report the performance of our algorithm while different values of *StartGreed*, *GreedyRange* and *DecreaseGreedy* are set. The chosen configuration of these parameters are 1, 0.2 and 0.02, respectively. Table VIII shows the impact of different settings of the angular velocities. The maximum number of iteration without improvements is set to 300. The presented results are obtained using the single satellite instance with 600 targets and Chinese distribution.

Table V presents the impact of the *StartGreed* parameter. It can be seen that as the value of this parameter increases, the solution quality gradually decreases. Thus, the best choice of this parameter is 1. The computational time remains stable since this parameter has little influence on the execution time.

Table VI presents the results of different values of *GreedyRange*. It can be clearly seen that the solutions

TABLE V
Impact of *StartGreedy*

<i>StartGreedy</i>	P_s	Cpu (s)
1	1552.6	126.63
0.95	1526.8	102.16
0.9	1509.2	104.64
0.85	1482.2	104.27
0.8	1469.4	103.97
0.75	1471.6	103.17

is improving and the computation time increases when *GreedyRange* increases. Increasing this parameter from 0.2 to 0.25 leads to higher computational time while only a slight improvement is obtained. Thus, the value of 0.2 is chosen for the best value of *GreedyRange*.

TABLE VI
Impact of *GreedyRange*

<i>GreedyRange</i>	P_s	Cpu (s)
0.1	1546	88.14
0.15	1546.2	109.62
0.2	1552.6	126.63
0.25	1553.2	163.66
0.3	1553.4	180.80
0.35	1554	201.80
0.4	1554.4	231.32

Table VII presents the influence of the *DecreaseGreedy*. Increasing this parameter leads to less frequency of recalling of the ILS algorithm, and thus the solution quality will get worse. However, a high value of *DecreaseGreedy* increases the computational time. From the results, setting this value from 0.01 to 0.02 will significantly decrease the computational time while the superior solution quality is maintained.

TABLE VII
Impact of *DecreaseGreedy*

<i>DecreaseGreedy</i>	P_s	Cpu (s)
0.01	1555.6	233.37
0.02	1552.6	126.63
0.03	1544	103.19
0.04	1548.6	86.93
0.05	1544.6	76.50

In order to investigate the impact of the angular velocities on the performance of our algorithm, we multiply the velocities v_1, v_2, v_3, v_4 by a factor α . To preserve the continuity of function (1), we keep the constant 11.66 and adjust other constant terms with the following the equations:

$$\begin{cases} a_1 = 11.66 - 10/v_1 \\ a_2 = 30/v_1 - 30/v_2 + a_1 \\ a_3 = 60/v_2 - 60/v_3 + a_2 \\ a_4 = 90/v_3 - 90/v_4 + a_3 \end{cases} \quad (22)$$

Table VIII shows the comparative results. Unsurprisingly, the solution quality increases as the angular velocities increase. It means the faster the mobility of the

camera is, the stronger observation capability the satellite has. We can also observe that as α decreases, the decrease in the solution quality becomes larger, which implies that in this case, increasing transition time limits the options for the INSERT operator. This also explains the lower Cpu time for lower values of α .

TABLE VIII
Impact of angular velocity

α	P_s	Cpu (s)
0.5	1428.4	45.56
1	1556.8	74.05
2	1642	103.47
4	1733.2	125.90

VI. Conclusions

This work proposed a Greedy Randomized Iterated Local Search (GRILS) algorithm to solve the Agile Earth Observation Satellite scheduling problem with time-dependent transition time. We model the time-dependent transition time from two perspectives: the FIFO rule and the triangle inequalities. We prove that both of these rules are respected under the given transition time function and the satellite parameters. These results support the introduction of the “minimal transition time” to replace the actual transition time. Due to the presence of time windows characteristic, we present a fast insert operator to avoid a full feasibility check for each insert attempt, and an assignment procedure to address the multi-orbit uniqueness constraint. A shake operator is defined to enable the algorithm to escape from a local optimum. Extensive experiments demonstrate that our algorithm outperforms two state-of-the-art algorithms on both the single satellite instances and the multi-satellite instances. Furthermore, our GRILS algorithm only requires very short computational times.

Further research could focus on developing exact algorithms which can possibly solve medium-sized instances within an acceptable time. Advanced cutting planes can be proposed with respect to the problem-specific characteristics. Moreover, uncertainty can be introduced to the problem, which provides huge practical values and theoretical challenges.

References

- [1] G. Peng, R. Dewil, C. Verbeeck, A. Gunawan, L. Xing, and P. Vansteenwegen, “Agile earth observation satellite scheduling: An orienteering problem with time-dependent profits and travel times,” *Computers & Operations Research*, vol. 111, pp. 84–98, 2019.
- [2] E. Bensana, M. Lemaitre, and G. Verfaillie, “Earth observation satellite management,” *Constraints*, vol. 4, no. 3, pp. 293–299, Sep 1999.
- [3] M. Lemaitre, G. Verfaillie, F. Jouhaud, J.-M. Lachiver, and N. Bataille, “Selecting and scheduling observations of agile satellites,” *Aerospace Science and Technology*, vol. 6, no. 5, pp. 367–381, 2002.
- [4] B. Dilikina and B. Havens, “Agile satellite scheduling via permutation search with constraint propagation,” 05 2005.

- [5] X. Liu, G. Laporte, Y. Chen, and R. He, "An adaptive large neighborhood search metaheuristic for agile satellite scheduling with time-dependent transition time," *Computers Operations Research*, vol. 86, pp. 41–53, 2017.
- [6] L. He, X. Liu, G. Laporte, Y. Chen, and Y. Chen, "An improved adaptive large neighborhood search algorithm for multiple agile satellites scheduling," *Computers & Operations Research*, vol. 100, pp. 12–25, 2018.
- [7] C. Malandraki and M. S. Daskin, "Time dependent vehicle routing problems: Formulations, properties and heuristic algorithms," *Transportation Science*, vol. 26, no. 3, pp. 185–200, 1992.
- [8] C. Verbeeck, P. Vansteenwegen, and E.-H. Aghezzaf, "The time-dependent orienteering problem with time windows: a fast ant colony system," *Annals of Operations Research*, vol. 254, no. 1, pp. 481–505, Jul 2017.
- [9] N. Bianchessi, J.-F. Cordeau, J. Desrosiers, G. Laporte, and V. Raymond, "A heuristic for the multi-satellite, multi-orbit and multi-user management of earth observation satellites," *European Journal of Operational Research*, vol. 177, no. 2, pp. 750–762, 2007.
- [10] P. Wang, G. Reinelt, P. Gao, and Y. Tan, "A model, a heuristic and a decision support system to solve the scheduling problem of an earth observing satellite constellation," *Computers & Industrial Engineering*, vol. 61, no. 2, pp. 322–335, 2011.
- [11] X. Chen, G. Reinelt, G. Dai, and A. Spitz, "A mixed integer linear programming model for multi-satellite scheduling," *European Journal of Operational Research*, vol. 275, no. 2, pp. 694–707, 2019.
- [12] J.-F. Cordeau and G. Laporte, "Maximizing the value of an earth observation satellite orbit," *Journal of the Operational Research Society*, vol. 56, no. 8, pp. 962–968, Aug 2005.
- [13] D. Habet, M. Vasquez, and Y. Vimont, "Bounding the optimum for the problem of scheduling the photographs of an agile earth observing satellite," *Computational Optimization and Applications*, vol. 47, no. 2, pp. 307–333, Oct 2010.
- [14] W. J. Wolfe and S. E. Sorensen, "Three scheduling algorithms applied to the earth observing systems domain," *Management Science*, vol. 46, no. 1, pp. 148–166, 2000.
- [15] W.-C. Lin, C.-Y. Liu, D.-Y. Liao, and Y.-Y. Lee, "Daily imaging scheduling of an earth observation satellite," in *Systems, Man and Cybernetics*, 2003. IEEE International Conference on, vol. 2, Oct 2003, pp. 1886–1891 vol.2.
- [16] L. Wang, C. Jiang, L. Kuang, S. Wu, L. Fei, and H. Huang, "Mission scheduling in space network with antenna dynamic setup times," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 55, no. 1, pp. 31–45, 2019.
- [17] L. Wang, C. Jiang, L. Kuang, S. Wu, and S. Guo, "TDRSS scheduling algorithm for non-uniform time-space distributed missions," in *2017 IEEE Global Communications Conference, GLOBECOM 2017*, Singapore, December 4-8, 2017. IEEE, 2017, pp. 1–6.
- [18] C. Pralet and G. Verfaillie, "Time-dependent simple temporal networks: Properties and algorithms," *RAIRO-Operations Research*, vol. 47, no. 2, p. 173–198, 2013.
- [19] W. Souffriau, P. Vansteenwegen, G. Vanden Berghe, and D. Van Oudheusden, "The multiconstraint team orienteering problem with multiple time windows," *Transportation Science*, vol. 47, no. 1, pp. 53–63, 2013.
- [20] T. Ibaraki, S. Imahori, M. Kubo, T. Masuda, T. Uno, and M. Yagiura, "Effective local search algorithms for routing and scheduling problems with general time-window constraints," *Transportation Science*, vol. 39, no. 2, pp. 206–232, 2005.
- [21] H. Hashimoto, M. Yagiura, and T. Ibaraki, "An iterated local search algorithm for the time-dependent vehicle routing problem with time windows," *Discrete Optimization*, vol. 5, no. 2, pp. 434–456, 2008.
- [22] P. Vansteenwegen, W. Souffriau, G. V. Berghe, and D. V. Oudheusden, "Iterated local search for the team orienteering problem with time windows," *Computers & Operations Research*, vol. 36, no. 12, pp. 3281–3290, 2009.
- [23] M. Gendreau, G. Laporte, and F. Semet, "A tabu search heuristic for the undirected selective travelling salesman problem," *European Journal of Operational Research*, vol. 106, no. 2, pp. 539–545, 1998.

Guansheng Peng received his bachelor's degree in Management Engineering and masters degree in Management Science and Engineering in 2013 and 2015, both from National University of Defense Technology, Changsha, China. He is currently pursuing his PhD in the same university. He has been a visiting researcher at KU Leuven, Belgium from 2017 to 2019. His research interests include mathematical programming, scheduling and planning of satellites, heuristic algorithms.

Guopeng Song is a lecturer at National University of Defense Technology, Changsha, China. He received his bachelor's degree in Systems Engineering and master's degree in Management Science and Engineering in 2013 and 2015, both from National University of Defense Technology. He holds a PhD (2019) in Business Economics from KU Leuven, Belgium. He has been a visiting researcher at the Hungarian Academy of Sciences (HU), 2018). His research has focused on mathematical

programming, optimization under uncertainty and their applications in planning and scheduling.

Yongming He received the bachelor's degree in Logistics Engineering from Chang'an University, China, in 2014. He is now pursuing his PhD degree in Management Science and Engineering at the College of Systems Engineering, National University of Defense Technology, China. He is a visiting Ph.D. student at University of Alberta, Edmonton, AB, Canada, from Nov. 2018 to Nov. 2019. His research interests include operations research, artificial intelligence, intelligent decision, task

scheduling and planning.

Jing Yu received the bachelor's degrees in statistics from Qufu Normal University, Qufu, China, in 2013, and the master's degree in mathematics from Changsha University of Science and Technology, Changsha, China, in 2016, where she is currently working toward the PhD degree in management science and engineering from College of Systems Engineering, National University of Defense Technology, Changsha, carrying out research in intelligent optimization methods and scheduling of

resources.

Shang Xiang is a PhD candidate at the College of Systems Engineering, National University of Defense Technology. He received his master degree from Xiangtan University in 2017. His research interest covers system planning and management decision making techniques.

Lining Xing received the bachelor's degrees in Economics and in Science from Xian Jiao-Tong University, Xian, China, in 2002, and the PhD degree in Management Science from National University of Defense Technology, Changsha, China, in 2009. Currently, he is a professor at National University of Defense Technology, Changsha, China. He visited the School of Computer, University of Birmingham, U.K., from Nov. 2007 to Nov. 2008. His research interests include intelligent optimization methods, and scheduling of resources.

Pieter Vansteenwegen is an associate professor at the Centre of Industrial Management, Traffic and Infrastructure at KU Leuven and chair of the KU Leuven Mobility Research Center. He obtained a master degree in Mechanical Engineering at KU Leuven in 2002 and a PhD in Operations Research at KU Leuven in 2008. His research focuses on the development of metaheuristics for solving complex logistic and public transportation planning problems.