Optim Lett
DOI 10.1007/s11590-014-0744-8

ORIGINAL PAPER

On the tractability of satellite range scheduling

Antonio J. Vazquez · R. Scott Erwin

Received: 4 October 2013 / Accepted: 1 April 2014

Abstract Scheduling of contacts between several satellites and ground stations has been historically sub-optimally approached. This fact raises the question: can this problem be solved in polynomial time? Although existing literature provided optimal solutions for some simplified versions of this problem and studied some connections with general scheduling, few has been said on the complexity of more general cases. We formally characterize the complexity of the satellite scheduling problem, and provide the sufficient conditions for polynomial time solvability. We back up these results with a survey on several problem instances covering existing and new connections between these problems and those from general scheduling, which allow us to formally define how satellite scheduling relates to general scheduling. To the best of our knowledge, this is the most extensive characterization of satellite range scheduling, covering its definition, complexity and connections to general scheduling.

Keywords Scheduling algorithm · Earth observation satellite · Ground station network

This research was performed while the author held a National Research Council Research Associateship Award at the Air Force Research Laboratory (AFRL).

A. J. Vazquez (✉)
National Research Council, Albuquerque, NM , USA
e-mail: ajvazquez.teleco@gmail.com

R. S. Erwin
Air Force Research Laboratory, Space Vehicles Directorate, AFRL/RVSV,
3550 Aberdeen Ave. SE, Kirtland AFB, Albuquerque, NM 87117-5776, USA

Published online: 10 April 2014

1 Introduction

Scheduling of the interactions between spacecraft and Earth-bound entities arises in multiple applications, including remote sensing (highly related to Earth observation satellites (EOS) [1]) and satellite operations (mainly those based on ground station networks (GSN) [2]). In remote sensing the scheduling problem is to schedule imaging of particular Earth locations; and in satellite operations the issue is the scheduling of communication times with ground stations. For problems where there are multiple scheduling options (e.g. more than one Earth location in view of a satellite), producing an optimal schedule that efficiently uses limited resources is nontrivial. This is exacerbated when trying to optimally schedule multiple constrained resources into an optimal integrated schedule.

Whereas ground stations can be considered to be moving with the surface of the Earth, satellites travel through different kinds of orbits generating visibility time windows when lines of sight to ground stations exist. Interactions are defined by these visibility windows and by time constraints specified by the operators of the satellites, which depending on the requirements of the specific mission require fixed or variable interaction times that can extend through the whole time window or just a portion of it.

Several instances of this general problem have been studied for the last decades [1–7], but different notations and vocabulary make comparing techniques difficult. This has been noticed for general scheduling [8] (specifically for fixed interval scheduling), and existing bibliography provided some steps on this generalization of the notation [1–4].

We present a formal classification of satellite range scheduling (SRS) based on the main parameters studied in the literature: number of resources, preemption, slack, redundancy, precedence and priority. Whereas previous studies focus on suboptimal solutions for discrete time multiple resource scheduling, we establish the frontiers on the complexity for the SRS problem.

In summary, the aim of this paper is to fill the mathematical gap between general and satellite scheduling, and the paper is the result of three main efforts: convergence of criteria in the definition of the problem, search for the conditions for tractability, and survey on previous works.

The paper is organized as follows. In Sect. 2 we establish a formal framework, based on existing literature, on which we rely to classify SRS in terms of complexity; in Sect. 3 we complete the unification of the notations from satellite and general scheduling, started for simplified versions of the problem in previous research works; in Sect. 4 we survey existing and new relations among problems from both fields, which we summarize in Sect. 5 for easy consultation; and in Sect. 6 we conclude this work formally defining the relation between SRS and general scheduling.

2 Problem formulation

This section presents a mathematical formulation of the problem which will support the definitions for the most important branches in SRS problems. Specifically, a dis-

cretization of the problem will be presented (as this is the main general approach in the SRS literature), which will allow to deal with a discrete problem rather than with continuous time optimization.

Let $S = \{s_h\}$ be a set of satellites, and $G = \{g_i\}$ a set of ground stations, t_0 a time instant and T a time window such that $t \in [t_0, t_0 + T]$. Let $J = \{\mathrm{j}_j\}$ be a set of N interaction (communication or sensing) requests:

$$J = \{\mathrm{j}_j\} \ : \ \mathrm{j}_j \triangleq (s_h, g_i, r_j, d_j, \underline{\rho_j}, \overline{\rho_j}, f_j^{\mathrm{w}}(t)), \tag{2.1}$$

where the indexes $h, i, j, N \in \mathbb{N}$, with $h \in [1, |S|]$, $i \in [1, |G|]$, $j \in [1, N]$, the lower and upper bounds for the durations $\underline{\rho_j}, \overline{\rho_j} \in \mathbb{R}$ subject to $0 \leqslant \underline{\rho_j} \leqslant \overline{\rho_j} \leqslant d_j - r_j$ where r_j and d_j are the release and due times of the request. The term $f_j^{\mathrm{w}}(t)$ characterizes the weight or priority (we use these two terms interchangeably) associated to that request, normalized between 0 and 1. This function is introduced in [1], providing a more general approach than simply constant priorities $f_j^{\mathrm{w}}(t) = w_j$.

No *preemption* is allowed, which means that requests cannot be interrupted while being served.

A general classification is done in the literature depending on the number of scheduling entities ($|G|$ and $|S|$).

Definition 1 *Single resource range scheduling problems* (SiRRSP) apply to those scenarios where there is only one satellite or only one ground station, *multiple resource range scheduling problems* (MuRRSP) to those where there are both multiple satellites and ground stations:

$$\text{SiRRSP} \Leftrightarrow (|S| = 1) \vee (|G| = 1), \tag{2.2}$$

$$\text{MuRRSP} \Leftrightarrow (|S| \geqslant 1) \wedge (|G| \geqslant 1). \tag{2.3}$$

From this definition, the SiRRSP can be seen as a subproblem of the MuRRSP.

Also, a general classification is provided in the literature regarding the values of the required durations for the requests ($\underline{\rho_j}$ and $\overline{\rho_j}$).

Definition 2 In the *no-slack* case the requested duration of the passes extends through the whole visibility window. In the *fixed-slack* case, the requested duration of the passes is fixed for each pass, but it can be smaller than the visibility window. And in the *variable-slack* case, the requested duration of the passes varies between two specified boundaries for each pass. According to the current notation:

$$\text{No slack} \Leftrightarrow 0 < \underline{\rho_j} = \overline{\rho_j} = d_j - r_j \ \forall \mathrm{j}_j \in J, \tag{2.4}$$

$$\text{Fixed slack} \Leftrightarrow 0 < \underline{\rho_j} = \overline{\rho_j} \leqslant d_j - r_j \ \forall \mathrm{j}_j \in J, \tag{2.5}$$

$$\text{Variable slack} \Leftrightarrow 0 < \underline{\rho_j} \leqslant \overline{\rho_j} \leqslant d_j - r_j \ \forall \mathrm{j}_j \in J. \tag{2.6}$$

From this definition the no-slack case is a subproblem of the fixed-slack case, which is a subproblem of the variable-slack case.

In some references [7] the fixed-slack case is associated to requests on high altitude orbits, and the no-slack one (also known as *fixed-interval scheduling* [8]) to requests on low altitude orbits. This is due to the short duration of the contact windows of the lower orbits, which generally makes necessary to extend the interaction through the whole window.

In order to build a common framework for the three slack cases, a transformation is presented to generate a set of requests of fixed start and end times (which is a generalization of the one presented in ref. [1]), thus converting every slack problem into a no-slack scheduling problem through discretization. Let Δt be the discretization step, and Δt^{-1} the inverse of Δt.

Transformation 1 Every request j_j will be associated with a set $P_j = \{p_k\}_j$ of *passes*. Let D_n describe this association:

$$D_n : \mathrm{j}_j \longrightarrow P_j = \{p_k\}_j \ : \ p_k \triangleq (s_h, g_i, n_{\mathrm{s}_k}, n_{\mathrm{e}_k}, w_k), \tag{2.7}$$

where s_h and g_i are the same as in the originating request j_j; $n_{\mathrm{s}_k}, n_{\mathrm{e}_k}, q_{j_k} \in \mathbb{Z}^*$ satisfy $n_{\mathrm{e}_k} = n_{\mathrm{s}_k} + q_{j_k}$ with $\lfloor \Delta t^{-1} r_j \rfloor \leqslant n_{\mathrm{s}_k} \leqslant n_{\mathrm{e}_k} \leqslant \lfloor \Delta t^{-1} d_j \rfloor$; and with $w_k \in \mathbb{R} \cap [0, 1]$.

Then, the pass p_k is a tuple entailing a satellite s_h and a ground station g_i (both taken from j_j), discrete start and end times (which are the integers n_{s_k} and n_{e_k}), a duration q_{j_k} (which is the difference among these discrete times), and a weight w_k. Thus the set P_j is the set of all the passes whose discrete start and end times comply with the conditions on the release and due times and with the durations constraints established in the request j_j following expression (2.1).

Proposition 1 *The transformation of the space of discrete slack requests into discrete no-slack passes is polynomial in the number of requests.*

Proof For practical purposes let us assume, without loss of generality, that the values w_k are obtained in polynomial time from a discretized version of the suitability function $f_j^{\mathrm{w}}(n\Delta t)\ \forall n \in \mathbb{Z}^*$. If this were not true, we could modify $f_j^{\mathrm{w}}(n\Delta t)$ to fit this constraint since no optimality claims have been stated.

Let $n_{\mathrm{r}_j} = \lfloor \Delta t^{-1} r_j \rfloor$ and $n_{\mathrm{d}_j} = \lfloor \Delta t^{-1} d_j \rfloor$ be the release and due discrete times for a request j_j, and q_{j_k} the duration of the pass $p_k \in P_j$. The number of windows p_k associated to each request j_j can be obtained [1], from transformation 1:

$$|P_j| = \sum_k (n_{\mathrm{d}_j} - n_{\mathrm{r}_j} - (q_{j_k} - 1)). \tag{2.8}$$

Let $\overline{\overline{q_j}} = n_{\mathrm{d}_j} - n_{\mathrm{r}_j}$ and $\overline{\overline{\rho_j}} = d_j - r_j$. The result of the summation (2.8) can be easily solved [1] (n_{d_j} and n_{r_j} are constant values, so the sum reduces to an arithmetic finite series of q_{j_k}), where the low and high bounds for the sizes of the windows q_{j_k} for the worst case (variable-slack) are $\underline{q_{j_k}} = 1$ and $\overline{q_{j_k}} = \overline{\overline{q_j}}\ \forall k$ is:

$$|P_j| = \frac{1}{2}\left(\left\lfloor \Delta t^{-1} \overline{\overline{\rho_j}} \right\rfloor^2 + \left\lfloor \Delta t^{-1} \overline{\overline{\rho_j}} \right\rfloor\right). \tag{2.9}$$

Applying transformation 1 to to the set J will generate a group $P = \{P_j\}$. Let $\overline{\overline{q}} = \max(\overline{\overline{q_j}})$ and $\overline{\overline{\rho}} = \max(\overline{\overline{\rho_j}})$. Then:

$$|P| = \sum_j |P_j|, \tag{2.10}$$

$$|P| \leqslant \tfrac{1}{2} N \left(\left\lfloor \Delta t^{-1} \overline{\overline{\rho}} \right\rfloor^2 + \left\lfloor \Delta t^{-1} \overline{\overline{\rho}} \right\rfloor \right). \tag{2.11}$$

Applying the slack definitions (Def. 2) on the sum (2.8) allows us to calculate the order of the transformation for every case as in inequality (2.10). Let J_{VS}, J_{FS}, J_{NS} be the sets of requests constrained to variable-slack, fixed-slack and no-slack respectively. Then:

$$|D_n(J_{\mathrm{VS}})| = O\left(N \left\lfloor \Delta t^{-1} \overline{\overline{\rho}} \right\rfloor^2\right), \tag{2.12}$$

$$|D_n(J_{\mathrm{FS}})| = O\left(N \left\lfloor \Delta t^{-1} \overline{\overline{\rho}} \right\rfloor\right), \tag{2.13}$$

$$|D_n(J_{\mathrm{NS}})| = O(N). \tag{2.14}$$

And thus, the transformation D_n is polynomial in the number of requests N. □

Let $P = \{P_j\}$ be the space of all the possible *passes* (or interaction requests with fixed times), and every subset $P_{\mathrm{sub}} \subseteq P$ a *schedule* (or set of passes to be tracked by the network).

2.1 Schedule feasibility

The schedule defined in previous subsection is not necessarily feasible, so that constraints have to be defined to completely specify the SRS problem. This subsection will present those constraints to finally provide a definition of feasible schedule.

For the sake of clarity, for each $p_k \in P_{\mathrm{sub}} \subseteq P = \{P_j\}$, the functions ϕ_g and ϕ_s are defined (e.g. $\phi_g : p_k \longrightarrow g_i : p_k = (s_h, g_i, n_{s_k}, n_{e_k}, w_k)$) for accessing the applicable elements in the tuple (so that $\phi_g(p_k) = g_i$ and $\phi_s(p_k) = s_h$), and the inverse D_n^{-1} for accessing the request originating the pass ($D_n^{-1} : p_k \longrightarrow \mathrm{j}_j \ : \ p_k \in D_n(\mathrm{j}_j)$). Note that this check is necessary for avoiding conflicts in passes generated from the same request. Allowing these conflicts is known in the literature as *preemption*, which as stated previously will not be considered in SRS.

Let C_{g} and C_{s} be conflict indicator boolean functions which yield a 1 if two passes $p_u, p_v \in P_{\mathrm{sub}} : u, v \in \mathbb{N} \cap [1, |P_{\mathrm{sub}}|]$ are generated by the same request or overlapping in time for a single ground station (C_{g}) or satellite (C_{s}):

$$C_{\mathrm{g}} : p_u, p_v \longrightarrow \begin{cases} 1, & \text{if } [D_n^{-1}]_{u,v} \vee \left([n]_{u,v} \wedge [g]_{u,v}\right), \\ 0, & \text{otherwise,} \end{cases} \tag{2.15}$$

$$C_{\mathrm{s}} : p_u, p_v \longrightarrow \begin{cases} 1, & \text{if } [D_n^{-1}]_{u,v} \vee \left([n]_{u,v} \wedge [s]_{u,v}\right), \\ 0, & \text{otherwise,} \end{cases} \tag{2.16}$$

where the bracketed boolean variables

$$[D_n^{-1}]_{u,v} = 1 \Leftrightarrow D_n^{-1}(p_u) = D_n^{-1}(p_v), \tag{2.17}$$
$$[g]_{u,v} = 1 \Leftrightarrow \phi_g(p_u) = \phi_g(p_v), \tag{2.18}$$
$$[s]_{u,v} = 1 \Leftrightarrow \phi_s(p_u) = \phi_s(p_v), \tag{2.19}$$
$$[n]_{u,v} = 1 \Leftrightarrow \{(n_{s_v} \in [n_{s_u}, n_{e_u}]) \vee (n_{e_v} \in [n_{s_u}, n_{e_u}])\}, \tag{2.20}$$

identify the types of conflicts (same request, same ground station, same satellite or time-overlapping).

Let $p_k \times P_{\text{sub}}$ be the Cartesian product of the element p_k with the set P_{sub}, so that $p_k \times P_{\text{sub}} = \{p_k, p_v\} \ : \ p_k, p_v \in P_{\text{sub}} \ \forall k, v \in \mathbb{N} \cap [1, |P_{\text{sub}}|]$. Now let C_{G} and C_{S} be the functions which generate the number of conflicts of a schedule P_{sub}, only for ground stations in the first case and only for satellites in the second:

$$C_{\text{G}} : p_k, P_{\text{sub}} \longrightarrow \sum C_{\text{g}}(p_k \times P_{\text{sub}}) = \sum_{v=1}^{|P_{\text{sub}}|} C_{\text{g}}(p_k, p_v) \ : \ k \neq v, \tag{2.21}$$

$$C_{\text{S}} : p_k, P_{\text{sub}} \longrightarrow \sum C_{\text{s}}(p_k \times P_{\text{sub}}) = \sum_{v=1}^{|P_{\text{sub}}|} C_{\text{s}}(p_k, p_v) \ : \ k \neq v. \tag{2.22}$$

Combining the functions C_{G} and C_{S}, and simplifying the notation, let C_Σ be the function which generates the total number of conflicts of a schedule P_{sub}, summing the number of conflicts for ground stations and satellites. Thus, for every $p_k \in P_{\text{sub}}$:

$$\begin{aligned} C_\Sigma : P_{\text{sub}} \longrightarrow & \sum C_{\text{G}}(P_{\text{sub}}) + \sum C_{\text{S}}(P_{\text{sub}}) \\ & = \sum_k C_{\text{G}}(p_k) + \sum_k C_{\text{S}}(p_k). \end{aligned} \tag{2.23}$$

Definition 3 A *nonredundant* schedule must be free of both kind of conflicts (C_{G} and C_{S}). Alternatively, conflicts of a kind are allowed in the *redundant* schedule (where the subscript x is whether G or S depending on the selected function, C_{G} or C_{S}).

$$\text{No redundancy} \Leftrightarrow C_\Sigma(P_{\text{sub}}) = 0, \tag{2.24}$$
$$\text{Redundancy} \Leftrightarrow C_{\text{x}}(P_{\text{sub}}) = 0. \tag{2.25}$$

From the function (2.23) and constraints (2.24) and (2.25), the nonredundant version is a subproblem of the redundant one, since $C_\Sigma(P_{\text{sub}}) = 0 \ \Leftrightarrow \ C_{\text{G}}(P_{\text{sub}}) = 0 \ \wedge \ C_{\text{S}}(P_{\text{sub}}) = 0$.

Note that the no-redundancy constraint is only applicable to the case where the numbers of ground stations and satellites are greater than one (otherwise it is equivalent to the redundancy constraint), and if not applied, the problem can be split into several one ground station (or satellite) problems. Therefore, no-redundancy will be the general approach for SRS.

If a ground station (or satellite) allows for a number of simultaneous contacts, the entity can be modeled as several nonredundant entities. Another way the literature refers to this classification is the *unitary capacity*, applicable to whether ground stations or satellites (redundant case), or to both simultaneously (nonredundant case).

The redundant request satellite scheduling problem proposed in ref. [3] bounds the number of allowed conflicts, thus laying in the midst of the two presented cases (subproblem of redundant case, generalization of nonredundant one); so that results proved for them will be applicable to it as well.

Additionally, some problems include precedence constraints [4]:

Definition 4 A problem has *precedence* constraints if its definition includes sets P_l^{P} (with $l = 1, 2, \dots$) of passes which can not be on the schedule without each of their elements. Also, by definition, these passes will have end times smaller than the start time of the pass. Otherwise there are *no precedence* constraints.

$$\begin{aligned} \text{No precedence} \Leftrightarrow \nexists P^{\mathrm{P}} : \forall p_k \in P^{\mathrm{P}} \subset P, \\ p_k \in P_{\mathrm{sub}} \Leftrightarrow P^{\mathrm{P}} \subset P_{\mathrm{sub}}, \end{aligned} \tag{2.26}$$

$$\begin{aligned} \text{Precedence} \Leftrightarrow \exists P_l^{\mathrm{P}} : \forall p_k \in P_l^{\mathrm{P}} \subset P, \\ p_k \in P_{\mathrm{sub}} \Leftrightarrow P_l^{\mathrm{P}} \subset P_{\mathrm{sub}}. \end{aligned} \tag{2.27}$$

Precedence constrains are generally external to the problem and defined on the requests, however it is easy to propagate them in transformation 1 to generate the subsets P_l^{P}. By definition precedence is a subproblem of no-precedence, as it includes additional constraints.

Definition 5 A *feasible schedule* P^{f} is a nonpreemptive schedule which meets the constraints for redundancy (redundant or nonredundant) and precedence (precedence constraints existing or not) as specified for the problem.

2.2 Schedule metrics

In order to compare two feasible schedules it is necessary to introduce a metric, which finally allows to define the SRS problem.

Definition 6 The weight or priority w_k introduced in the discretization (2.7) will depend on the weight function of the originating request $f_j^{\mathrm{w}}(t)$ (introduced in ref. [1] as the *suitability function*) for each pass, and on the limits of the new window $(n_{\mathrm{s}_k}, n_{\mathrm{e}_k})$. Furthermore a normalization factor a_k is introduced:

$$w_k \triangleq \sum_{n=n_{\mathrm{s}_k}}^{n_{\mathrm{e}_k}} a_k f_j^{\mathrm{w}}(n\Delta t). \tag{2.28}$$

Prioritization approaches depend on the values of a_k and $f_j^{\mathrm{w}}(n\Delta t)$. Let $\Delta n_k = n_{\mathrm{e}_k} - n_{\mathrm{s}_k}$ and $\Delta n_{\max} = \max(\Delta n_k)\ \forall k$ such that $p_k \in P_j$. A problem is said to have *no priority* constraints if all the requests are assigned the same priority, which for

simplicity is 1 ($a_k = \Delta n_k{}^{-1}$; $f_j^{\mathrm{w}}(n\Delta t) = 1 \Rightarrow w_k = 1$). The general case is said to have *priority* constraints, and takes into account both duration and best spot of the pass ($a_k = \Delta n_{\max}{}^{-1}$; $f_j^{\mathrm{w}}(n\Delta t) = f_j^{\mathrm{suit}}(n\Delta t) \Rightarrow w_k = \Delta n_{\max}{}^{-1} \sum_{n=n_{s_k}}^{n_{e_k}} f_j^{\mathrm{suit}}(n\Delta t)$).

It is easy to see that nonprioritized problems are subproblems of prioritized problems.

Given a schedule P_{sub}, let the metric $\|\cdot\|_{\Sigma w}$ be:

$$\|P_{\mathrm{sub}}\|_{\Sigma w} \triangleq \sum_k w_k \quad \forall p_k \in P_{\mathrm{sub}}. \tag{2.29}$$

Definition 7 The optimal schedule P^* is a feasible schedule with maximal metric.

$$P^* \in \{P^{\mathrm{f}}\};\ \nexists P_{\mathrm{sub}} \in \{P^{\mathrm{f}}\}\ : \|P_{\mathrm{sub}}\|_{\Sigma w} > \|P^*\|_{\Sigma w}\ , \tag{2.30}$$

where $\{P^{\mathrm{f}}\}$ is the set of all the feasible schedules.

Finally, the SRS problem can be defined as finding the optimal schedule. Some references [2,3,5] add to the problem the term *oversubscribed*, when not all the passes can be served ($|P^*| \neq |J|$).

Definition 8 Given the set of all the feasible schedules $\{P^{\mathrm{f}}\}$, the problem of satellite range scheduling (SRS) can be stated as finding the optimal schedule P^*:

$$P^* \triangleq \arg\max(\|P_{\mathrm{sub}}\|_{\Sigma w})\ \forall P_{\mathrm{sub}} \in \{P^{\mathrm{f}}\}. \tag{2.31}$$

2.3 Complexity of SRS

The purpose of this section is to classify the main SRS problems in terms of complexity. Whereas complexity for specific cases have been studied in some references [1,2,4,5,7], most of the literature on SRS focus on suboptimal algorithms. To the best of our knowledge this is the first formal classification on the complexity of SRS.

The first result we provide is obtained by studying the most general case as a generalization of simpler problems from the literature. Then we study the discretized problem, differentiating between arbitrary and fixed number of resources, and show that this difference is crucial for finding an optimal solution in polynomial time. Finally we extend these results to general discretized SRS problems.

Lemma 1 *SRS is NP-hard.*

Proof Reference [2] proves equivalence of the single resource satellite scheduling problem with fixed slack, no preemption, no redundancy, no precedence and no priorities to the general scheduling problem of the minimization of the number of late jobs in one machine where tasks have different release times. This last problem is classified as NP-hard in refs. [4,9].

From the reducibility relations given in Definitions 1, 3 and 6, the generalized problems with variable slack, multiple resources, redundancy and priorities have to be

at least as complicated as the subproblem, so that they are also NP-hard. The no-slack case is proven to be polynomial for a single resource in ref. [2] (and we prove later in this paper that also its multiple resource version is in P, but these are subproblems of the fixed-slack version). NP-hardness for the precedence case is proven in ref. [4], completing the proof for Slack SRS. Since this is a subproblem of SRS, it can be stated without loss of generality that SRS is NP-hard. □

Theorem 1 *Discrete no-slack SRS is NP-complete for arbitrary numbers of ground stations and satellites.*

Proof The aim of this demonstration is to show equivalence of this problem to an existing NP-complete problem. We start with the subproblem no-slack MuRRSP with no redundancy, no preemption, no precedence and no priority constraints. A graph will be created from the initial set of passes P. Every pass will be mapped into a node in the graph (due to the no-slack condition $|P| = |J| = N$ nodes). Edges will be created between every two nodes if they are not conflicting $C_g(p_u, p_v) = 0$ and $C_s(p_u, p_v) = 0$ for every $p_u, p_v \in P$. Then finding a feasible schedule of size M is equivalent to finding a clique of size M in the graph (known as the "Clique" problem [10]), which is NP-complete.

Finding the optimal schedule, from Def. 7, is equivalent to solve this decision problem at most N times (for $M = 1, 2, \ldots, N$), and then selecting the solution with the highest metric (which for the nonprioritized case is the number of passes). Both the calculation of the metric and the selection of the highest value can be computed in polynomial time. Then if algorithm A solves the decision problem in $O(f_A(N))$ time (where $f_A(N)$ is a function of N), the maximal size problem can be solved in $O(Nf_A(N))$ time, and thus it is also NP-complete. Thus, the no-slack MuRRSP problem with no redundancy, no preemption, no precedence and no priority is NP-complete.

The result is also extensible to the redundant case by deleting one of the conditions for the creation of an edge ($C_g(p_u, p_v) = 0$ or $C_s(p_u, p_v) = 0$) during the graph generation; and also to the precedence case, by grouping into an only node all the passes that are in the same precedence subset P_l^P (see Def. 4), and creating edges from this new node to those nodes outside this subset which had edges with every node in the subset. This guarantees that all the passes in the same subset are included or excluded from the final schedule. Additionally, if a pass belongs to two precedence subsets, they will be merged together into an only node. Then, for every generated node, it has to be checked that its members are a clique (i.e. they do not conflict), and if they are not, they can be deleted from the graph. These operations are polynomial on the number of nodes.

And finally, the result is also extensible to the prioritized case by introducing priorities in the passes and effectively computing the highest metric. Thus, no-slack SRS is NP-complete for arbitrary numbers of stations and satellites. □

The problem is simplified though if the number of ground stations or satellites is fixed:

Theorem 2 *Discrete no-slack SRS is in class P for a fixed number of ground stations or satellites.*

Proof This problem will be shown to be equivalent to another problem known to be solvable by a polynomial time algorithm. Suppose a discrete no-slack MuRRSP problem with priorities and allowing for redundancy (only for satellites or ground stations) but not for precedence. Equivalently, priorities are assigned to the passes, only conflicts of a kind are allowed (C_G or C_S) and passes are mutually independent. This problem corresponds to the general scheduling problem defined in ref. [9], where machines can be associated to whether ground stations (if conflicts are allowed in C_S) or satellites (if conflicts are allowed in C_G). The only difference is that in the SRS version every request is assigned to an only "machine" due to the structure of the requests (2.1), which makes the problems easier as fewer states are possible in the graph. Note that conflicts are allowed for the entities with variable number, so that this number is not relevant.

Given that the problem from ref. [9] can be solved in polynomial time for a fixed number of machines, the result is also extensible to the no-slack redundant no-precedence SRS problem with a fixed number of ground stations or satellites.

From Definitions 1, 3, 4 and 6, SiRRSP is a subproblem of MuRRSP, no-redundancy of redundancy, precedence of no-precedence, and no-priority of priority. Therefore, the result is also extensible to the rest of the cases, completing the proof for discretized no-slack SRS. □

Some tedious *manipulation in the notation allows to extend the results from Theorems 1 and 2 to the continuous time versions of the problems*, as there is no difference other than taking discrete values for the times when checking for the conflicts.

The two following results are however constrained to the discretized versions of the problem, as they take advantage of a polynomial transformation (2.7) only available in the discretized problem.

Corollary 1 *Discretized SRS is NP-Complete for arbitrary numbers of ground stations and satellites.*

Proof From Proposition 1, the equivalent set of passes can be generated in polynomial time from the set of discrete slack request. Then following the procedure from Theorem 1 a similar graph can be generated, and following the same reasoning the problem is also NP-complete. □

Corollary 2 *Discretized SRS is in class P for a fixed number of ground stations or satellites.*

Proof From Proposition 1, the discretized slack problem is polynomially transformable to the discretized no-slack problem in polynomial time. Applying Theorem 2 yields that discretized SRS with a fixed number of ground stations or satellites is in polynomial class, since the discretized no-slack version is as well. □

Figure 1 summarizes the complexity and relations among the main SRS problems, with arrows identifying subproblems (origins) and generalizations (ends). Note that slack (fixed or variable) SRS is NP-hard whether with an arbitrary or fixed number of resources (FNR) (see Theorem 1).

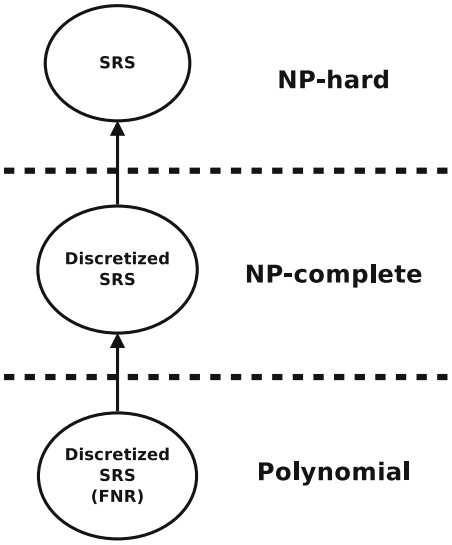

Fig. 1 SRS problems classification

3 General scheduling problems

In this section we study the connections between general and satellite scheduling problems. An existing widely-used notation from general scheduling (presented in ref. [11]) is applied (the notation used so far in this paper has been selected to match this one to the greatest possible extent to ease the reading), thus allowing to formally relate problems from the two scheduling fields.

This notation is summarized for the sake of completeness (and extended to cover SRS specific cases), and a survey is presented to shed light on existing and new relations between specific problems.

3.1 Problem classification

In general scheduling problems [11] there is a set of n jobs $\{\mathfrak{J}_j\}\forall j \in \mathbb{N} \cap [1, n]$ to be processed in a set of $\mathfrak{m}$ machines $\{\mathfrak{M}_i\}\forall i \in \mathbb{N} \cap [1, \mathfrak{m}]$. Furthermore, each job $\mathfrak{J}_j$ has a processing time $\mathfrak{p}_j$ ($\mathfrak{p}_{ij}$ if it is different for each machine $\mathfrak{M}_i$), a release date $\mathfrak{r}_j$, a due date $\mathfrak{d}_j$, a weight (or priority) $\mathfrak{w}_j$ and a cost function $\mathfrak{f}_j(t)$ for evaluating the penalization of non satisfying the request before t.

Note that this notation is similar to the previously introduced, with the following differences: n is used for the number of requests N, the set of $\mathfrak{m}$ machines will be $|S|$ or $|G|$ depending on whether the scheduling entities are the satellites or the ground stations, and the time duration $\mathfrak{p}$ substitutes ρ and its discrete homologous.

Reference [11] defines general scheduling problems based on three parameters: $\alpha|\beta|\gamma$.

– α designates the relation among the machines: *1* for a single machine; $\mathfrak{P}$ for parallel identical machines (processing time for a task is the same for every machine); $\mathfrak{R}$

for unrelated parallel machines (processing times are different and even a task can be only compatible with a subset of machines).

- β covers the constraints on the jobs. Terms grouped in the same bullet are mutually exclusive:
  - $\mathfrak{r}_j$ indicates different release times for every job, which is generally assumed for SRS.
  - we introduce $\overline{\overline{\mathfrak{p}_j}}$, for indicating that the processing time extends from the release date to the due date (no-slack case), $\overline{\overline{\mathfrak{p}_j}} = \mathfrak{d}_j - \mathfrak{r}_j$; and another notation is introduced (for the variable-slack case), $\underline{\mathfrak{p}_j} \leqslant \mathfrak{p}_j \leqslant \overline{\mathfrak{p}_j}$ (existing notation $\underline{\mathfrak{p}} \leqslant \mathfrak{p}_j \leqslant \overline{\mathfrak{p}}$ assumes constant bounds for all the jobs) for indicating that the processing times are allowed to be within lower and upper bounds. The absence of these constraints indicates that $\mathfrak{p}_j$ is fixed and $0 \leqslant \mathfrak{p}_j \leqslant \overline{\overline{\mathfrak{p}_j}}$ (fixed-slack case), which is the general case in general scheduling problems.
  - we also introduce C_Σ, which is specific for SRS, and indicates that no redundancy is allowed in tasks assignable to an only machine (whether station or satellite). As noted in the Definition 3 of no-redundancy, it is indifferent whether to apply it or not in SiRRSP, and if not applied in MuRRSP, the problem is equivalent to solving several instances of the one machine problem due to this constraint on the definition of the passes.
  - The term "prec" introduces precedence constraints, for cases where some tasks require the completion of others to be served.
  - And finally, the term "pmtn" allows for preemption, which is not considered for SRS, and thus will not be used in any case.
- γ describes the optimization function: $\sum \mathfrak{U}_j$ stands for the total number of late jobs ($\mathfrak{U}_j = 1$ if a job $\mathfrak{J}_j$ is late); and $\sum \mathfrak{w}_j \mathfrak{U}_j$ takes also into account the weights assigned to each job in the computation. Minimizing the sum of the weight of late jobs is equivalent to maximizing the sum of the weight of the jobs in the schedule.

3.2 Problem reducibility

Reference [11] also describes the reducibility between the general scheduling problems. In this subsection equivalent relations will be obtained combining those from the reference with the definitions from Sect. 2.

The variable-slack problem $\underline{\mathfrak{p}_{ij}} \leqslant \mathfrak{p}_{ij} \leqslant \overline{\mathfrak{p}_{ij}}$ can be seen as a generalization of the fixed-slack problem $\overline{\overline{\mathfrak{p}_j}}$ also known as knapsack problem) [1]). And given that $\overline{\overline{\mathfrak{p}_j}}$ is a specific case of $\mathfrak{p}_{ij}$ (when $\mathfrak{p}_{ij} = \overline{\overline{\mathfrak{p}_j}}\ \forall j$), and $\mathfrak{p}_j$ is a specific case of $\underline{\mathfrak{p}_{ij}} \leqslant \mathfrak{p}_{ij} \leqslant \overline{\mathfrak{p}_{ij}}$ (when $\underline{\mathfrak{p}_{ij}} = \overline{\mathfrak{p}_{ij}}$):

$$1 \subset \mathfrak{R}, \tag{3.1}$$

$$\sum \mathfrak{U}_j \subset \sum \mathfrak{w}_j \mathfrak{U}_j, \tag{3.2}$$

$$\overline{\overline{\mathfrak{p}_j}} \subset \mathfrak{p}_{ij} \subset \underline{\mathfrak{p}_{ij}} \leqslant \mathfrak{p}_{ij} \leqslant \overline{\mathfrak{p}_{ij}}. \tag{3.3}$$

Some of the relations between general and satellite-specific problems will be summarized in the following subsections. Note that from the previous relations, properties

and references are inherited by the problem subsets, so that references will not be repeated unless explicitly necessary.

3.3 Optimality and performance

There are some differentiations not considered so far, that will be introduced before the survey to completely specify the field where this research is aimed, and to suggest on further connections to be done.

General scheduling literature differentiates between dynamic and static scheduling. *Dynamic Scheduling* considers a schedule window T short enough to decide for only a small set of passes. The most extreme case in dynamic scheduling is known as *Online Scheduling* [12], where $|J| = 1$, so that decisions are made individually for each request. The case where this condition is relaxed is known as *Static Scheduling*.

Another important differentiation is whether there is an only scheduling entity (*Centralized Scheduling*), or several entities working whether cooperatively or competitively (*Distributed Scheduling* [6]). By definition, centralized algorithms focus on optimality whereas distributed do on performance.

Since this text is focused on the optimality of the solutions, *only static-centralized algorithms will be considered.*

4 Relating satellite and general scheduling problems

In this section the relations between general scheduling (GS) and satellite scheduling (SRS) problems are explored. Although specific relations have been found in the literature for some problems (as indicated where applicable), to the best of our knowledge, no reference provides a survey to the extent of the one presented as follows. As usual, *no-preemption is considered for SRS*.

4.1 One machine problems

One machine scheduling problems correspond to different variants of SiRRSP. Note that this constraint on the number of machines eliminates the relevance for the redundancy definitions, so both versions will be equivalent.

- $1 \mid \mathfrak{r}_j, \overline{\overline{\mathfrak{p}_{ij}}} \mid \sum \mathfrak{U}_j$.
  - GS: ref. [9] provides an optimal solution in $O(n^2)$ as a specific case of a more complex problem.
  - SRS: a demonstration of the equivalence of this problem with the *no-slack no-precedence no-priorities SiRRSP* is provided in ref. [7], as well as an *optimal* algorithm in polynomial time for its solution.
- $1 \mid \mathfrak{r}_j, \overline{\overline{\mathfrak{p}_{ij}}} \mid \sum \mathfrak{w}_j \mathfrak{U}_j$.
  - GS: An *optimal* $O(n^2)$ algorithm is provided in ref. [9], based on finding a shortest path in a directed acyclic graph.

  - SRS: a demonstration of the equivalence of this problem to the *no-slack no-precedence prioritized SiRRSP* has been provided in Theorem 2 in this document.
- $1 \mid \mathfrak{r}_j \mid \sum \mathfrak{w}_j \mathfrak{U}_j$.
  - GS: ref. [11] shows NP-hardness of this problem.
  - SRS: this problem is equivalent to *fixed-slack no-precedence prioritized SiRRSP* [2].
- $1 \mid \mathfrak{r}_j, \mathfrak{p}_{ij} \leqslant \mathfrak{p}_{ij} \leqslant \overline{\mathfrak{p}_{ij}} \mid \sum \mathfrak{w}_j \mathfrak{U}_j$.
  - GS: From the reducibility relation (3.3) [11], this problem is at least as complex as the previous one, which is NP-hard.
  - SRS: this problem is specified in ref. [1], and it is by definition equivalent to the *variable-slack no-precedence prioritized SiRRSP*.

4.2 Several identical machines problems

These problems correspond to the case of scheduling in an entity with limited capacities, which for simplicity can be broken down into basic entities with unitary capacity. Specifically, for satellite scheduling, these problems are equivalent to the scenarios where it is possible to find a subset division of the set of requests, such that all the subsets have the same elements:

$$Q = \{Q_l\} \ : \ Q_l = Q'_l \ \forall l, l'. \tag{4.1}$$

An illustrative example of this scenario is scheduling for a ground station with different independent and compatible communication systems, as they share the same position and thus the scheduling tasks can be assigned to each of them. The usefulness of this approach, specially for the case of LEO (Low Earth Orbit) tracking stations, is limited to scenarios where the coverage regions do not overlap.

By definition, these problems are not equivalent to any of the ones which fit on the notation used in this text.

- $\mathfrak{P} \mid \mathfrak{r}_j, \overline{\overline{\mathfrak{p}_{ij}}}, C_\Sigma \mid \sum \mathfrak{U}_j$.
  - GS: ref. [9] provides an optimal solution in $O(n^2 \log(n))$ as a specific case of a more complex problem (redundancy and priorities). Also several references are provided in the survey in Interval Scheduling in ref. [12].
  - SRS: proof of polynomial optimality is provided in ref. [2]. Note that in SRS combining the constraint C_Σ with similar resources and passes assigned to an only resource translates duplicate passes (except for the assigned resource) into single tasks assignable to any machine.

4.3 Several unrelated machines problems

As introduced before, problems involving several unrelated machines are suitable to be applied to different variants of MuRRSP.

Note that in SRS the absence of the constraint C_Σ, along with the unitary limitation for the compatible scheduling entities for every request, transforms the problem into

the centralized resolution of $|G|$ (or $|S|$ if the satellites are considered machines) independent one machine versions of the original problem.

- $\mathfrak{R} \mid \mathfrak{r}_j \overline{\overline{\mathfrak{p}_{ij}}}, C_\Sigma \mid \sum \mathfrak{w}_j \mathfrak{U}_j$.
  - GS: N/A (not applicable), due to the constraint C_Σ this problem is specific to the satellite scheduling domain.
  - SRS: a demonstration of the equivalence of this problem to the *no-slack no-redundant no-precedence prioritized MuRRSP* has been provided in Theorem 2 in this document. Also it is proved that the problem is in class P for a fixed number of machines.
- $\mathfrak{R} \mid \mathfrak{r}_j \overline{\overline{\mathfrak{p}_{ij}}}, C_\Sigma, \text{prec} \mid \sum \mathfrak{w}_j \mathfrak{U}_j$.
  - GS: N/A.
  - SRS: a demonstration of the equivalence of this problem to the *no-slack no-redundant prioritized MuRRSP with precedence priorities* has been provided in Theorem 2 in this document. Same results on the complexity as for the previous problem are applicable here.
- $\mathfrak{R} \mid \mathfrak{r}_j, C_\Sigma \mid \sum \mathfrak{w}_j \mathfrak{U}_j$.
  - GS: N/A.
  - SRS: equivalence to *fixed-slack no-redundant no-precedence prioritized MuRRS* of this problem, as well as NP-hardness, are demonstrated in ref. [4]. A survey on several suboptimal algorithms is provided in ref. [5] for the discretized version of the problem.
- $\mathfrak{R} \mid \mathfrak{r}_j, C_\Sigma, \text{prec} \mid \sum \mathfrak{w}_j \mathfrak{U}_j$.
  - GS: N/A.
  - SRS: equivalence to *fixed-slack redundant prioritized MuRRS with precedence constraints* of this problem, as well as NP-hardness, are demonstrated in ref. [4]. Additional constraints are applied, specifically setup times (which could be modeled on the suitability function and start time) and on-board data storage limits.
- $\mathfrak{R} \mid \mathfrak{r}_j, \mathfrak{p}_{ij} \leqslant \mathfrak{p}_{ij} \leqslant \overline{\mathfrak{p}_{ij}}, C_\Sigma \mid \sum \mathfrak{w}_j \mathfrak{U}_j$.
  - GS: $\overline{\text{N}}$/A.
  - SRS: this problem is defined in ref. [1], and it is by definition equivalent to *variable-slack no-redundant no-precedence prioritized MuRRSP*. As a generalization of the analogous fixed-slack problem, it is at least NP-hard.

5 Summary

Characteristics of the enumerated problems have been tabulated into Table 1 for easy consultation. Although a $\mathfrak{P}$ (several identical machines) problem has been included in this table as MuRRSP (marked as x'), it is a very specific case (all the resources have the same position), so as stated previously, the applicability of this case is found to be reduced.

The notation $\mathfrak{p}_{ij} \leqslant \mathfrak{p}_{ij} \leqslant \overline{\mathfrak{p}_{ij}}$ has been changed to $\mathfrak{p}_{ij}^{\text{var}}$, and some terms have been abbreviated: $\overline{\text{R}}$es. (resource), Si. (single), Mu. (multiple), Redund. (redundancy), Prec. (precedence) Fix. (fixed), Var. (variable) and t.p. (this paper). The complexity is displayed for the three cases studied in Sect. 2.3, t (general case in continuous time),

Table 1 Relations between general and satellite scheduling problems

Problem	Res.		Priority		Prec.		Slack			Class ($t/\Delta t/\Delta t$+FNR)	References	
	Si.	Mu.	No	Yes	No	Yes	No	Fix.	Var.		GS	SRS
$1\|\mathfrak{r}_j,\overline{\overline{\mathfrak{p}_{ij}}}\|\sum \mathfrak{U}_j$	x		x		x		x			P/P/P	[9]	[7]
$1\|\mathfrak{r}_j,\overline{\overline{\mathfrak{p}_{ij}}}\|\sum \mathfrak{w}_j\mathfrak{U}_j$	x			x	x		x			P/P/P	[9]	t.p.
$1\|\mathfrak{r}_j\|\sum \mathfrak{w}_j\mathfrak{U}_j$	x			x	x			x		NPH/P/P	[11]	[2]
$1\|\mathfrak{r}_j,\mathfrak{p}_{ij}^{\text{var}}\|\sum \mathfrak{w}_j\mathfrak{U}_j$	x			x	x				x	NPH/P/P	[11]	[1]
$\mathfrak{P}\|\mathfrak{r}_j,\overline{\overline{\mathfrak{p}_{ij}}},C_\Sigma\|\sum \mathfrak{U}_j$		x'	x		x		x			P/P/P	[9,12]	[2]
$\mathfrak{R}\|\mathfrak{r}_j,\overline{\overline{\mathfrak{p}_{ij}}},C_\Sigma\|\sum \mathfrak{w}_j\mathfrak{U}_j$		x		x	x		x			NPC/NPC/P	N/A	t.p.
$\mathfrak{R}\|\mathfrak{r}_j,\overline{\overline{\mathfrak{p}_{ij}}},C_\Sigma,\text{prec}\|\sum \mathfrak{w}_j\mathfrak{U}_j$		x		x		x	x			NPC/NPC/P	N/A	t.p.
$\mathfrak{R}\|\mathfrak{r}_j,C_\Sigma\|\sum \mathfrak{w}_j\mathfrak{U}_j$		x		x	x			x		NPH/NPC/P	N/A	[4,5]
$\mathfrak{R}\|\mathfrak{r}_j,C_\Sigma,\text{prec}\|\sum \mathfrak{w}_j\mathfrak{U}_j$		x		x		x		x		NPH/NPC/P	N/A	[4]
$\mathfrak{R}\|\mathfrak{r}_j,\mathfrak{p}_{ij}^{\text{var}},C_\Sigma\|\sum \mathfrak{w}_j\mathfrak{U}_j$		x		x	x				x	NPH/NPC/P	N/A	[1]

Δt (discretized case), Δt+FNR (discretized case with fixed number of machines); and the classes of complexity are NPH (for NP-hard), NPC (for NP-complete), and P (for polynomial).

6 Conclusions

In this paper we have provided the first formal classification on the complexity of SRS problems, supported by proofs and references. These proofs are endorsed by a mathematical formulation which has been shown to be equivalent, with some new constraints, to that from general scheduling. A survey has been presented showing existing and new connections between specific instances of general and satellite scheduling problems, backing up previous results.

We have shown that SRS is a subproblem of general scheduling for unrelated machines with no preemption constraints, no redundancy (understood on both kind of entities), and with each request assigned to an only scheduling resource.

SRS for multiple resources has been historically discretized and suboptimally approached. We have shown that these discretized problems can be solved in polynomial time with a fixed number of entities of a kind (ground stations or satellites), which is the general case in practical scenarios.

7 Future work

Based on the demonstrations we provided in this paper, it is easy to see that optimal algorithms from general scheduling (like the one from ref. [9]) can be adapted to satellite range scheduling. We are working on this migration and also on distributed and on-line approaches.

Acknowledgments This research was performed while the author held a National Research Council Research Associateship Award at the Air Force Research Laboratory (AFRL). We thank Configurable Space Microsystems Innovations & Applications Center (COSMIAC, www.cosmiac.org) for the infrastructure support during this research, and we also thank two anonymous reviewers whose comments helped to improve the manuscript.

References

1. Wolfe, W.J., Sorensen, S.E.: Three scheduling algorithms applied to the earth observing systems domain. Manag. Sci. Informs **46**(1), 148–168 (2000)
2. Barbulescu, L., Watson, J.P., Whitley, L.D., Howe, A.E.: Scheduling space-ground communications for the air force satellite control network. J. Sched. **7**(1), 7–34 (2004)
3. Schmidt, M.: Ground Station Networks for Efficient Operation of Distributed Small Satellite Systems. PhD Thesis. University of Wurzburg (2011)
4. Marinelli, F., Rossi, F., Nocella, S., Smriglio, S.: A Lagrangian heuristic for satellite range scheduling with resource constraints. Comput. Oper. Res. **38**(11), 1572–1583 (2005)
5. Barbulescu, L.V.: Oversubscribed Scheduling Problems. Thesis Proposal (2002)
6. Jung, H., Tambe, M., Barret, A., Clement, B.: Enabling efficient conflict resolution in multiple spacecraft missions via DCSP. In: Proceedings of the NASA Workshop on Planning and Scheduling, NASA (2002)
7. Burrowbridge, S.E.: Optimal Allocation of Satellite Network Resources. MSc Thesis. Virginia Polytechnic and State University (1999)
8. Kovalyov, M.Y., Ng, C.T., Edwin, T.C.: Fixed interval scheduling: Models, applications, computational complexity and algorithms. Eur. J. Oper. Res. **178**, 331–342 (2007)
9. Arkin, E.M., Silverberg, E.B.: Scheduling jobs with fixed start and end times. Discret. Appl. Math. **18**, 1–8 (1987)
10. Garey, M.R., Johnson, D.S.: Computers and Introcatability: A guide to the Theory of NP-Completeness. Freeman (1979). ISBN-13 978-0-7167-1044-8
11. Graham, R.L., Lawler, E.L., Lenstra, J.K., Rinnoy, A.H.G.: Optimization and approximation in deterministic sequencing and scheduling: a survey. Ann. Discret. Math. Discret. Optim. II **5**, 287–326 (1979)
12. Kolen, A.W.J., Lenstra, J.K., Papadimitrou, C.H., Spieksma, F.C.R.: Interval scheduling: a survey. Nav. Res. Log. **54**(5), 530–543 (2007)