A Heuristic for the Multi-Satellite, Multi-Orbit and Multi-User Management of Earth Observation Satellites

Nicola Bianchessi[1], Jean-François Cordeau[2], Jacques Desrosiers[3], Gilbert Laporte[2*] and Vincent Raymond[2]

[1] Dipartimento di Tecnologie dell'Informazione, Università degli Studi di Milano
Via Bramante, 65, I-26013 Crema (CR), Italy
bianchessi@dti.unimi.it

[2] Canada Research Chair in Distribution Management and GERAD, HEC Montréal
3000, chemin de la Côte-Sainte-Catherine, Montréal, Canada H3T 2A7
{cordeau,gilbert,vraymond}@crt.umontreal.ca

[3] GERAD, HEC Montréal
3000, chemin de la Côte-Sainte-Catherine, Montréal, Canada H3T 2A7
jacques.desrosiers@gerad.ca

December 2, 2005

Abstract

Earth observation satellites are platforms equipped with optical instruments that orbit the Earth in order to take photographs of specific areas at the request of users. This article is concerned with the management of several satellites performing multiple orbits over a given planning horizon. It describes a tabu search heuristic for the problem of selecting and scheduling the requests to be satisfied, under operational constraints. An upper bounding procedure based on column generation is used to evaluate the quality of the solutions. The results of extensive computational experiments performed on data provided by the French Centre National d'Études Spatiales are reported.

Keywords: Earth observation satellites, multiple orbits, multiple users, tabu search heuristic, column generation.

*Corresponding author

1 Introduction

Earth observation satellites are platforms equipped with optical instruments that orbit the Earth in order to take photographs of specific areas at the request of users. To each request are associated a time window during which the photograph can be taken, and a potential profit representing the value to the user of acquiring the photograph. Because the number of requests typically exceeds what can feasibly be accommodated during a mission, the problem consists of selecting and scheduling a subset of requests yielding the maximal profit, subject to operational constraints.

The present article is concerned with the management of several satellites performing multiple orbits over a given planning horizon. Because a given request can sometimes be satisfied by several satellites in more than one of their orbits, the problem is not separable by satellite or by orbit. Instead, planning must be performed simultaneously for all satellites and orbits considered.

This problem is inspired from the *PLEIADES* constellation of satellites which is being planned by the Centre National d'Études Spatiales (CNES) in France, and due to be launched in 2008. In this system, satellites perform a cycle of orbits around the Earth over a period of several days. Each orbit is phased out with respect to the preceding one and its trajectory is cyclic in the sense that the satellite recovers its initial position after a predefined number of orbits. Furthermore, a full cycle enables the satellite to view each area of the planet. During the course of one particular orbit, a satellite can take several photographs by rotating itself between consecutive shots.

The planning of the *PLEIADES* system first gave rise to the 2003 ROADEF Challenge (Verfaillie et al., 2002a,b) in which teams were required to develop algorithms for the management of a single satellite over a single orbit. Cordeau and Laporte (2005) developed a tabu search heuristic for this simplified problem and won the second prize in the competition. Following the competition, we undertook to generalize this approach to the case of multiple satellites and orbits. This work was initiated in collaboration with the CNES which provided us with a description of planned operating conditions for the *PLEIADES* system as well as more representative benchmark instances.

Problems related to Earth observation by satellites have received only limited attention in the operational research literature. Some authors have studied the management of *SPOT* satellites for which technical constraints restrict the time window of a photograph to a single moment. As a result, scheduling constraints can be interpreted as mutual exclusions. Bensana et al. (1999) have introduced large scale benchmark instances for the uncapacitated and capacitated versions of the problem involving one satellite performing one or several orbits (capacity refers here to the total information that can be recorded in the satellite). Vasquez and Hao (2001, 2003) have presented a tabu search algorithm together with a "logic-constrained" knapsack formulation, as well as some upper bounds for this problem. The management of agile satellites similar to those used in the *PLEIADES* system has also been investigated by Verfaillie and Lemaître (2001) and Lemaître et al. (2002). In particular the latter authors have described dynamic programming, constraint programming and local search methods. Finally, other studies on different variants of the satellite scheduling problem have been performed, among others, by Gabrel and Murat (2003), Harrison et al. (1999), Morris et al. (1997), Pemberton (2000) and Wolfe and Sorensen (2000).

The purpose of this paper is to describe the algorithm we have developed for the *Multiple Satellite and Multiple Orbit Problem* (MSMOP), and to introduce upper bounds based on column generation which can be used to assess the quality of the solutions produced by the heuristic. We also report results on instances provided by the CNES. The remainder of the paper is organized as follows. The next section formally defines the problem and introduces some notation. Section 3 then describes the tabu search heuristic developed for the MSMOP. This is followed by the column generation approach in Section 4, by computational results in Section 5, and by conclusions in Section 6.

2 Problem Description

In the MSMOP, users can submit two types of requests: a *target*, i.e., a circle of limited dimension, or a *polygon* which may cover a wide geographical area. Because of their size, polygons cannot usually be photographed in a single shot and are therefore partitioned into strips of equal width but possibly unequal lengths. For our purposes a target can be seen as a polygon comprising a single strip.

The time required to photograph or *acquire* a strip is proportional to its length. To each strip is also associated a time window during which the acquisition must be performed. In the course of a single orbit, a satellite may be able to photograph several strips of the same polygon by rotating the camera between consecutive shots. Multiple strips associated with the same polygon must, however, be acquired consecutively, both in space and time. Consecutiveness in space means that if multiple strips from the same polygon are acquired, then these must be contiguous. Consecutiveness in time means that between the acquisitions of two strips from a given polygon, the satellite cannot acquire a strip belonging to another polygon. Some requests are *mono* while others are *stereo*. A mono request consists of a single shot of each strip in the polygon. A stereo request consists of two shots of each strip at different angles (and thus within different time windows). A strip from a stereo request is considered to have been acquired only if its *twin strip* has been photographed.

Let R be the set of requests formulated by users and let T be the set of all orbits performed by the satellites within the temporal horizon. To each orbit $t \in T$ is associated a set $R^t \subseteq R$ of requests that can be totally or partially satisfied during the orbit. Of course, the sets R^t are not mutually exclusive. A subset $\overline{R} \subseteq R$ of *priority requests* is also introduced to denote requests that must be fully satisfied in the solution. Let N be the set of all strips, N^t the set of strips associated with requests in R^t and $\overline{N}$, the set of strips originating from priority requests. To each strip $i \in N^t$ are associated an acquisition duration d_i, a profit p_i, and a time window $[a_i, b_i]$. The time window of a strip corresponds to the interval during which the strip lies within the field of view of the satellite for the particular orbit considered. Finally, for each pair of strips $i, j \in N^t$, let c_{ij} be the transition time between the end of strip i and the beginning of strip j.

Because time windows are moderately large, there often exist several feasible orderings with different total transition times for a given subset of strips. The problem thus consists not only of selecting which strips to acquire, but also of determining their sequence and the time at which each of them should be executed. Here we assume that the profit associated with a partially acquired polygon is the fraction of the polygon's surface being acquired, multiplied by the profit associated with the full acquisition of the polygon.

Because the *PLEIADES* system will be co-funded by multiple users, a sophisticated system will be used to share the satellite resources between the different users. This system follows a three-phase allocation process. In phase A, priority requests will be selected through an external procedure not described here. In phase B, requests from a subset of users will then be selected by solving an optimization problem. In this phase, however, a limit will be imposed on the total utilization of the satellites by each user. Finally, phase C will allocate the remaining capacity of the satellites between all users, again through the solution of an optimization problem.

Our algorithm addresses phases B and C of this process. In each of these phases, the objective function considered is the weighted sum of the normalized utilities associated with the different users of the system. The *utility* of a user is defined as the sum of the profits associated with the (possibly partially) satisfied requests of that user. This utility is normalized by dividing it by the maximal utility that could be achieved for this user if it were the only one to use the system. The latter value cannot be known exactly unless the MSMOP is solved to optimality for this user, but it can nevertheless be approximated by means of a heuristic. Let $u_i(s)$ be the utility of user i in solution s and let u_i^* be the maximal utility of user i. The normalized utility is then defined as $u_i'(s) = u_i(s)/u_i^*$. A solution s can thus be characterized by the utility vector $u'(s) = (u_1'(s), u_2'(s), \ldots, u_m'(s))$, where m denotes the number of users in the system. The value of this solution is then given by

$$v(s) = \sum_{i=1}^{m} w_i \tilde{u}_i(s), \qquad (1)$$

where $\tilde{u}(s) = (\tilde{u}_1(s), \tilde{u}_2(s), \ldots, \tilde{u}_m(s))$ is the vector of utilities sorted in increasing order, and the weights w_i are given by

$$w_i = \frac{\alpha^{i-1}}{\sum\limits_{j=0}^{m-1} \alpha^j},$$

with $0 < \alpha \leq 1$.

Equation (1) is in fact called an ordered weighted average (see Yager, 1988) and is used to ensure the fairness of the solution. Because of the way in which the vector of utilities is sorted, this objective function tends to assign higher weights to the users with the smaller utilities. For example, with $m = 4$ users, setting $\alpha = 0.5$ will yield weights of 1/1.875, 0.5/1.875, 0.25/1.875 and 0.125/1.875, with the largest weight being assigned to the user with the smallest utility in the given solution. Setting $\alpha = 1$ will give equal weight to each user whereas values of α close to 0 will give rise to a so-called *lexicographic min-ordering* or *leximin* (Moulin, 1988; Ehrgott, 2000) objective. The leximin objective is similar to the classical maximin objective. In case of equality between the mininum values, however, the leximin will distinguish between solutions by repeatedly considering the next smallest term. For example, the solutions $(1, 1, 2, 3)$ and $(1, 2, 2, 3)$ would be equivalent for the maximin objective while the leximin would favour the second one. For an in-depth discussion of equitable sharing of Earth observation satellite resources, we refer the reader to the work of Bataille et al. (1999) and Lemaître et al. (1999, 2003).

In phase B, the limit imposed on the use of the system by each user is expressed in terms of the total acquisition time (i.e., the sum of the acquisition times of the selected strips). This limit is also imposed when computing the maximal utility u_i^* of user i. To solve a phase B problem, one must therefore proceed in two steps. First, the value u_i^* of each user i must be estimated while imposing an upper bound on the total acquisition time of the selected strips for that user. Second, a multi-user problem must be solved in which the normalized utilities that appear in the objective function (1) are computed with respect to the u_i^* values computed in the first step.

In phase C, no particular constraint is imposed on the use of the system by each user. However, the maximal utility u_i^* is multiplied by a weight q_i (with $1 \leq q_i \leq 100$) when normalizing the utility. In other words, the normalized utility of user i becomes $u'_i(s) = u_i(s)/(u_i^* q_i)$. This approach will tend to favour solutions in which the normalized utilities of the users are proportional to the coefficients q_i. Again, one must proceed in two steps to solve the problem: first estimate the maximum utility u_i^*, and then solve the multi-user problem under the objective function (1).

Finally, in both phases B and C, priority requests must be satisfied in all solutions and will therefore be taken into account both in the evaluation of the maximal utility and in the solution of the multi-user problems.

3 Tabu Search Heuristic

Our tabu search heuristic for the MSMOP is partly based on the method previously developed by Cordeau and Laporte (2005) for the single satellite, single orbit case, but the problem studied in this paper is different in several respects:

1. we now consider several satellites with multiple orbits per satellite;
2. priority requests are now considered as an input;
3. consecutiveness constraints are now imposed on strips from a polygon;
4. the objective function now maximizes a function related to the utility of multiple users;
5. the objective function is now linear with respect to the proportion of the polygon's area being acquired (instead of being piecewise-linear convex);
6. only one direction of acquisition is now considered for each strip instead of two (forward and backward).

The proposed tabu search heuristic explores the solution space by moving at each iteration from the current solution s, defined as sequences of strips to acquire in each orbit, to the best solution in its neighbourhood $M(s)$. Since this rule allows the solution to deteriorate between two successive iterations, we have implemented an anti-cycling mechanism which attributes a tabu status to any solution possessing some attributes of recently visited solutions. An important feature of our algorithm is the possibility of exploring infeasible solutions during the search. In particular, time window constraints are relaxed and their violations are added as penalties to the objective function.

3.1 Initial solution construction

An initial solution is constructed by sequentially inserting all priority requests. When a priority request can be performed in several different orbits, the orbit to which the request is assigned is randomly selected. As a result, this solution may turn out to be infeasible with respect to the time window or utilization constraints, even though the set of priority requests is known to be feasible. Feasibility will then be recovered through iterations of the tabu search heuristic.

3.2 Relaxation mechanism

The value of solution s is defined as $f(s) = v(s) - \beta w(s)$, where $v(s)$ is the objective function value defined by (1), and $w(s)$ is the total time window violations in solution s. The parameter β is initially set equal to 1 and self-adjusts during the course of the search to allow a mix of feasible and infeasible solutions. More precisely, at each iteration the value of β is multiplied by $1 + \delta$, with $\delta \geq 0$, if the current solution is infeasible and divided by $1 + \delta$ otherwise. In our implementation, the value of δ is randomly selected at each iteration in the interval $[0, 1]$.

When satellite utilization constraints are imposed (in phase B), these constraints are also relaxed and their violations are penalized in a similar fashion: a term $\gamma d(s)$ is added to the objective function where $d(s)$ is the total violation and γ is again a self-adjusting parameter.

3.3 Neighbourhood structure

At each iteration, the following six types of moves are considered for all strips to define the neighbourhood $M(s)$ of the current solution s:

1. insert a mono strip i in the solution;
2. remove a mono strip i from the solution;
3. insert twin strips i and j in the solution;
4. remove twin strips i and j from the solution;

5. move a mono strip i from orbit k to orbit l;

6. move twin strips i and j from orbit k to orbit l.

Each of these moves maintains feasibility with respect to all constraints, except for time window and utilization constraints. Recall that if multiple strips from a given polygon are acquired in a solution, then these strips must be contiguous in the polygon and they must be acquired consecutively in the solution. To ensure that these constraints are satisfied at all time, a simple feasibility checking procedure is applied to each potential move, and infeasible moves are discarded. In addition, because priority constraints must be part of the solution, they can never be removed through exchanges of type 2 or 4. They can, however, be moved from one orbit to another through exchanges of type 5 or 6.

When inserting a mono strip in the solution, the position of this strip is chosen so as to maximize the value of $f(s)$, taking into account the increase in the violations of time windows and utilization constraints. This is accomplished by performing a simple insertion, i.e., the ordering of the strips already in the solution remains unchanged. Even though solutions violating time windows are allowed during the search, computation times are reduced by considering the insertion of a strip between two successive strips only if it is possible to satisfy the time windows of all three strips with this sequence. Of course, the time windows of the strips that follow may become violated after performing the insertion.

When handling twin strips, the first strip is inserted in its best possible position and, keeping this position fixed, the second strip is then inserted in its best position. Treating each of the two strips as the first yields two different ways of performing these operations, and both are evaluated to identify the best insertion.

To evaluate the impact of a move, the time variables associated with some of the strips already in the solution must be recomputed in order to measure the total impact on time window violations. Furthermore, the impact of the move on the objective function must be evaluated. Computing this value is non-trivial because the objective function depends on the normalized utilities of all the users, sorted in increasing order. As a result, the correct computation of the impact of a move on the objective function value requires *i)* evaluating the utility of the user affected by the move; *ii)* sorting the normalized utilities to reflect this new value, and *iii)* evaluating the objective function (1).

To prevent cycling, tabu tenures are imposed on solution attributes. Whenever a strip is removed from the solution, its reinsertion is forbidden for θ iterations, where the value of θ is drawn randomly from the interval $[0, \sqrt{|N|}]$ and rounded to the nearest integer. Through an aspiration criterion, the tabu status of a strip can, however, be revoked if that would allow the search process to reach a solution of larger profit than that of the best solution found in which this strip was present.

3.4 Diversification mechanisms

The algorithm uses three diversification mechanisms to ensure a broad exploration of the search space. The first is a continuous diversification scheme now common to several tabu search implementations. Let s denote the current solution. When evaluating the members of $M(s)$, any solution $\bar{s} \in M(s)$ such that $f(\bar{s}) \leq f(s)$ is penalized by a factor proportional to the frequency of its distinguishing strips and a scaling factor. More precisely, let ρ_k be the number of iterations during which strip k has been part of the solution since the start of the search process and let η be the total number of iterations performed. Here, the distinguishing strips refer to those, if any, that would be added to s to obtain $\bar{s}$. Hence, if moving from solution s to solution $\bar{s}$ requires the insertion of strip k in the sequence, then a penalty $q(\bar{s}) = \lambda\, v(\bar{s})\, \rho_k/\eta$ is subtracted from $f(\bar{s})$. The penalty is thus the product of three terms: a control parameter λ, the objective function value $v(\bar{s})$, and the proportion of iterations during which strip k has been part of the solution, ρ_k/η. The presence of the objective function value ensures that the penalties are scaled appropriately with respect to total solution cost. The parameter λ, equal to $\log_{10}(|N|)$ in our implementation, controls the intensity of the diversification. These penalties drive the search process toward less explored regions of the search space whenever a local optimum is reached.

The second diversification mechanism perturbs the solution under certain circumstances. If the best known solution has not improved for 100μ iterations, where μ is the number of times the perturbation mechanism has been applied since the last improvement, then the search stops and restarts from the best known solution s^*. However, solution s^* is perturbed by removing a proportion $\pi = 0.1 \times \min\{\mu, 10\}$ of randomly selected strips. When a strip from a stereo request is selected for removal, its twin strip is also removed. The reinsertion of these strips is then declared tabu for θ iterations. The value of μ is initially set equal to 1 and is reset to 1 whenever a new best solution has been identified.

Finally, the algorithm uses false starts to avoid being trapped in a poor local optimum because of the moves performed in the first few iterations. Specifically, the tabu search procedure is run three times for $100 \ln(|N|)$ iterations, and the best solution identified during these runs is used as the starting solution for the main search.

3.5 Intensification mechanisms

To intensify the search around promising solutions, the algorithm alternates between two modes: global search and orbit search. In global search all six types of exchanges are considered whereas in orbit search, the neighbourhood is limited to the first four types. In the latter case, it is thus impossible to move a strip from one orbit to another. As a result, the problem decomposes by orbit and each one can be optimized independently of the others. The alternation between the two modes depends on the CPU time elapsed since the beginning of the search. If a total of ϕ units of CPU time are allowed for the search, then global search is performed for 0.3ϕ units, followed by 0.2ϕ units of orbit search and, finally, 0.5ϕ units during which the search switches between the two modes at a periodic interval of 0.1ϕ. Finally, false starts are used only the first time global search is performed.

Every 100 iterations, a rescheduling of the strips in the solution is also performed in the hope of improving feasibility. Each strip currently in the solution is removed from its orbit and reinserted in its best position (in the same orbit) in order to minimize $w(s)$. To this end, the strip is temporarily inserted in every possible position in the sequence and the total increase of time window violations that result from the insertion is computed. When a strip can be inserted in several locations with the same effect on $w(s)$, the insertion position is chosen arbitrarily among these locations. Again, these exchanges maintain feasibility of all constraints except time windows.

4 Upper Bounds through Column Generation

We now present a column generation approach that can be used to maximize the utility of a single user, i.e., to determine the value of u_i^* in the phase B and C problems. To this end, we reformulate the problem as a set partitioning model in which each variable (column) represents a set of strips acquired during one orbit of one satellite. Of course, these variables must take the value 0 or 1 in any feasible solution. The column generation algorithm should thus be embedded within a branch-and-bound search to obtain feasible integer solutions. The resulting approach is commonly called *branch-and-price* (see, e.g., Barnhart et al., 1998; Desaulniers et al., 1998). Of course, solving the problem optimally through branch-and-price is likely to be very time consuming for large instances. However, solving only the linear programming relaxation of the set partitioning formulation will provide an upper bound $\dot{u}_i^*$ on the true maximal utility u_i^* of user i. This is the approach we have adopted here.

The formulation that we use is inspired from the unified framework for vehicle routing and scheduling problems described by Desaulniers et al. (1998). We thus decompose the problem into a set partitioning master problem and a set of subproblems, one for each orbit. According to this framework, a strip and an orbit can be seen, respectively, as a *task* and a *commodity.*

For each orbit t, we define a directed graph $G^t = (V^t, A^t)$, where V^t and A^t denote the sets of nodes and arcs, respectively. In the set $V^t = N^t \cup \{o^t, d^t\}$, N^t contains a node for each strip that can be acquired during the orbit and o^t and d^t represent the source and sink nodes for orbit t. Each arc (i, j) is characterized by a profit $p_{ij} = p_i$ and a positive duration $t_{ij} = d_i + c_{ij}$ (with $p_{o^t} = d_{o^t} = 0$).

Several graph reductions are possible. First, all arcs that do not satisfy the feasibility condition $a_i + t_{ij} \leq b_j$ can be removed from the graph since this implies that strip j cannot be acquired after strip i. Because of the way in which two twin strips i and j must be acquired, there is always a single sequence in which the acquisitions can be performed: either i must precede j, or j must precede i. Hence, the following reductions can be performed by considering intermediate nodes. Consider a couple of nodes i and j $(i, j \in N^t)$ representing twin strips and a third node $k \in N^t$. If arcs (i, j) and (i, k) are in A^t but arc (k, j) has been deleted, then one can also delete arc (i, k) since there cannot be a path in G^t containing all three nodes. In the same way if arcs (i, j) and (k, j) exist but arc (i, k) has been deleted, then one can also delete (k,j). Finally if all three arcs (i,j), (i,k) and (k,j) exist but $a_i + t_{ik} + t_{kj} > b_j$, then both (i,k) and (k,j) can be deleted.

For each commodity (orbit) $t \in T$, let Ω^t be the set of feasible paths from o^t to d^t in G^t, and let r^t_ω denote the profit of path $\omega \in \Omega^t$. This profit corresponds to the sum of the profits associated with the individual strips belonging to the path. Let also θ^t_ω be a binary variable taking the value 1 if and only if $\omega \in \Omega^t$ is selected for orbit t in the solution. Finally for each path $\omega \in \Omega^t$, each commodity $t \in T$ and each strip $i \in N$, define a binary parameter $a^t_{i\omega}$ equal to 1 if strip i is covered by path ω.

To handle the twin strip constraints without adding constraints to the problem we introduce for each orbit $t \in T$ an "artificial orbit" $t' \in T'$ characterized by a graph $G^{t'} = (V^{t'}, A^{t'})$. The node set $N^{t'}$ contains a node for each strip that is not related to a priority request and can be acquired during orbit t. The arc set $A^{t'}$ contains the arc $(o^{t'}, d^{t'})$ and a pair of arcs $(o^{t'}, i), (i, d^{t'})$ for each node $i \in N^{t'}$. Finally, if i and j are nodes in $N^{t'}$ representing twin strips, we delete from $A^{t'}$ the arcs $(o^{t'}, j)$ and $(i, d^{t'})$ and introduce the arc (i,j). As a result, either both i and j will be covered in a path from $\Omega^{t'}$, or they will both be covered in a path from Ω^t.

The problem for a given user can then be formulated as follows:

$$\text{maximize} \quad \sum_{t \in T} \sum_{\omega \in \Omega^t} r^t_\omega \theta^t_\omega \tag{2}$$

$$\text{subject to} \quad \sum_{t \in T} \sum_{\omega \in \Omega^t} a^t_{i\omega} \theta^t_\omega = 1 \qquad \forall i \in \overline{N} \tag{3}$$

$$\sum_{t \in T} \sum_{\omega \in \Omega^t} a^t_{i\omega} \theta^t_\omega + \sum_{t \in T'} \sum_{\omega \in \Omega^t} a^t_{i\omega} \theta^t_\omega = 1 \qquad \forall i \in N \setminus \overline{N} \tag{4}$$

$$\sum_{\omega \in \Omega^t} \theta^t_\omega = 1 \qquad \forall t \in T \tag{5}$$

$$\theta^t_\omega \geq 0 \text{ and integer} \qquad \forall \omega \in \Omega^t,\ \forall t \in T \cup T'. \tag{6}$$

Constraints (3) and (4) ensure that each strip from a priority request is acquired and that all strips are acquired at most once. Constraints (5) ensure that a feasible path is assigned to each orbit. The LP relaxation of this formulation can be solved by a column generation algorithm in which positive reduced cost columns are generated by solving a resource constrained shortest path problem.

Let P^t be the set of polygons that can be acquired during orbit t. To handle the polygon constraints, we first change the structure of the graph $G^t = (V^t, A^t)$ as follows. For each polygon $p \in P^t$, the nodes $\{i_1, ..., i_n\}$ associated with strips belonging to p are duplicated by creating nodes $\{i'_1, ..., i'_n\}$ which are linked to the original ones with arcs (i'_k, i_k) for $k = 1, \ldots, n$. If j is a node associated with a strip not belonging to the considered polygon, then arcs of the form (j, i_k) are replaced by arcs (j, i'_k) for $k = 1, \ldots, n$ (i.e., incoming arcs of the nodes $\{i_1, ..., i_n\}$ become incoming arcs of the nodes $\{i'_1, ..., i'_n\}$). Furthermore, all arcs that connect nodes of $\{i_1, \ldots, i_n\}$ corresponding to non-contiguous strips are deleted. Finally, let $e^t_{p\omega}$ be an integer parameter representing the number of times that an arc of the form (j, i_k)

from polygon p is used in path ω. Polygon constraints can be imposed through the following inequalities which are added to the master problem:

$$\sum_{\omega \in \Omega^t} e^t_{p\omega} \theta^t_\omega \leq 1 \qquad \forall p \in P. \tag{7}$$

Finally, the constraint on the total utilization time imposed in phase B instances can be expressed as follows:

$$\sum_{t \in T} \sum_{\omega \in \Omega^t} d^t_\omega \theta^t_\omega \leq L, \tag{8}$$

where L denotes the allowed time and d^t_ω is the sum of the acquisition times of the strips acquired in path ω.

5 Computational Results

To assess the quality of the heuristics, computational experiments were performed on 13 data sets provided by the CNES. Each set considers two satellites performing 12 or 13 orbits in a 24 hour time horizon. Four users, identified by the digits $1, \dots, 4$. are considered in each set.

We first summarize in Table 1 the main characteristics of the instances. This table contains 52 lines and each line provides the statistics for one data set and one user. In this table, $|T|$ is the total number of relevant orbits for the given user, $|R|$ is the number of requests formulated by that user, and $|N|$ is the number of strips in these requests (with $|\overline{R}|$ and $|\overline{N}|$ indicating the number of priority requests and strips, respectively). The remaining three columns indicate the number of twin strips (TS), the number of polygon requests (PR), and the number of strips belonging to these polygon requests (SPR). Instances for users 1 and 2 are the largest ones with an average number of strips exceeding 2000 and 1500, respectively. In these instances, twin strips represent more than 50% of all strips. Polygon requests also constitute an important part of all requests. Instances for users 3 and 4 are much smaller with just over 150 strips, on average.

Since the tabu search heuristic relies on a number of parameters whose values must be set by the user, we have first performed a sensitivity analysis for the most important parameters: λ which controls the intensity of the continuous diversification, and π which controls the fraction of requests which are removed from the solution when it is perturbed. The heuristic is rather insensitive to the values of these parameters but it consistenly yielded good results on all instances when setting $\lambda = \log_{10} |N|$ and $\pi = \min\{\mu, 10\}$, where μ is the number of time the perturbation mechanism has been applied since the last improvement. We have thus used these settings in all remaining experiments.

For each data set, we have then performed a time-constrained optimization (phase B) for users 1 and 2. In this phase, user 1 was allowed 750 seconds of satellite utilization and user

Table 1: Characteristics of the test instances

	$\|T\|$	$\|R\|$	$\|\overline{R}\|$	$\|N\|$	$\|\overline{N}\|$	TS	PR	SPR
16950_1	24	850	25	1788	26	1284	268	905
16950_2	25	688	25	1610	26	1114	244	968
16950_3	4	81	25	110	26	28	15	30
16950_4	2	38	25	46	26	2	3	10
16951_1	24	852	29	1764	31	1272	256	862
16951_2	24	680	29	1555	31	1110	235	912
16951_3	4	84	29	112	31	28	14	28
16951_4	3	42	29	50	31	4	2	8
16952_1	24	826	28	1704	29	1226	247	823
16952_2	25	637	28	1464	29	1060	225	869
16952_3	4	105	28	149	29	38	24	49
16952_4	3	35	28	37	29	2	1	2
16953_1	26	1272	26	2124	27	1188	240	807
16953_2	25	647	26	1483	27	1040	229	886
16953_3	5	107	26	157	27	48	25	51
16953_4	3	47	26	61	27	10	8	17
16954_1	26	1331	26	2230	27	1222	265	887
16954_2	24	678	26	1637	27	1164	254	1024
16954_3	5	106	26	154	27	46	24	49
16954_4	3	50	26	69	27	16	9	20
16955_1	26	1386	25	2298	26	1252	269	894
16955_2	25	712	25	1705	26	1150	262	1065
16955_3	5	109	25	156	26	46	23	47
16955_4	3	54	25	80	26	20	10	26
16956_1	26	1409	25	2314	26	1270	254	845
16956_2	26	705	25	1677	26	1172	258	1026
16956_3	5	98	25	137	26	34	21	43
16956_4	3	61	25	87	26	20	11	27
16957_1	26	1408	25	2314	26	1262	254	856
16957_2	25	695	25	1610	26	1122	245	955
16957_3	4	71	25	96	26	24	13	26
16957_4	3	61	25	89	26	24	10	26
16958_1	26	1398	30	2278	31	1224	244	820
16958_2	25	660	30	1521	31	1116	234	901
16958_3	4	97	30	131	31	32	18	36
16958_4	3	51	30	66	31	10	6	16
16959_1	26	1318	26	2148	27	1166	230	773
16959_2	25	644	26	1472	27	1046	233	882
16959_3	4	107	26	155	27	44	25	51
16959_4	3	50	26	63	27	10	8	16
16960_1	26	1289	27	2175	29	1238	251	848
16960_2	25	655	27	1567	29	1146	242	967
16960_3	5	108	27	161	29	54	25	51
16960_4	2	48	27	67	29	18	8	18
16961_1	26	1339	25	2240	26	1242	262	882
16961_2	24	686	25	1645	26	1132	258	1024
16961_3	5	109	25	158	26	46	25	51
16961_4	3	52	25	75	26	16	10	25
16962_1	25	1396	25	2295	26	1246	259	862
16962_2	24	709	25	1712	26	1176	270	1074
16962_3	5	107	25	151	26	40	23	47
16962_4	3	59	25	85	26	22	10	25

2 was allowed 600 seconds. The tabu search heuristic was first executed for 10 minutes on a 2.53 GHz Pentium 4 processor with the requests of each individual user to estimate the maximal utility u_i^*. It was then run for 60 minutes on the full problem. The results of these experiments are reported in Table 2. We indicate, for each data set, the estimate $\hat{u}_i^*$ of the maximal utility of each user i, the utility $u_i(s)$ of this user in the final solution s, and the normalized utility $u_i'(s)$. For all data sets, the value of α in the objective function is set equal to 0.001. As a result, the value of the objective function is very close to that of the smallest normalized utility. One can see from these results that the utility of a user in the final solution is close to the estimation of the maximal utility $\hat{u}_i^*$: the normalized utilities all exceed 0.92. Furthermore, the difference between the utilities of the two users are very small and often below 1%. We also indicate in Table 2 the value of an upper bound $\dot{u}_i^*$ computed by column generation. This bound was obtained by solving the LP relaxation of model (1)-(8) with a maximum CPU time of 24 hours. For user 1, the solution process was often stopped before reaching optimality. However, the reported values should be close to the true LP values since the rate of improvement was usually very small when the algorithm was stopped. Considering that the value $\dot{u}_i^*$ is an upper bound on the LP relaxation of the problem (which is itself an upper bound on the optimal integer solution value), one may conclude, by comparing the $\dot{u}_i^*$ and $\hat{u}_i^*$ values, that the heuristic properly evaluates the true maximal utility of a user. This result can probably be explained by both the strength of the upper bounds provided by the LP relaxation of the set partitioning formulation, and the quality of the heuristic.

We have then performed the unconstrained optimization (phase C) for all users. Here, the values q_i used in the computation of the normalized utilities are $q_1 = 40$, $q_2 = 40$, $q_3 = 10$ and $q_4 = 10$. In Table 3 we again report the estimate $\hat{u}_i^*$ of the maximum utility computed with the tabu search heuristic. The table also shows the utility $u_i(s)$ of user i in the final solution s as well as the normalized utility $u_i'(s)$. Again, the tabu search heuristic was executed for 10 minutes to estimate the maximal utility of each user and for 60 minutes on the full problem. Recall that the computation of the normalized utility in this phase involves a large multiplier in the denominator, which yields small values. From these results, one can see that for users 3 and 4, the lower and upper bounds on the true maximal utility are very close. For users 1 and 2, the gap is larger but it is nevertheless below 3% for most instances. In this phase, there usually exists a significant difference between the maximal utility of a user and the utility $u_i(s)$ achieved when considering all users concurrently. This is explained by the fact that the maximal utility is computed without imposing any limit on the use of the satellites by a given user. Finally, one can also observe that the normalized utilities of the users with a small utility (and thus a large weight in the objective function) are usually very close. For all instances, the difference between the normalized utilities of users 1 and 2 is at most 0.00003.

6 Conclusions

We have considered the Multiple Satellite and Multiple Orbit Problem, which consists of selecting and scheduling requests with the aim of maximizing the total utility associated with

Table 2: Results on phase B instances

Instance	$\dot{u}_i^*$	$\hat{u}_i^*$	$u_i(s)$	$u_i'(s)$
16950_1	3805223404.2	3767992300	3609293167	0.95788
16950_2	3300098874.6	3211262510	3071674027	0.95653
16951_1	3744713939.5	3705839270	3562169010	0.96123
16951_2	3310306815.1	3264970158	3144754273	0.96318
16952_1	3628348154.8	3607404960	3424899787	0.94941
16952_2	3378020344.1	3315387600	3122056060	0.94169
16953_1	3640794101.2	3607518450	3352385807	0.92928
16953_2	3496010922.8	3437097170	3185268955	0.92673
16954_1	3824463515.6	3785693880	3600660870	0.95112
16954_2	3714752867.9	3659962180	3429036736	0.93691
16955_1	3813543894.8	3775211230	3577474600	0.94762
16955_2	3706782423.7	3629722570	3475516450	0.95752
16956_1	3786328424.2	3763423030	3617069480	0.96111
16956_2	3567789302.6	3463522270	3332601000	0.96220
16957_1	3630315249.6	3571726490	3426302090	0.95928
16957_2	3475485107.3	3367949820	3245502186	0.96364
16958_1	3599806509.2	3528185090	3393748870	0.96190
16958_2	3310679570.2	3244964180	3122502100	0.96226
16959_1	3591576836.1	3545725890	3348409600	0.94435
16959_2	3348676556.0	3273139990	3077852180	0.94034
16960_1	3774322817.8	3705166030	3485119880	0.94061
16960_2	3783371342.7	3670826820	3392853280	0.92428
16961_1	3834729713.3	3777339880	3658476260	0.96853
16961_2	3625005495.4	3538572480	3442668090	0.97290
16962_1	3819718271.3	3818995360	3677728640	0.96301
16962_2	3554436120.6	3465608200	3358266150	0.96903

the selected requests under operational constraints. A tabu search heuristic was developed and upper bounds on the profit were derived by means of a column generation techniques. Tests performed on large scale instances provided by the CNES suggest that the proposed heuristic yields near-optimal solutions.

Acknowledgements

We thank Jean-Michel Lachiver of the French Centre National d'Études Spatiales for his valuable contribution to this study. This work was partially supported by the Canadian Natural Science and Engineering Research Council under grants 227837-04, 6815-04 and OGP0039682. We are also thankful to two anonymous referees.

Table 3: Results on phase C instances

Instance	$\dot{u}_i^*$	$\hat{u}_i^*$	$u_i(s)$	$u'_i(s)$
16950_1	8121434414.40	7921269380	5794265244	0.01829
16950_2	8519725566.33	8307825750	6074941556	0.01828
16950_3	1148040470.00	1143215000	216675200	0.01895
16950_4	460304200.00	460304200	126474500	0.02748
16951_1	8299202107.50	8142567589	5950002479	0.01827
16951_2	8521161500.06	8307970310	6070364420	0.01827
16951_3	1162619900.00	1162619900	236034500	0.02030
16951_4	432417350.00	432417350	83638850	0.01934
16952_1	8120025507.10	8072741767	6083873787	0.01884
16952_2	8334727139.47	8187730922	6176835994	0.01886
16952_3	1450018375.00	1448856900	322637850	0.02227
16952_4	201099900.00	201099900	96330450	0.04790
16953_1	10688895474.30	10517114800	7398039770	0.01759
16953_2	8050833594.00	7941764820	5590262230	0.01760
16953_3	1645170525.00	1638592300	300537800	0.01834
16953_4	454393137.50	453712550	83607000	0.01843
16954_1	11181438393.00	10964766610	7882419570	0.01797
16954_2	8767285963.40	8627522850	6202799590	0.01797
16954_3	1626248245.00	1581862300	296253800	0.01873
16954_4	502219480.00	500463800	124630000	0.02507
16955_1	11227212010.60	10965313340	7892586240	0.01799
16955_2	9307786704.70	9162273460	6598211680	0.01800
16955_3	1583466041.35	1561031550	285777750	0.01831
16955_4	730112125.00	727084100	163303300	0.02246
16956_1	11141678306.70	10913704640	7755558920	0.01777
16956_2	9111973423.27	8952492378	6352532918	0.01774
16956_3	1419425350.00	1419425350	258220700	0.01819
16956_4	741393060.21	737224100	132656300	0.01808
16957_1	11321455323.30	11036306720	7799680740	0.01767
16957_2	8972665281.66	8744696924	6175159114	0.01765
16957_3	918296683.33	916173400	190520750	0.02080
16957_4	706337350.00	706321900	134500800	0.01924
16958_1	11045919087.20	10771721876	7666688497	0.01779
16958_2	8290378662.40	8133205239	5780656980	0.01777
16958_3	1255131200.00	1255131200	235841150	0.01879
16958_4	534143750.00	534143750	160855750	0.02975
16959_1	10785093906.10	10569211897	7322514210	0.01732
16959_2	7695890218.20	7544282880	5227410050	0.01732
16959_3	1569000650.00	1556985000	270817800	0.01739
16959_4	473587600.00	473587600	91669650	0.01936
16960_1	10943618893.70	10785717007	7663478717	0.01776
16960_2	8772225328.02	8632661360	6133195690	0.01776
16960_3	1658119175.00	1645067000	403266000	0.02451
16960_4	339828650.00	339828650	60129500	0.01787
16961_1	11251155019.10	10969531510	7960693080	0.01814
16961_2	8930314251.43	8807377840	6380489300	0.01811
16961_3	1649541270.20	1624832200	306900100	0.01889
16961_4	682521083.33	676261250	122533850	0.01847
16962_1	11160184506.10	10903136180	7814246450	0.01792
16962_2	9228142163.32	9051374719	6485421340	0.01791
16962_3	1576907010.00	1573619000	350635350	0.02228
16962_4	705670500.00	696441700	203346200	0.02935

References

C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh and P.H. Vance. “Branch-and-Price: Column Generation for Solving Integer Programs.” *Operations Research*, **46**:316–329 (1998).

N. Bataille, M. Lemaître and G. Verfaillie. “Efficiency and fairness when sharing the use of a satellite.” In *Proceedings of the 5th International Symposium on Artificial Intelligence, Robotics and Automation in Space*, pages 465–470, Noordwijk (1999).

E. Bensana, M. Lemaître, and G. Verfaillie. “Earth observation satellite management.” *Constraints*, **4**:293–299 (1999).

J.-F. Cordeau and G. Laporte. “Maximizing the Value of an Earth Observation Satellite Orbit.” *Journal of the Operational Research Society*, **56**:962–968 (2005).

G. Desaulniers, J. Desrosiers, I. Ioachim, M.M. Solomon, F. Soumis and D. Villeneuve. “A Unified Framework for Deterministic Time Constrained Vehicle Routing and Crew Scheduling Problems.” In T.G. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, pages 57–93. Kluwer, Norwell, MA, 1998.

M. Ehrgott. *Multicriteria Optimization, Lecture Notes in Economics and Mathematical Systems Volumne 491*. Springer, New York, 2000.

V. Gabrel and C. Murat. “Mathematical Programming for Earth Observation Satellite Mission Planning.” In T. Ciriani, G. Fasano, S. Gliozzi and R. Tadei, editors, *Operations Research in Space and Air*, chapter 7. Kluwer, Boston, 2003.

S.A. Harrison, M.S. Philpott and M.E. Price. “Task Scheduling for Satellite Based Imagery.” In *Proceedings of the 18th Workshop of the UK Planning and Scheduling Special Interest Group*, pages 64–78, University of Salford, UK (1999).

M. Lemaître, G. Verfaillie and N. Bataille. “Exploiting a Common Property Resource under a Fairness Constraint: a Case Study.” In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 206–211, Stockholm (1999).

M. Lemaître, G. Verfaillie, H. Fargier, J. Lang, N. Bataille and J.-M. Lachiver. “Equitable Allocation of Earth Observing Satellites Resources.” In *Proceedings on the 5th ONERA-DLR Aerospace Symposium (ODAS’03)*, Toulouse (2003).

M. Lemaître, G. Verfaillie, F. Jouhaud, J.-M. Lachiver and N. Bataille. “Selecting and scheduling observations of agile satellites.” *Aerospace Science and Technology*, **6**:367–381 (2002).

R.A. Morris, J.L. Bresina and S.M. Rodgers. “Automatic Generation of Heuristics for Scheduling.” In M. Pollack, editor, *Proceedings of the 15th International Joint Conference on Artificial Intelligence*, pages 1260–1266, Nagoya. Morgan Kaufmann (1997).

H. Moulin. *Axioms of Cooperative Decision Making.* Cambridge University Press, Cambridge, MA, 1988.

J. Pemberton. "Towards Scheduling Over-constrained Remote-sensing Satellites." In *Proc. of the 2nd NASA International Workshop on Planning and Scheduling for Space*, pages 84–89, San Francisco (2000).

M. Vasquez and J.-K. Hao. "A "Logic-constrained" knapsack formulation and a tabu algorithm for the daily photograph scheduling of an Earth observation satellite." *Computational Optimization and Applications*, **20**:137–157 (2001).

M. Vasquez and J.-K. Hao. "Upper bounds for the SPOT 5 daily photograph scheduling problem." *Journal of Combinatorial Optimization*, **7**:87–103 (2003).

G. Verfaillie and M. Lemaître. "Selecting and Scheduling Observations for Agile Satellites: Some Lessons from the Constraint Reasoning Community Point of View." In T. Walsh, editor, *Principles and Practice of Constraint Programming (CP-2001)*, pages 670–684, Paphos, Cyprus (2001).

G. Verfaillie, M. Lemaître, N. Bataille and J.-M. Lachiver. "Management of the mission of Earth observation satellites. Challenge description." Technical report, Centre National d'Études Spatiales, France, 2002a.

G. Verfaillie, M. Lemaître, N. Bataille and J.-M. Lachiver. "Management of the mission of Earth observation satellites. Informal description of the global problem." Technical report, Centre National d'Études Spatiales, France, 2002b.

W.J. Wolfe and S.E. Sorensen. "Three scheduling algorithms applied to the Earth observing systems domain." *Management Science*, **46**:148–168 (2000).

R. Yager. "On ordered weighted averaging aggregation operators in multicriteria decision making." *IEEE Transactions on Systems, Man and Cybernetics*, **18**:183–190 (1988).