UNIVERSITÀ DEGLI STUDI DI MILANO
Facoltà di Scienze Matematiche, Fisiche e Naturali
Dottorato di Ricerca in Informatica - XVIII ciclo

Planning and Scheduling Problems for Earth Observation Satellites: Models and Algorithms

Nicola Bianchessi[1]
Dipartimento di Tecnologie dell'Informazione
Via Bramante, 65, I-26013 Crema (CR), Italy

[1]E-mail: bianchessi@dti.unimi.it

Contents

1 Introduction **1**
- 1.1 Global picture of the system . . . 1
- 1.2 Basic problem: informal description . . . 3
- 1.3 Basic problem extensions . . . 7
- 1.4 Literature overview . . . 8
- 1.5 Outline . . . 12

2 The Multi-Orbit Optical Constellation Problem **14**
- 2.1 Problem description . . . 14
- 2.2 The MOOCP with Satellites Sharing . . . 15
- 2.3 Upper bounding procedure . . . 16
  - 2.3.1 Column generation . . . 16
  - 2.3.2 Problem formulation . . . 17
  - 2.3.3 GENCOL . . . 20
- 2.4 A heuristic for the MOOCP with Satellites Sharing . . . 22
- 2.5 Computational results . . . 23
- 2.6 Conclusions . . . 24

3 The Multi-Orbit Radar Constellation Problem **29**
- 3.1 Problem description . . . 29
  - 3.1.1 Setup constraints . . . 30
  - 3.1.2 Splitted requests . . . 31
  - 3.1.3 Memory constraints . . . 31
  - 3.1.4 Operational profile constraints . . . 31
  - 3.1.5 Transmission . . . 32
- 3.2 Algorithms . . . 36
  - 3.2.1 Preprocessing . . . 36
  - 3.2.2 Initialization . . . 36
  - 3.2.3 Main loop . . . 38
  - 3.2.4 Finalization . . . 47
  - 3.2.5 Decision policies . . . 49

3.2.6 Objective functions 49
3.3 Computational results 50
3.3.1 Planning and scheduling scenari 50
3.3.2 Test results 51
3.3.3 High priority requests 54
3.3.4 Failures 54
3.4 Conclusions 54

4 An alternative solution methodology for the MORCP 58
4.1 A relaxed problem 58
4.2 Relaxed problem formulation 59
4.2.1 Commodity network 59
4.2.2 Operational profile resources 62
4.2.3 Memory resources 65
4.2.4 A model 65
4.3 Lagrangean relaxation 67
4.3.1 A dynamic programming algorithm for the SPPRC 68
4.3.2 Solving the Lagrangean dual 71
4.4 Defining quasi-feasible solutions 71
4.5 Computational results 72
4.6 Conclusions 76

5 Conclusions 77

Chapter 1

Introduction

1.1 Global picture of the system

The mission of an Earth Observing Satellite (EOS) is to acquire images of specific areas of the Earth surface, in response to customers' observation requests.

EOS are platforms equipped with instruments for optical, radar or infra-red observation, placed in low orbits around the Earth. The satellite orbit is circular: this induces a constant altitude. To achieve repetitive observations under comparable light conditions, whichever area is observed and whatever the observation day is, the satellite orbit is heliosynchronous (constant angle throughout the year between the orbital plan of the satellite and the Earth-Sun axis; figure 1.1). This is useful for observation with optical instruments. A heliosynchronous orbit is nearly a polar orbit: the orbital plan passes almost through the North and South poles of the Earth. The fact that the orbit is heliosynchronous, together with the rotational movement of the Earth from West to East around the polar axis, allows a potential covering of the whole Earth surface (figure 1.2). Finally, the orbit is phased: after a cycle of a fixed number of revolutions the satellite goes back exactly to its previous positions with respect to the Earth.

In the general scenario, as considered in this work, a constellation of EOS is available. Thus, the scientific activities of satellites have to be coordinated in order to fully exploit the capacities of the system.

From the viewpoint of the scientific activities, satellites are operated by an Image Programming and Processing Center. This center receives observation requests from customers (scientists and users) and distributors and builds the daily imaging workload of each satellite. Then it receives back the data associated with acquired images, processes and evaluates them. Finally it sends the results back

[1]This figure is taken from (Verfaillie et al., 2002b).

[3]This figure is taken from (Verfaillie et al., 2002b).

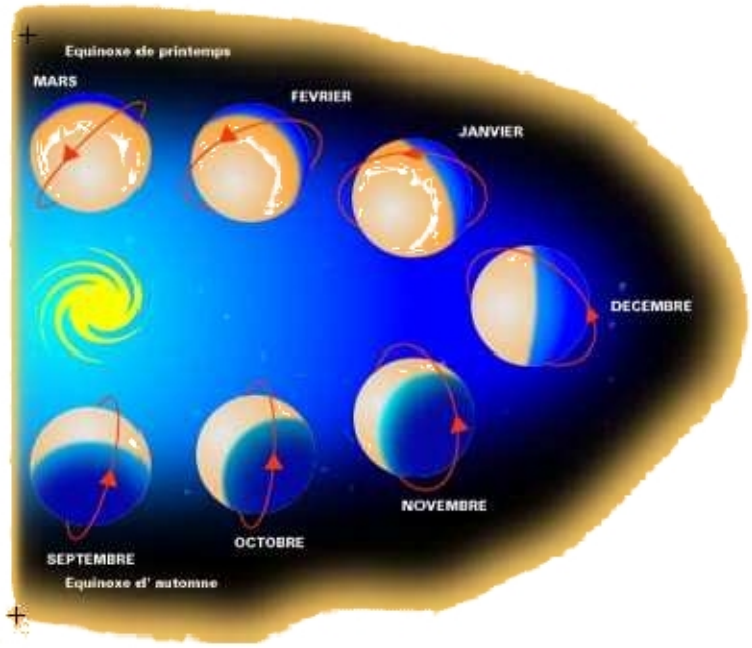


Figure 1.1: the movement of the orbital plan month after month [1].

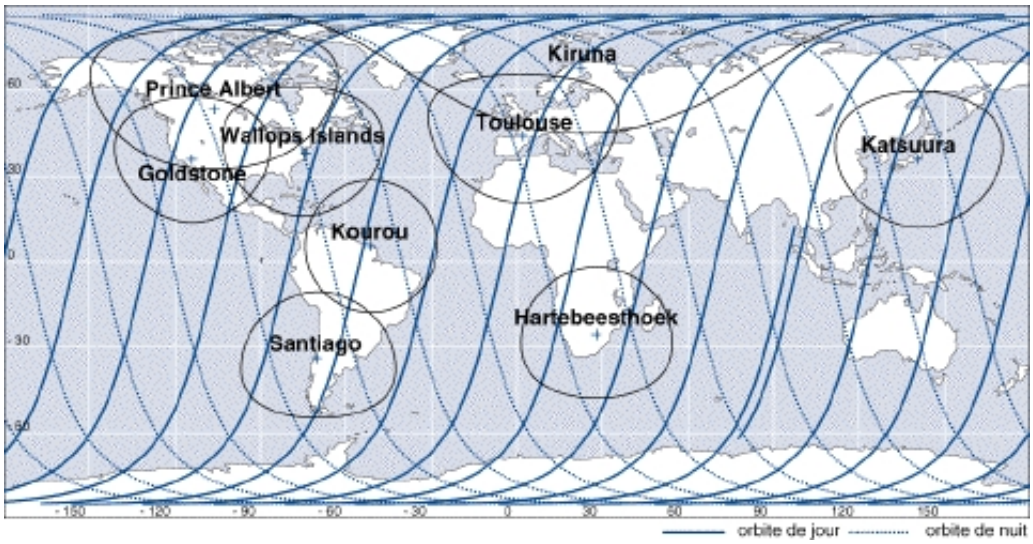


Figure 1.2: the daily track of a satellite on the Earth surface. The labelled points represent cities where ground stations are located; they appear together with the corresponding visibility window available for ground station-satellite connections[3].

to the customers.

In the remainder we will only consider the management of scientific activities of EOS satellites, disregarding all the other aspects concerning the global management of the overall system, such as flight dynamics.

1.2 Basic problem: informal description

A customer *request* generally involves several contiguous images (this will be explained below in more details). Due to the large number of requests concerning some zones, in general all of these requests cannot be satisfied on a given planning horizon. Accordingly, the overall problem consists of selecting a feasible sequence of images that will be acquired by the satellite constellation over the planning horizon, with the objective of providing maximum satisfaction of the customer requests.

The set of customers' observation requests (the input data of the planning problem) evolves almost continuously: it includes new incoming requests and eliminates those which have been satisfied or which are out of date. Each request is associated with the following data:

- a geometrical description of the area to be imaged on the Earth surface;
- a set of angular constraints concerning the satellite setup during the acquisition;
- a validity period which defines the utility of the request (usually given in days).

The Earth surface associated with a request can be an area of limited dimensions, or a polygon which may cover a wide geographical area. Because of their size, polygons cannot usually be acquired in a single shot and are therefore partitioned into contiguous strips (or "swaths") of rectangular shape (i.e. a request is splitted into a set of *images*). The width of a strip depends on the design of the instrument. The length of a strip may vary up to a given limit depending on the hardware characteristics of the satellite (figure 1.3). For our purpose an area of limited dimensions can be seen as a polygon comprising a single strip.

For each satellite-image pair we can compute a set of available time windows in which the satellite can start the acquisition by using the satellite's orbital parameters together with the angular constraints and the validity period associated with the image. To this purpose we can distinguish at least two cases according to satellite features. If the satellite is able to observe the desired Earth surface

[5]This figure is taken from (Lemaître et al., 2002).

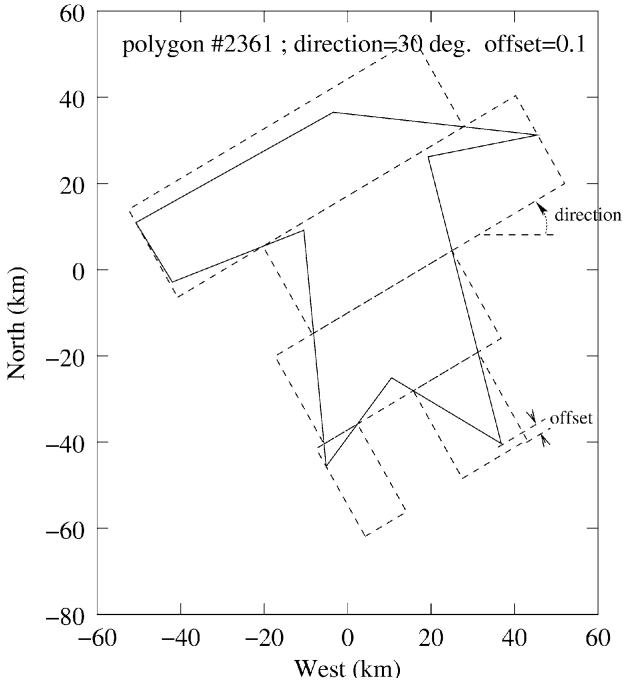


Figure 1.3: the cutting up of a polygonal area into contiguous parallel strips of constant width[5].

before or after being exactly at its zenith (figure 1.4), time windows are continuous intervals of possible starting times; otherwise each time window reduces to a single starting time.

The time needed to observe or *acquire* a strip is proportional to its length. On the other hand, the setup change requires that a minimum transition time elapses between two consecutive acquisitions.

Since tipically the number of requests exceeds what can feasibly be accomodated during a mission, the problem consists in selecting the subset of images to be taken by satellites during the planning horizon in order to satisfy a maximal part of the set, provided that:

- each satellite takes at most one image at a time;
- each satellite starts the acquisition of an image in one of the time windows associated with the satellite-image pair;
- sufficient transition time elapses between two consecutive acquisitions performed by a satellite.

The difficulty of this planning problem increases taking into account additional constraints, but it also depends on satellite capacities. For example, the *SPOT* satellites, a current generation of optical EOS, are characterized only by one degree of freedom, thanks to a mirror that can be moved on the roll axis, placed in front of each observation instrument. This mirror can be moved only during transitions between images, but it is fixed during an image acquisition. Therefore all images are acquired with the same azimuth (the one parallel to the satellite track) thanks to the movement of the satellite on its track. Moreover the starting time of any candidate image is fixed (it is the exact time at which the satellite flies over the beginning of the area to be acquired). As a consequence the order of image acquisitions cannot change, and compatibilities among acquisitions can be pre-computed. On the other hand, the new generation of EOS, like those studied in the French *PLEIADES* project, are Agile Earth Observing Satellites (AEOS). They have the advantage to be more maneuverable: the unique on-board optical observing instrument is fixed on the satellite but the whole satellite can move on the three axes (roll, pitch and yaw). In this way, maneuverability for image acquisitions as well as for transitions between consecutive acquisitions is allowed. New agility capabilities let azimuth and the starting time of an image acquisition be now free, within given limits (figure 1.4 and figure 1.5). Thus the satellite can image a given area on the Earth surface in a potentially infinite number of ways. This leads to a potential better efficiency of the whole system exploitation, while increasing the problem difficulty.

[6]This figure is taken from (Lemaître et al., 2002).

[7]This figure is taken from (Lemaître et al., 2002).

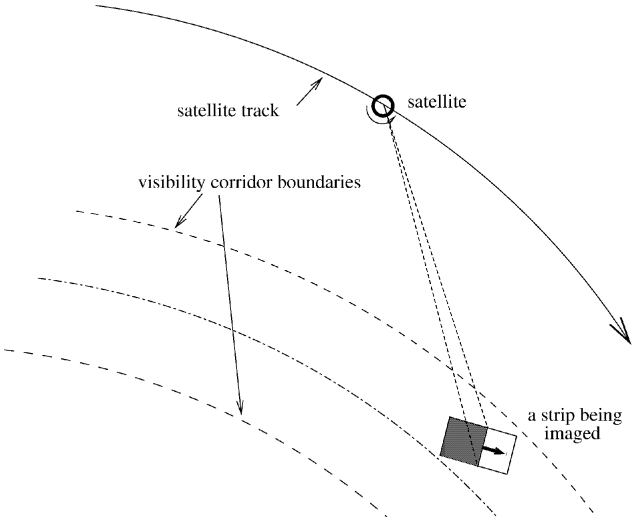


Figure 1.4: an image acquistion performed by an AEOS [6].

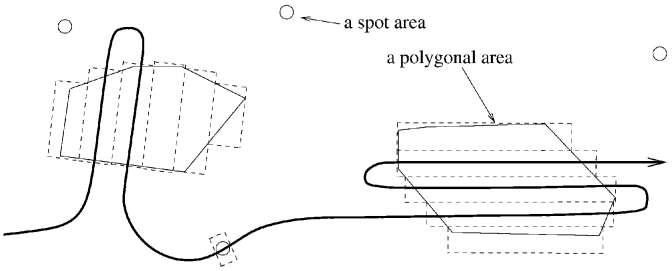


Figure 1.5: a possible sequence of image acquistions for an AEOS [7].

In a similar way, also a greater number of satellites and a longer planning horizon may increase the number of opportunities to observe a specific area, and this may result again into an increase of the problem difficulty.

1.3 Basic problem extensions

As already mentioned, the goal of Earth Observation Satellites missions consists in maximizing the requested images allocation on the payload. The mathematical formalization of the physical problem considers the sum of the acquired images as the objective function of the problem to be solved; a different approach consists in labelling each image to be acquired with a non-negative weight. The criterion to maximize becomes the sum of the weights of the acquired images (standard utility criterion). It must be noticed that if images are associated with weights, a further constraint must be considered to acquire each image at most once.

The introduction of non-negative weights helps us to evaluate requests partially covered by satisfactory images. A simple policy is to choose a reward proportional to the acquired surface for each request. The corresponding criterion is named the *linear quality criterion*. The drawback of this policy is the fact that many requests can be covered only partially. The Programming Center would like to favour the ending of acquisition of the almost acquired requests, before beginning new ones. This preference constraint can be taken into account by a modified quality criterion: to maximize the sum of *partial rewards* obtained on each request. The partial reward obtained on a request is a convex function of the acquired surface, instead of a linear one. This criterion is named the *non-linear quality criterion*.

Moreover weights are useful to correct the negative effects of a short term planning horizon, as well as the effects of meteorological uncertainties (in case of satellites equipped with optical instruments). Images with the smallest number of remaining feasible opportunities and the highest realization likehood (good meteorological forecast for the next planning horizon) can be favoured by modifying their original weight.

As far as constraints are concerned, those considered in more realistic version of the problem are the following:

- a request can be *mono* or *stereo*: a mono request consists of a single shot (acquisition) of each strip in the polygon; a stereo request consists of two shots for 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 also been acquired (*stereoscopic constraints*); depending on the satellite characteristics, the two twin strips have to be acquired either by the same satellite during the same over-flight or by two different satellites;

- multiple images associated with the same request must be acquired by satisfying specific criteria depending on the features of the considered satellites (in the following chapters, these constraints will be explained in details for two specific problems);
- there can be a subset of *high priority* requests that are known to be all satisfiable and must be satisfied;
- acquired images are stored on board until they can be downloaded towards a ground station: the recording capacity on board cannot be exceeded (*memory constraints*);
- the connection between a satellite and a ground station (for downloads) can take place only within specific time windows;
- in every periodical time window of a given width, the overall energy consumed by a satellite cannot exceed a certain threshold (*energy constraints*);
- if some satellites are equipped with several cameras on board, there can be some specific constraints which link their usage.

It must be noticed that whenever specific constraints manage the acquisition of multiple images associated with the same request, the usage of the non-linear quality criterion to evaluate the goodness of the solution can be substituted for the linear one.

1.4 Literature overview

One of the first work related to Earth Observation Satellites has been carried out by Hall and Magazine (1994). In their work they present a problem, named the Space Mission problem (SM), consisting in selecting and scheduling a set of jobs on a single machine, among a set of candidate jobs. Each candidate job is associated with a fixed duration, a given time window and a weight. The aim is to select a feasible sequence of jobs maximizing the sum of weights. The SM is NP-hard in the strong sense, since it is a generalization of the problem 'Sequencing with release times and deadlines' (Garey and Johnson, 1979). The authors develope some heuristic methods and two upper bounding procedures, based upon a preemptive relaxation of the problem (a job can be fragmented), and upon the use of Lagrangean relaxation. The heuristics and bounding procedures have been incorporated into a dynamic programming algorithm tested on 30 randomly generated instances with up to 200 candidates; instances with about one hundred jobs have been solved to optimality.

Some authors have studied the management of *SPOT5* satellites requiring a fixed starting time for any candidate image due to technical reasons. As a result, even if each satellite is equipped with 3 cameras, scheduling constraints can be interpreted as mutual exclusions. Each image is associated with a weight and there is a subset of stereoscopic images. The problem is to find a feasible subset of image satisfying the standard utility criterion. This optimization problem is NP-hard, and can be formulated in the general Valued Constraint Satisfaction Problem model (Schiex et al., 1995). 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 (the term "capacity" refers here to the total information that can be recorded in the satellite). Bensana et al. (1996) and Verfaillie et al. (1996) describe dedicated exact and approximate methods to solve benchmark instances, but they find optimal solutions only for few of them. The column generation technique has been used in (Gabrel and Murat , 2003) to compute upper bound on the benchmark instances. Vasquez and Hao (2001, 2003) presented a tabu search algorithm together with a "logic-constrained" knapsack formulation: the authors obtained the best solutions and good upper bounds for almost all benchmark instances.

Gabrel et al. (1997) presented a work concerning a problem of selection and scheduling for a kind of semi-agile satellite: the satellite is weakly mobile on two axes (pitch and roll), but remains fixed during an image acquisition; so there is only one possible azimuth for an acquisition (parallel to the satellite motion). The maximization criterion is the number of selected images (images are not weighted). In (Gabrel et al., 1997) two related problems have been stated. In the first one, named the Maximum Shot Sequencing Problem (MSP), multiple orbits are considered, so several possible disjoint time windows (continuous intervals) are given for each image. In the second problem, named the Maximum Shot Orbit Sequencing Problem (MSOP), only one orbit is processed, thus a single time window is associated with each candidate image. MSOP and MSP are NP-hard, actually the Shortest Path Problem with Time Windows (SPPTW) arises as a subproblem in both of them. The authors propose exact and approximate algorithms for both the discretized and the continuous time model. They have done simulations and tests on a set of randomly generated instances, assessing the proposed algorithms against upper bounds, and measuring the impact of the discretization. The results show the good quality of the proposed approximate algorithms.

As far as AEOS are concerned, one of the most studied problems is inspired by the *PLEIADES* constellation of optical satellites which is being planned by the Centre National d'Études Spatiales (CNES) in France, and due to be launched in 2008. The first version of the problem 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. In this version of the problem (and in the subsequents to our knowledge), the cutting up of areas in strips is pre-computed, using a single direction (azimuth) for all requests of the same *track* (where the track is defined to be the enlightened half-revolution of the satellite). A constant direction is chosen, as to vary the azimuth between two images is very expensive in terms of transition times. Therefore optimizing the cutting up of each polygon separately (in order to obtain for example the less possible number of strips for each of them) is likely to increase the duration of transition maneuvers and, as a consequence, to decrease the available productive time for proper imaging. An important restriction to the satellite capabilities seems to be a single and constant direction for the cutting up of areas in strips, because in this way its agility is not completely exploited. Actually, the satellite agility is still used (thus making a difference with non-agile satellites): the cutting up direction is maintained constant for a given track, but it can be freely chosen and can change between two consecutive tracks (or even between two groups of images of the same track separated by a large gap); once the direction is fixed, two opposite azimuths can still be used for imaging (*forward view* and *backward view*). As an important side effect, if the cutting up direction is constant, a better approximation of the minimum transition times between each pair of candidate images of the processed track is possible. Actually, due to the difficulty of the real computation, transition times are usually approximately calculated, and if a produced schedule does not pass the final check, they are majored and the schedule is computed again.
In conclusion, in this problem the satellite can acquire a strip with two possible azimuths, and for each *view* it can start the acquisition within the related continuous time window. Some requests are mono while others are stereo (therefore stereoscopic constraints have to be respected), requests are weighted and the non-linear quality criterion has to be maximized. Kuipers (2003), Cordeau and Laporte (2005) presented two different local search algorithms for this version of the problem; they obtained the best solutions for the benchmark instances. For these algorithms they have won respectively the first and the second prize of the challenge.
Lemaître et al. (2002) deal with a version of the problem concerning one satellite performing multiple orbits (tracks) over a scheduling horizon of 1 day. The authors have given the problem the name of AEOS Track selection and Scheduling Problem, because they simplified the problem by optimizing one track at a time. This simplification leads to sub-optimal solutions. Actually most of the images have several other opportunities to be taken after those in the current track. To mitigate this negative effect, the authors suggest to work on the weight associated with the images. In particular they refer to the work of Verfaillie et al. (1999), where for each image the original weight, the remaining opportunities and

the weather forecast are combined into a working weight. To solve the simplified problem, the authors develope four methods: a greedy algorithm, a dynamic programming algorithm, a method based on the existing constraints programming framework and a local search algorithm. This latter is based on neighbourhoods defined considering the possibility to insert or remove images into or from a sequence of image acquistions. They compare the performaces of the methods on six representative instances involving up to 375 requests, giving evidence of the advantages and drawbacks of each method. In particular only the last two cited methods have been able to solve all the instances satisfying all the operational constraints.

Harrison et al. (1999) presented a scheduling problem involving a satellite equipped with a radar instrument. This satellite has the advantage to be very agile on the pitch axis thanks to an electronic scan but the disadvantage to be slow on the roll axis. Since only one azimuth is available for acquiring images, this problem is equivalent to the SM problem (Hall and Magazine, 1994) with transition times. The authors describe a partial enumeration algorithm and give preliminarly results for randomly generated instances, concerning the schedule of a single satellite with up to 50 requests, in a time window of a few minutes. A richer model is considered by Frank et al. (2001), who develope a greedy stochastic algorithm, without reporting about computational results obtained.

In the context of the NASA's Earth observing System domain, the work reported in (Wolfe and Sorensen, 2000) is about the so-called Window-Constrained Packing problem (WPC). Unlike the above cited problems, in this one the observations have a minimum and a maximum duration, and preference is given to higher priority observations, with longer durations, and best placed inside their time-window. There are no transition times among acquisitions. This problem is similar to the SM problem (Hall and Magazine, 1994), with a particular quality function, depending on the starting times of observations. The authors presented two simple construction heuristics as well as a genetic algorithm to schedule acquisition on a single satellite, and reported computational results on randomly generated instances involving up to 50 requests.

Other studies on similar problems have also been performed, among others, by Bataille et al. (1999), Morris et al. (1997) and Globus et al. (2003). In particular in (Lemaître et al., 1999) a new problem concerning the exploitation of a satellite by several entities is taken into account. The aim is both to give maximal satisfaction to each entity and to enforce a kind of fairness constraint on selections.

1.5 Outline

This thesis is focused on the application of Operations Research (OR) tecniques to problems arising for AEOS. In the OR literature related to the management of scientific activities for AEOS, most of the works consider a single satellite performing one or more orbits. The aim of this thesis is to provide insights to build computational instruments applicable in more realistic scenari.

The next chapter deals with a new version of the problem concerning the management of *PLEIADES* constellation of optical satellites. The version under study is more realistic with respect to those considered in the ROADEF challenge and by Lemaître et al. (2002); it differs from the previous ones in what follows:

- we will consider two satellites performing multiple orbits over a given planning horizon of 1 day;
- there is a subset of high priority requests;
- specific constraints are imposed on the acquisitions of multiple images derived from the same requests;
- the objective function is linear with respect to the proportion of the polygon area being acquired (instead of being piecewise-linear convex);
- only one azimuth is available for imaging (the forward one).

Since 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.

For a similar problem version, Bianchessi et. al (2005) defined a tabu search heuristic that can be applied also to our problem. Then we will introduce an upper bounding procedure based on column generation with the aim to find tight dual bounds. We will report results about extensive computations done on instances provided by the CNES that show that the approach adopted is successful.

In the subsequent two chapters we will consider the problem arising as part of the Italian *COSMO-SkyMed* project. The *COSMO-SkyMed* constellation is made of four satellites equipped with SAR (Synthetic Aperture Radar) instruments. The optimization problem concerns the management of these last four satellites performing multiple orbits in a planning horizon of 1, 4 or 16 days. Owing to the SAR technology, there is no difference between day and night observations and the full orbit (not a track) is available for acquisitions. As previously said, for this kind of satellites, only one azimuth is available for acquiring images. In particular each time window associated with a swath reduces to a single starting time, and it is called Data Take Opportunity. Requests are weighted and some of them can have

high priority. Moreover, memory and energy constraints as well as transmission features are considered. The default objective is to maximize the linear quality criterion, but we will consider also the one calling for the minimization of the time necessary to satisfy high priority requests. In the third chapter we will present greedy constructive (randomized) algorithms to solve the problem in its real version. Whereas in the fourth one we will describe a solution methodology that allows us to compute dual bounds and quasi-feasible solutions that can guide the greedy algorithms in finding better feasible solutions than those computed from scratch.

Finally, based on the results obtained, some considerations will be drawn about the applicability of the presented tecniques.

Chapter 2

The Multi-Orbit Optical Constellation Problem

This chapter is concerned with the management of the scientific activities for the *PLEIADES* constellation of optical satellites performing multiple orbits over a given planning horizon. We call this problem the *Multi-Orbit Optical Constellation Problem* (MOOCP). The chapter describes an upper bounding procedure based on column generation that can be used to evaluate the quality of heuristic solutions. In the first section we formally defines the problem introducing some notation. Then in section 2.2 we discuss a variant of the MOOCP arising when it is necessary to share the satellite resources among different users. Section 2.3 illustrates a column generation approach that can be used either to maximize the MOOCP's objective or, as far as sharing is concerned, to maximize the objective associated with a given user. In section 2.4 we shortly describe a heuristic presented by Bianchessi et. al (2005) for the MOOCP with Satellites Sharing and in section 2.5 we use the results produced by this heuristic to assess the strength of our formulations. Finally section 2.6 reports some conclusions.

2.1 Problem description

The MOOCP is an extension of the generic problem presented in section 1.2. In particular here the time windows are continuous intervals of possible starting times (the reference scenario is described in (Verfaillie et al., 2002a)). In the MOOCP we do not consider memory and energy constraints, the additional features taken into account are the following ones. Multiple strips associated with the same polygon must 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*, moreover some of them can be high priority. 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. The objective is to maximize the linear-quality criterion.

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 *high 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. An acquisition duration d_i, a profit p_i, and a time window $[a_i, b_i]$ are associated with each strip $i \in N^t$. 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. This transition time results from the need to move the satellite at the appropriate angle to acquire strip j after strip i. As time windows are moderately large, there often exist several feasible orderings with different total transition times for a given subset of strips (a subset associated either with the same request or with different requests).

2.2 The MOOCP with Satellites Sharing

To share the satellite resources among different users, the *PLEIADES* system will use a three-phase allocation process. In phase A, priority requests will be selected through an external procedure, while in phase B, requests from a subset of users will be selected by an optimization algorithm. In this phase, a limit will be imposed on the total utilization of satellites by each user. This limit is expressed in terms of the total acquisition time (i.e., the sum of the acquisition times of the selected strips). Finally, phase C will allocate the remaining capacity of the satellites between all users. In this phase, no particular constraints will be imposed. In both phases B and C, priority requests must be satisfied in all solutions and will therefore be taken into account in the evaluation of the maximal utility.

This additional feature is modelled considering a new objective: the maximization of 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 he were the only one to use the system (the time limit imposed on the total utilization of the satellites by each user must be considered also in the computation of maximal utility, as well as the additional profit due to the presence of priority requests). The latter value cannot be known exactly unless the MOOCP is solved to optimality for this user, but it can nevertheless be either approximated by means of a heuristic or bounded by solving a relaxation. 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

$$v(s) = \sum_{i=1}^{m} w_i \tilde{u}_i(s), \tag{2.1}$$

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

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

with $0 < \alpha \leq 1$.

Because of the way in which the vector of utilities is sorted, this objective function always assigns a higher weight to the users with the smallest utility. This approach is used to ensure the fairness of the solution.

2.3 Upper bounding procedure

We are now presenting a column generation approach that can be used either to maximize the MOOCP's objective or, as far as sharing is concerned, to maximize the utility of a single user (i.e. to determine the value of u_i^* in phase B and phace C problems).

2.3.1 Column generation

Column generation is one of the most successful approaches in large scale optimization. It consists in solving huge linear programs by considering at once only a small subset of the variables. Let us consider the linear program P, also called the *master problem*, and its dual D, where A is an $m \times n$ matrix, $\mathbf{c}, \mathbf{x} \in \mathbb{R}^n$ and $\mathbf{b}, \pi \in \mathbb{R}^m$:

$$v(P) = min\ \mathbf{c}^T\mathbf{x} \qquad v(D) = max\ \mathbf{b}^T\pi$$
$$A\mathbf{x} = \mathbf{b} \qquad A^T\pi \leq \mathbf{c}$$
$$\mathbf{x} \geq \mathbf{0}$$

At any iteration i of the column generation solution process, only a subset A_i of the columns of A is handled. By solving a restricted master problem, we obtain a primal feasible solution $\mathbf{x}_i$ along with a vector π_i of dual multipliers. With these multipliers, we compute the minimum reduced cost over all columns of matrix A to generate some negative reduced cost columns, if any, that are added to the restricted master problem. If no such column exists, $(\mathbf{x}_i, \pi_i)$ is a pair of primal and dual optimal solutions for P and D, respectively. Computing the minimum reduced cost should be performed in such a way that it is much less costly than evaluating the reduced cost of all columns explicitly. This is the case when the minimum reduced cost is given by the solution of a well structured problem, usually called the *oracle* or subproblem. When columns of A are given as elements of a set $\mathcal{A}$, and the respective cost coefficients can be computed via a function $c : \mathcal{A} \rightarrow \mathcal{Q}$, then the subproblem can be defined as $min\{c(\mathbf{a}) - \pi_i^T\mathbf{a} \mid \mathbf{a} \in \mathcal{A}\}$.

2.3.2 Problem formulation

In order to apply the column generation approach, we have decomposed the MOOCP into a set-partitioning master problem and a set of Elementary Shortest Path Problem with Time Windows (ESPPTW) as subproblems. An ESPPTW is defined for each orbit. The formulation adopted is inspired from the unified framework for vehicle routing and scheduling problems described by Desaulniers et al. (1998). 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 (i.e. stereoscopic constraints) without adding constraints to the problem for each orbit $t \in T$ we introduce 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 the arcs $(o^{t'}, j)$ and $(i, d^{t'})$ from $A^{t'}$ 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 MOOCP can then be formulated as follows:

$$\text{maximize} \quad \sum_{t \in T} \sum_{\omega \in \Omega^t} r^t_\omega \theta^t_\omega \tag{2.2}$$

$$\text{subject to} \quad \sum_{t \in T} \sum_{\omega \in \Omega^t} a^t_{i\omega} \theta^t_\omega = 1 \qquad \forall i \in \overline{N} \tag{2.3}$$

$$\sum_{t \in T} \sum_{\omega \in \Omega^t} a^t_{i\omega} \theta^t_\omega + \sum_{t \in T'} \sum_{\omega \in \Omega^t} a^t_{i\omega} \theta^t_\omega = 1 \qquad \forall i \in N \setminus \overline{N} \tag{2.4}$$

$$\sum_{\omega \in \Omega^t} \theta^t_\omega = 1 \qquad \forall t \in T \tag{2.5}$$

$$\theta^t_\omega \geq 0 \text{ and integer} \qquad \forall \omega \in \Omega^t,\ \forall t \in T \cup T'. \tag{2.6}$$

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.

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

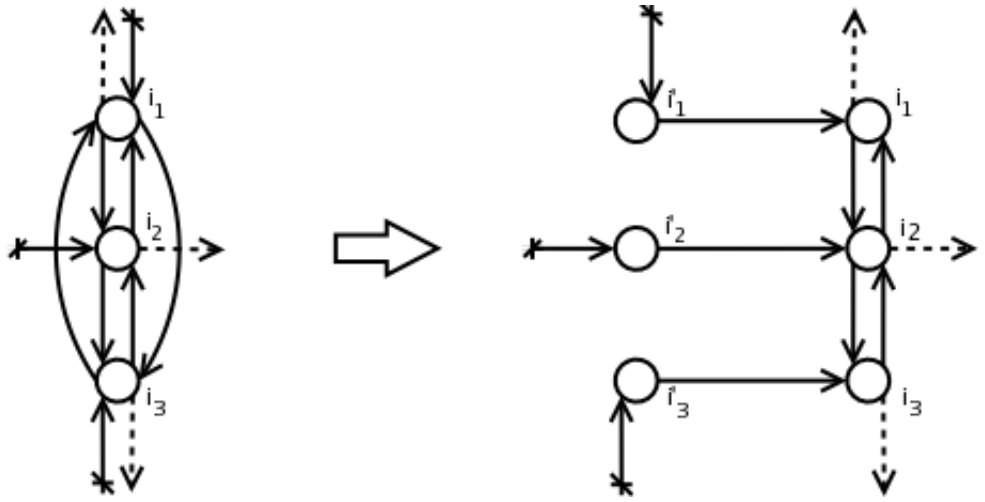


Figure 2.1: change in the structure of graph G^t concerning the nodes $\{i_1, i_2, i_3\}$ associated with strips belonging to a polygon $p \in P^t$; in particular nodes i_1 and i_3 correspond to non-contiguous strips.

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 (figure 2.1). 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:

$$\sum_{\omega \in \Omega^t} e^t_{p\omega} \theta^t_\omega \leq 1 \qquad \forall p \in P. \tag{2.7}$$

The formulation (2.2)-(2.7) is also valid, in case of a single user, for the MOOCP with Satellites Sharing (phase C problem). Moreover, for a given user, the constraints on the total utilization time imposed in phase B problem can be expressed as follows:

$$\sum_{t \in T} \sum_{\omega \in \Omega^t} d^t_\omega \theta^t_\omega \leq L, \tag{2.8}$$

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

The Linear Programming relaxation (LP relaxation) of formulations (2.2)-(2.7) and (2.2)-(2.8) can be solved by a column generation algorithm in which positive reduced cost columns are generated by solving an ESPPTW.

2.3.3 GENCOL

We maximize the LP relaxations of models (2.2)-(2.7) and (2.2)-(2.8) using GENCOL. This commercial solver has been developed in 1981 by Jacques Desrosiers and Franois Soumis while attempting to optimally solve the problem of assigning buses to school routes with a certain flexibility on the arrival and departure times of the buses at the schools. Since its conception, GENCOL has evolved continually through the search of many graduate students and computer analysts. This solver is specially designed to solve, through column generation, general set-partitioning problems where each variable in the master problem is associated with a solution of the (Elementary) Shortest Path Problem with Resource Constraints – (E)SPPRC. In particular, the objective of the set-partitioning problems solved in GENCOL consists in finding a minimum cost partition of the tasks.

GENCOL solves the restricted master problem using either the primal or dual simplex algorithm, whereas each subproblem is solved using the label setting algorithm described in (Desrosier et al., 1995). This algorithm not only finds the path associated with the minimum reduced cost path variable, but also other feasible paths associated with negative reduced cost path variables that may be added to the restricted master problem. Each time the restricted master problem is solved for a subset of path variables, the dual variables are updated and used to modify the cost of the arcs in the subproblem graphs. Once a feasible path is found by a subproblem, a variable is associated with this path in the restricted master problem and the coefficients of the corresponding column are generated according to the contributions of the nodes and arcs found along this path. The iterative process begins by solving a restricted master problem composed of artificial variables, and ends when no paths of negative reduced cost are generated by the subproblems.

The set-partitioning constraints impose that each task is covered exactly once in the solution. Therefore, two paths of the optimal solution will not contain a same task. This simple observation has led to the implementation of a *disjoint column* strategy in GENCOL. This strategy aims at choosing from the feasible path variables generated by the subproblem algorithm the ones that are associated with paths covering different tasks and adding them to the restricted master problem at each iteration. For each subproblem solved at a given iteration, the generated columns are grouped into disjoint blocks. Each disjoint block is formed of disjoint columns, i.e., columns associated with paths covering different tasks. Therefore, a given task is included only once in the set of paths of a disjoint block. When several subproblems need to be solved, it is possible to temporarily eliminate from the graphs tasks covered by paths that were generated by previous subproblems at the current iteration, namely those covered by the paths associated with the columns of the first disjoint block. In this way, the subsequent subproblems cannot

generate paths covering these tasks. These blocks can also be used to establish a priority between generated columns in order to reduce the number of path variables added to the restricted master problem at an iteration.

The search for a lower bound generally requires a significant number of iterations involving the solution of the restricted master problem and the solution of the subproblems. Several parameters in GENCOL control different aspects of this iterative process; a particular parameters setting is called "model" and by breach of terminology, the linear relaxation solved using a given model is also referred to as a model. GENCOL allows to modify the model after a certain number of iterations and this mechanism may accelerate the solution process. A series of less constrained or approximate models can first be solved in order to approach the optimal solution quickly. The most complex model defined by the original problem can finally be solved to complete the optimal solution process.

GENCOL can also handle integrality constraints imposed on path variables. Thus GENCOL is more than a column generation algorithm, it is a branch-and-price algorithm. The column generation scheme is embedded in a branch-and-bound algorithm and if the optimal solution of the linear relaxation does not satisfy integrality constraints, these requirements are enforced by taking branching decisions on variables representing the flow on the arcs of the graph (note that no optimal branching strategy on path variables is available). The column generation process is then applied at each branch node to determine a lower bound at that node.

Several branching strategies have been implemented and stored in GENCOL. The solution process of a given problem can use more than one of these strategies. At each branching node, the chosen branching methods compete with one another as follows. First, a score in [0, 1] is attributed to each potential branching strategy. Then, a hierarchy between these strategies is determined via a mapping of each strategy's score onto a global axis. The mapping is specific to each strategy and implemented as a scaling followed by a translation which are determined by two parameters; a maximum score value and a minimum score value. Finally, the branching strategy with the best final score is chosen. This strategy-dependent mapping process allows a strategy to be favoured to the detriment of others by defining a mapping that will produce higher final scores. For example, consider two branching methods, method 1 and method 2. Let the minimum and maximum score values for method 1 be 0 and 1, respectively. Also, let 0.5 and 1.25 be the minimum and maximum score values for method 2, respectively. The original score (in [0, 1]) is transformed according to the extremal score values for each method. In this case, method 2 is favoured compared to method 1. Indeed, for equal original scores, the transformed score of method 2 will be greater than the transformed score of method 1. In addition, if the score for method 1 is smaller

than 0.5 and possible branching decisions can be imposed by method 2, method 1 will never be chosen (since the minimal value for method 2 is 0.5).

The exploration of the search tree can be done according to three different search policies: depth first, best first and a combination of these two policies.

For a detailed description of the theoretical aspects behind GENCOL we refer to (Desaulniers et al., 1998) and (Desrosier et al., 1995).

2.4 A heuristic for the MOOCP with Satellites Sharing

In this section we will shortly describe a tabu search presented by Bianchessi et. al (2005) for the MOOCP with Satellites Sharing. Actually we will use the results produced by this heuristic to assess the strength of our formulations.

The authors developed a tabu search heuristic partly based on the method previously developed by Cordeau and Laporte (2005). The proposed algorithm 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 $M(s)$. The neighbourhood contains all the solutions that can be generated inserting or removing a mono strip (twin strips) in or from s and moving a mono strip (twin strip) from one orbit to another. Since this rule allows the solution to deteriorate between two successive iterations, they 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 the algorithm is the possibility of exploring infeasible solutions during the search. In particular, time window constraints are relaxed and their violations are added as penalities to the objective function. Let $f(s)$ be the value of solution s defined as $f(s) = v(s) - \beta w(s)$, where $v(s)$ is the objective function value defined by (2.1), 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 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. Finally the algorithm uses diversification and intensification mechanisms. The mechanism concerning diversification are the continuous scheme (now common to several tabu search implementations), the possibility to perturb the solution under certain cir-

cumstances, and the usage of false starts to avoid being trapped in a poor local optimum because of the moves performed in the first few iterations. On the other hand, 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 moves are considered whereas in orbit search, the neighbourhood is limited to insertion and removal moves. 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. Moreover, from time to time, a rescheduling of the strips in the solution is also performed in the hope of improving feasibility.

2.5 Computational results

To assess the strength of the formulation proposed, extensive computational experiments were performed on 13 data sets provided by the French Centre National d'Études Spatiales (CNES). The upper bounds founds have been compared with the solution values produced by the tabu search heuristic described in section 2.4.

Each data set (scenario) considers two satellites performing 12 or 13 orbits in a 24 hour time horizon and contains requests coming from 4 different entities: A', A'', B' and B''. We consider the problem arising for each entity of each scenario, for a total of 52 instances. When needed, in the following we refer to A' and A'' instances as A instances and to B' and B'' instances as B instances.

We first summarize in table 2.1 the main characteristics of the instances. This table contains 52 lines and each line provides the statistics for one data set and one user (users are identified by the digits $1, \ldots, 4$). 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 comprising more than one strip (PR), and the number of strips belonging to these polygon requests (SPR).

A instances are characterized by a mean number of orbits equal to 25.19 and are the biggest ones. The mean number of strips is equal to 2128.62 for A' instances and to 1589.08 for A'' instances. B instances have at most 5 orbits and 161 strips to consider. The presence of twin strips is more relevant in A than in B instances. The mean percentage of twin strips is equal to 58.86% and 70.49% respectively for A' and A'' instances, while it is always less than 33.60% for B instances. As far as strips associated with polygon requests are concerned, their mean number is approximatively the same in A instances. It is equal to 851.08 for A' instances and to 965.62 for A'' instances, with a mean number of strips per polygon equal

respectively to 3.35 and 3.94. Strips belonging to polygon requests in B instances are very few, always less than 51, with a mean number of strips per polygon equal to 2.24. Finally the number of strips generated by priority requests is always less than 32 and greater than 25 for all istances.

Experiments have been done with the requests of each individual user to estimate the maximal utility u_i^*. We used GENCOL 4.3.18 on a 2.53GHz Pentium 4 machine.

For each data set, we first found upper bounds for the time-constrained optimization (phase B) for users A' and A'': the users associated with the most complex instances. In this phase, user A' was allowed 750 seconds of satellite utilization and user A'' was allowed 600 seconds. The results of these experiments are reported in table 2.2. We indicate, for each data set, the estimate $\hat{u}_i^*$ of the maximal utility of each user i computed by the heuristic and the value of the upper bound $\dot{u}_i^*$ computed by column generation. The LP relaxation of model (2.2)-(2.8) was solved with a maximum CPU time of 24 hours.

For user A', 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. The gap between a dual bound and the correspondent heuristic value is always less than 3%.

We then found upper bound for the unconstrained optimization (phase C) for all users. In table 2.3 we again report the estimate $\hat{u}_i^*$ of the maximum utility computed with the tabu search heuristic and the upper bound computed by column generation $\dot{u}_i^*$. All the B instances' relaxations have been optimally solved within 6 minutes, while only about the 30% of A instances have been solved to the optimum with a time limit of 24 hours.
From the values listed in the third column, one can see that for users B instances, the lower and upper bounds on the true maximal utility are very close. For users A' and A'', the gap is larger but it is nevertheless below 3% for most instances.

Finally, in column $\ddot{u}_i^*$ upper bounds found disregarding twin strips constraints are reported; the mean worsening Δ of the dual bounds for A instances is equal to 4.99%. These upper bounds all represent the optimal value of the (2.2)-(2.7) LP relaxation, and the execution time needed to compute each of them was less than 1 hour; this fact can give intuitive understanding about the difficulties to handle twin strip constraints.

2.6 Conclusions

The planning and scheduling problem studied in this chapter involves two optical satellites performing multiple orbits over a given planning horizon.

Table 2.1: 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

Table 2.2: Results on phase B instances

Instance	$\dot{u}_i^*$	$\hat{u}_i^*$	gap (%)
16950_1	3805223404.2	3767992300	0.98
16950_2	3300098874.6	3211262510	2.69
16951_1	3744713939.5	3705839270	1.04
16951_2	3310306815.1	3264970158	1.37
16952_1	3628348154.8	3607404960	0.58
16952_2	3378020344.1	3315387600	1.85
16953_1	3640794101.2	3607518450	0.91
16953_2	3496010922.8	3437097170	1.69
16954_1	3824463515.6	3785693880	1.01
16954_2	3714752867.9	3659962180	1.47
16955_1	3813543894.8	3775211230	1.01
16955_2	3706782423.7	3629722570	2.08
16956_1	3786328424.2	3763423030	0.60
16956_2	3567789302.6	3463522270	2.92
16957_1	3630315249.6	3571726490	1.61
16957_2	3475485107.3	3367949820	3.09
16958_1	3599806509.2	3528185090	1.99
16958_2	3310679570.2	3244964180	1.98
16959_1	3591576836.1	3545725890	1.28
16959_2	3348676556.0	3273139990	2.26
16960_1	3774322817.8	3705166030	1.83
16960_2	3783371342.7	3670826820	2.97
16961_1	3834729713.3	3777339880	1.50
16961_2	3625005495.4	3538572480	2.38
16962_1	3819718271.3	3818995360	0.02
16962_2	3554436120.6	3465608200	2.50

Since 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. We have considered scenari arising for 2 satellites performing 12 or 13 orbits in a 24 hour time horizon with up to about 1400 requests.

We have dealt with the reachest model presented in the literature so far. In particular, in most of the previous works only a single satellite performing one or more orbits has been considered. Moreover in case of multiple orbits the problem has been semplified: it has been optimized one orbit at a time.

We have adopted a multi-commodity flow formulation which allows to decompose the MOOCP into a set-partitioning master problem and an ESPPTW for each orbit. In doing this we modelled twin strip constraints only at the subproblems level, and constraints concerning the acquisition of polygon requests at both levels. Owing to this formulation we have been able to successfully apply the

column generation technique in finding good dual bounds. Bounds obtained for large scale test instances provided by the CNES are at most 3.09% greater than heuristic solution values.

Finally we have put in evidence how twin strip constraints are more difficult to handle.

Table 2.3: Results on phase C instances

Instance	$\dot{u}_i^*$	$\hat{u}_i^*$	gap (%)	$\ddot{u}_i^*$	Δ (%)
16950_1	8121434414.40	7921269380	2.46	8613784020.00	5.72
16950_2	8519725566.33	8307825750	2.49	8967604188.89	4.99
16950_3	1148040470.00	1143215000	0.42	1148040470.00	0.00
16950_4	460304200.00	460304200	0.00	460304200.00	0.00
16951_1	8299202107.50	8142567589	1.89	8812323931.00	5.82
16951_2	8521161500.06	8307970310	2.50	8995192663.33	5.27
16951_3	1162619900.00	1162619900	0.00	1162619900.00	0.00
16951_4	432417350.00	432417350	0.00	432417350.00	0.00
16952_1	8120025507.10	8072741767	0.58	8575154725.33	5.31
16952_2	8334727139.47	8187730922	1.76	8721944699.83	4.44
16952_3	1450018375.00	1448856900	0.08	1450018375.00	0.00
16952_4	201099900.00	201099900	0.00	201099900.00	0.00
16953_1	10688895474.30	10517114800	1.61	11113519535.14	3.82
16953_2	8050833594.00	7941764820	1.35	8559587148.75	5.94
16953_3	1645170525.00	1638592300	0.40	1645170525.00	0.00
16953_4	454393137.50	453712550	0.15	454393137.50	0.00
16954_1	11181438393.00	10964766610	1.94	11616926583.36	3.75
16954_2	8767285963.40	8627522850	1.59	9299944360.00	5.73
16954_3	1626248245.00	1581862300	2.73	1635578775.00	0.57
16954_4	502219480.00	500463800	0.35	503491825.00	0.25
16955_1	11227212010.60	10965313340	2.33	11721749153.28	4.22
16955_2	9307786704.70	9162273460	1.56	9803741668.00	5.06
16955_3	1583466041.35	1561031550	1.42	1585216210.00	0.11
16955_4	730112125.00	727084100	0.41	730459278.57	0.05
16956_1	11141678306.70	10913704640	2.05	11540186616.36	3.45
16956_2	9111973423.27	8952492378	1.75	9683552034.67	5.90
16956_3	1419425350.00	1419425350	0.00	1419425350.00	0.00
16956_4	741393060.21	737224100	0.56	741465250.00	0.01
16957_1	11321455323.30	11036306720	2.52	11719446050.34	3.40
16957_2	8972665281.66	8744696924	2.54	9446975760.07	5.02
16957_3	918296683.33	916173400	0.23	918296683.33	0.00
16957_4	706337350.00	706321900	0.00	706533300.00	0.03
16958_1	11045919087.20	10771721876	2.48	11472186428.43	3.72
16958_2	8290378662.40	8133205239	1.90	8735993036.00	5.10
16958_3	1255131200.00	1255131200	0.00	1255131200.00	0.00
16958_4	534143750.00	534143750	0.00	534143750.00	0.00
16959_1	10785093906.10	10569211897	2.00	11189255707.21	3.61
16959_2	7695890218.20	7544282880	1.97	8178269269.73	5.90
16959_3	1569000650.00	1556985000	0.77	1569000650.00	0.00
16959_4	473587600.00	473587600	0.00	473587600.00	0.00
16960_1	10943618893.70	10785717007	1.44	11325278269.05	3.37
16960_2	8772225328.02	8632661360	1.59	9303287642.67	5.71
16960_3	1658119175.00	1645067000	0.79	1661300037.50	0.19
16960_4	339828650.00	339828650	0.00	339828650.00	0.00
16961_1	11251155019.10	10969531510	2.50	11744643925.97	4.20
16961_2	8930314251.43	8807377840	1.38	9454247950.00	5.54
16961_3	1649541270.20	1624832200	1.50	1651038146.67	0.09
16961_4	682521083.33	676261250	0.92	682521083.33	0.00
16962_1	11160184506.10	10903136180	2.30	11517864422.49	3.11
16962_2	9228142163.32	9051374719	1.92	9731362185.00	5.17
16962_3	1576907010.00	1573619000	0.21	1576907010.00	0.00
16962_4	705670500.00	696441700	1.31	705670500.00	0.00

Chapter 3

The Multi-Orbit Radar Constellation Problem

This chapter describes a work stemming from a project commissioned by Space Software Italia. The project concerns the management of the scientific activities of four SAR satellites, that corresponds to the scenario of the Italian *COSMO-SkyMed* project, currently under study. Owing to the SAR technology, the daylight data acquisition constraint can be removed, and the full orbit is available for acquisitions. We call this problem the *Multi-Orbit Radar Constellation Problem* (MORCP). This chapter describes greedy constructive (randomized) algorithms to solve the real problem taking into consideration all the details specified by Space Software Italia. It is organized as follows. In section 3.1 we describe the MORCP. Algorithms are explained in details in the subsequent section. To better highlight the bottleneck activities and slack in resources, in section 3.3 we report and discuss computational results obtained running the algorithms on different scenari. Finally, in section 3.4 concluding remarks are outlined.

3.1 Problem description

The MORCP is a complex combinatorial problem subject to a lot of technical and managerial constraints. Actually some aspects of the system behaviour are exceptions of the standard operating mode.

We recall that for this kind of satellites, only one azimuth is available for acquiring images (see section 1.4). In particular each time window associated with a swath reduces to a single starting time, and it is called Data Take Opportunity (in the remainder, when we say that a satellite is taking a DTO, we mean that it is acquiring the swath with which the DTO is associated). Even if the azimuth is constant, whenever a large target is split into several images, there still may exist a

combinatorial number of ways to combine the possible swaths in order to cover the target area. Also in this context it is assumed that the choice of which combination to consider has been done a priori, according to rules and considerations related to how easy it is to reconstruct the overall image (for instance, the orbits from which the images are taken must be either all descending or all ascending). Requests are weighted and since the system is intended for use by multiple classes of users with different priorities some requests can be classified as *high priority*, while others are *low priority*. Morevover, memory and energy constraints, as well as transmission features, are considered. The default objective is to maximize the linear quality criterion, but we consider also the one calling for the minimization of the time needed to satisfy high priority requests.

Both satellites and ground stations may be unavailable for some given time periods. During a period in which a station is unavailable no transmissions can be scheduled to that station. During a period in which a satellite is unavailable no acquisitions, transmissions or set-up changes can be planned for that satellite. Some unavailability periods for the satellites are artificially imposed because such periods of inactivity are used to replace an old plan by a new one.

Let us introduce some notation that will be used to illustrate in details the problem features. K is the set of satellites, indexed by k, and R represents the set of requests to satisfy in a given time horizon. For each request $r \in R$, we know its priority level p_r (1 for a priority request and 0 otherwise), and the set of images W_r associated with it. Then we define $W = \cup_{r \in R} W_r$ as the set of images. For each image $w \in W$, v_w specifies its value, $r(w)$ is the request to which it belongs, and D_w is the set of DTOs associated with it. Finally let $D = \cup_{w \in W} D_w$ be the set of DTOs. For each DTO $d \in D$, $[a_d,\ b_d]$ and s_d identify respectively the time interval and the satellite set-up *necessary* to take it, and $w(d)$ is the image with which it is associated. The duration of a DTO d is implicity given by $(b_d - a_d)$.

3.1.1 Setup constraints

There are three parameters that define the set-up of a satellite while it is taking a DTO. First, the satellite has two *orientations*, named "right-looking" and "left-looking" and it can rotate from one orientation to the other. Second, the SAR instrument can take images with different *look-angles*: to this purpose it can switch between l different positions. Third, the SAR instrument may work in six different *operating modes*, that is it can observe swaths of different width with different resolution and it can consume different amounts of energy for each acquisition. Thus the configuration needed to take a DTO $d \in D$ is described by the triplet <*orientation, look-angle type, operating mode*>. A set S contains all these triplets and for a DTO $d \in D$ the value of these three parameters is specified in $s_d = (or_d, lka_d, opm_d)$. Set-up operations may be necessary be-

tween two consecutive image acquisitions and the set-up times only depend on the two DTOs involved. Moreover the SAR instrument must be off for a minimum given amount of time between any two consecutive acquisitions. During this time interval it is not allowed to execute set-up operations either. Since the changes in orientation, look-angle and operating mode cannot be done simultaneously, the overall set-up time $t_{d_1 d_2}$ between two consecutive DTOs, d_1 and d_2, is the sum of four terms corresponding to the four mentioned parameters: $t_{d_1 d_2} = \delta_{or}(or_{d_1}, or_{d_2}) + \delta_{lka}(lka_{d_1}, lka_{d_2}) + \delta_{opm}(opm_{d_1}, opm_{d_2}) + \delta_{off}$.

3.1.2 Splitted requests

Whenever there are multiple images associated with the same request, they must be acquired with look-angles of the same type (*polygon constraints*). For look-angles of the same type we mean look-angles whose values belong to the same range. In particular there are three types of look-angle: *nominal, extended-high* and *extended-low*. A look-angle σ is nominal if $\sigma_{min} \leq \sigma \leq \sigma_{max}$, it is extended-low if $\sigma < \sigma_{min}$ and finally it is extended-high if $\sigma > \sigma_{max}$. To this purpose we make the following assumption. As soon as the first partial acquisition of the target is done, all the other images related to the same target are marked as *medium priority* and they are given precedence with respect to low priority ones. This is done in order to make unlikely the case in which a split target is only partially acquired.

3.1.3 Memory constraints

Each satellite, say k, has two indipendent memory blocks b_I^k and b_E^k of a given capacity $M_{b_I}^k$ and $M_{b_E}^k$. The number of files that can be stored on each block is also limited by $F_{b_I}^k$ and $F_{b_E}^k$. The meaning of subscripts I and E is respectively *internal* and *external*. Actually the efficiency with which the two memory devices can be used is different, and the usage of the internal block has to be preferred since it is the most efficient. A further constraint is that an acquired image must be stored entirely in the same block. Obviously for each image $w \in W$ we know its size m_w, and the number of files f_w in which it must be splitted once it is stored on a satellite memory device (all image segment files must have the same size multiple of 1 Mbit; this is guaranted through a pre-processing).

3.1.4 Operational profile constraints

There are some operational constraints on the sequence of image acquisitions that the SAR instrument can do. The constraints concern the activity time of the SAR instrument in each operating mode and they represent in a compact way

the constraints on the energy consumption of the instrument. For two operating modes (SPOTLIGHT1 and SPOTLIGHT2), the incidence on the workload is fixed, while for the other four (WIDEFIELD modes), the incidence is proportional to the acquisition duration. In the remainder we will use expression like "to acquire SPOTLIGHT1 images" to mean "to acquire images with SPOTLIGHT1 operating mode".

Nominal profiles. In every time window one day large the time spent to acquire WIDEFIELD images must be less than or equal to a given limit called T_{day} and the total number of SPOTLIGHT1 and SPOTLIGHT2 acquired images must be less than or equal to S_{day}. In every orbit the two limits are respectively T_{orbit} and S_{orbit} (*nominal profiles*). Considering a $24h$ time window, the former constraints bound the total workload while the latter are used to force a uniform distribution of it.

Peak profiles. Moreover for every satellite there are two possibilities to violate the nominal orbit profile constraints in a controlled way. A satellite can perform a *peak orbit* or a *triplet of peak orbits*.

- **Peak orbit:** for a time interval one orbit large, it can spend up to $4T_{orbit}$ time in acquisitions (a SPOTLIGHT1 or SPOTLIGHT2 are normalized through a constant factor δ and $S_{orbit}\delta = T_{orbit}$).

- **Triplet of peak orbits:** for a time interval three orbits large, the time spent to acquire WIDEFIELD images must be less than or equal to T_{orbit}, the number of SPOTLIGHT1 and SPOTLIGHT2 acquired images must be less than or equal to given limits called respectively $S1_{orbit}$ and $S2_{orbit}$. The three orbits defining the triplet must be consecutive.

A peak orbit and a triplet of peak orbits cannot overlap. Between two consecutive peak orbits a time window of at least $24h$ must elapse; the same holds for triplets of peak orbits.

As described by Space Software Italia, the constraints concerning the operational profiles represent the constraints concerning the energy consumption in a conservative way. Thus, in the remainder all the considerations that will be drawn about operational profiles directly apply to the energy resource.

3.1.5 Transmission

All data acquired by radar instruments have to be transmitted to ground stations. A set X of available ground stations is given and for each element in X we known

if it is a civil or a military station. A connection between a satellite and a ground station is possible only when the satellite footprint area is close enough to one of the ground stations located on the Earth surface. We call *down-link opportunity* (DLO) the visibility time-window. A DLO set is given for each satellite and ground station pair.

Transmission Channels. Transmission may occur through different channels, labelled as channel 1 and channel 2, corresponding to two different antennae mounted on board of each satellite $k \in K$. Both channels are capable of transmission at a given bit-rate BT and each of them can be used independently. Each station may receive information on one or two channels simultaneously; this is a given characteristic depending on the station. The DLOs of different stations for a same satellite may overlap; in this case the satellite can communicate with two of them simultaneously.

Transmission Modes. With each request $r \in R$ is associated a set of ground stations X_r and a transmission mode t_r. The transmission mode can be either "AND" or "OR" (t_r is equal to 1 for "AND" and 0 otherwise). If the transmission mode is "AND", each image $w \in W_r$ must be transmitted to every ground station listed in X_r, otherwise w must be transmitted only to one of the associated ground stations. If an image is segmented in more than one file, the image segment files can be transmitted separately and indipendently and, if the transmission mode is "OR", they can be transmitted to different ground stations. Each image segment file is indivisible: it must be transmitted completely and without interruption on the same channel to the same station.

GPS data. Ground stations can be either *military* or *civil*. If a DLO refers to a military station a certain amount of GPS data must be downloaded to the station. GPS data are automatically acquired by each satellite between each consecutive pair of DLOs related to military stations (the acquisition is done through an instrument different from the one used to acquire ordinary images). Thus the amount of GPS data that have to be transmitted in each DLO (which is small compared with the average size of the ordinary images) can be computed at preprocessing time. Within each military DLO, the given amount of GPS data must be downloaded without interruptions.

Transmission vs. Acquisition. Each image $w \in W$ has a given acquisition bit-rate B_w (in Mbps) associated with it, which is equal to the ratio between the size of the image and the duration of its acquisition. Moreover the default bit-rate BA available to store information is also given.

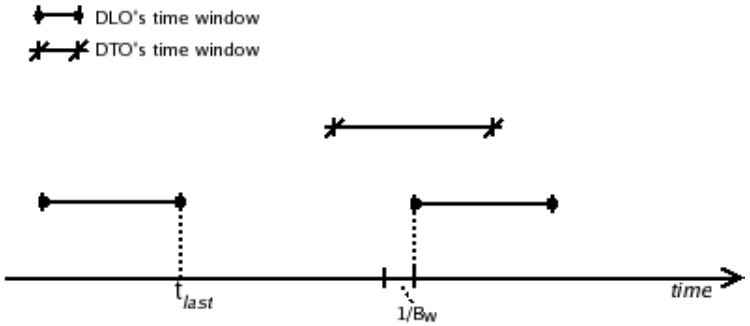


Figure 3.1: the DTO can be acquired and transmitted in (low-rate) pass-through mode.

As specified by Space Software Italia, storage and transmission can be generally done simultaneously. The conditions under which it is not possible vary depending on the operative mode used: *store-and-downlink* or *pass-through*.

The satellite is in *store-and-downlink* mode when it is acquiring an image $w' \in W$ and it is transmitting a file of an image $w'' \in W$, $w' \neq w''$. The usage of the store-and-downlink mode is forbidden if the satellite is transmitting w'' using channel 2 and $B_{w'} > BA$. Actually if $B_{w'} > BA$ the satellite must use also channel 2 to acquire w'.

On the other hand, the *pass-through* mode is used when the satellite is acquiring and transmitting the same image $w \in W$. Let t_{last} be the last usage time of the channel through which the transmission is intended to be done. The usage of the pass-through mode is forbidden if $f_w > 1$ or if t_{last} is greater than the starting acquisition time of w. Finally there is a further condition applying in case of *low-rate pass-through*, that is when $B_w < BT$. In this case, if the difference between the starting acquisition time of w and its starting transmission time is less than $1/B_w$, low-rate pass-through is forbidden. Actually $1/B_w$ is the time needed to store 1 Mbit in the memory buffer, the minimum transmissible data size (figure 3.1).

Download time. Except in *low-rate pass-through* case, the time τ needed to transmit a file of an image $w \in W$ is equal to the ratio between the image file size and BT. In the former case τ is equal to the maximum between τ' and τ'', where $\tau' = m_w/BT$ and τ'' is equal to $1/B_w$ plus the difference between the final acquisition time of w and its starting transmission time (figures 3.2 and 3.3).

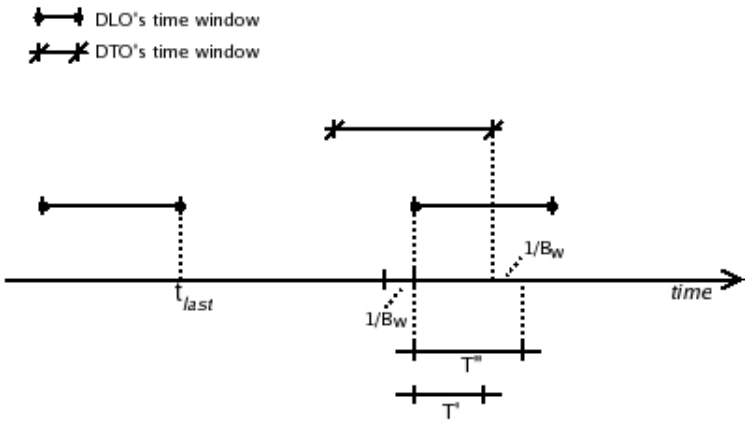


Figure 3.2: the DTO can be acquired and transmitted in low-rate pass-through mode; the time τ needed for transmission is equal to $T'' \equiv \tau''$.

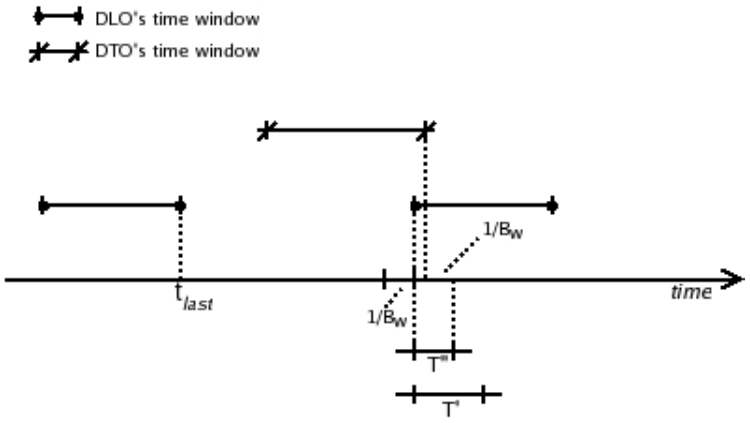


Figure 3.3: the DTO can be acquired and transmitted in low-rate pass-through mode; the time τ needed for transmission is equal to $T' \equiv \tau'$.

3.2 Algorithms

Planning and scheduling operations are subject to some limits on the computing resources available: in particular, the computation of a plan on sixteen days can take up to approximately 2.5 hours, while 50 minutes are allotted for computing a four-day plan. After each run, the algorithm must store all pieces of information needed to define the subsequent plan in a consistent way.

We addressed the planning and scheduling problem defined above by designing greedy constructive (randomized) algorithms. Actually these types of algorithms have the following advantages. First, their computational complexity is linear in the number of DTOs and DLOs and this allows to deal with very large problem instances. Then, they are easy to modify in order to take into account successive modifications of the model.

We are going to describe heuristics that produce feasible solutions with respect to all technical constraints presented in section 3.1. On the other hand, through look-ahead and backtracking capabilities, the algorithms try to acquire the maximum number of high priority requests, but there are no guarantees that all of them will be taken.

3.2.1 Preprocessing

For each satellite and for each ground station, unavailability periods must be considered. All DLOs overlapping with ground station unavailabilities or with satellite unavailabilities are either partially reduced or disregarded. All DTOs overlapping with satellite unavailability periods are disregarded too. Unavailabilities play a role also during the execution that will be illustrated in the remainder.

3.2.2 Initialization

First of all, a global variable v, in which is stored the total value of taken images, is set to 0.

Then, for each satellite $k \in K$, a DTO list $DTOList^k$ (indexed by i^k) and a DLO list $DLOList^k$ (indexed by j^k) are loaded. Like for DTOs, with each DLO $d \in \cup_{k \in K} DLOList^k$ is associated a time window $[a_d,\ b_d]$ during which the corresponding ground station can receive data. Each of these lists is sorted chronologically, according to the starting time of DTOs and DLOs. The DTO lists are consistent with the actual requests set, in particular we have that $\cup_{k \in K} DTOList^k = D$.

The state of each satellite that comes out at the end of the previous planning is then restored. The main pieces of information maintained in the state of each satellite $k \in K$ are the following:

- f_I^k, f_E^k, m_I^k and m_E^k: number of files stored respectively in the internal (I) and external (E) memory blocks, and the corresponding memory occupation;
- L^k: list of stored files;
- a record op^k of values concerning operational profiles.

In particular, the record op^k comprises seven fields:

- t_{day}^k and s_{day}^k: time spent in acquiring WIDEFIELD images and number of SPOTLIGHTx acquisitions done in the time window one day large defined as $\left[t^k - 24h, t^k\right]$;
- $t_{orbit}^k, s1_{orbit}^k$ and $s2_{orbit}^k$: time spent in acquiring WIDEFIELD images and number of SPOTLIGHT1 and SPOTLIGHT2 acquisitions done in the time window one orbit large defined as $\left[t^k - \frac{24h}{N}, t^k\right]$, where N is the number of orbits performed by a satellite in $24h$;
- t_{PO}^k: the starting time of the last peak orbit performed by the satellite k;
- t_{TPO}^k: the starting time of the last triplet of peak orbits performed by the satellite k.

Finally, an acquisition time and two transmission times are defined for each satellite $k \in K$. The acquisition time t_0^k is the time instant in which satellite k can start the next acquisition; the transmission time t_c^k for satellite k and channel $c = 1, 2$ is the time instant in which satellite k can start the transmission on channel c. A decision time t^k is then defined for each satellite k as $t^k = min(t_0^k, t_1^k, t_2^k)$.

It must be noticed that for a satellite $k \in K$, the value of t_{day}^k and s_{day}^k are computed considering all the acquisitions whose final time is greater than $(t^k - 24h)$. For a given time window, this avoids the problem to estimate the correct workload due to a fraction of an acquisition. $t_{orbit}^k, s1_{orbit}^k$ and $s2_{orbit}^k$ are computed in a similar way.

Routine Initialize
Input: sorted lists of DTOs and DLOs for each satellite; the state of each satellite $k \in K$ at the end of the previous planning.
Output: decision time t^k for each satellite $k \in K$.
begin
  $v = 0$;
  for each $k \in K$
    Load $DTOList^k$ and $DLOList^k$;
    $i^k = 1$; $j^k = 1$;
    Restore satellite's state;

/* Initialize acquisition and transmission times */
$\hat{d} = DTOList^k[i^k]$;
$t_0^k = a_{\hat{d}}$;
$\bar{d} = DLOList^k[i^k]$;
$t_1^k = t_2^k = a'_{\bar{d}}$;
/* Initialize decision time */
$t^k = min(t_0^k, t_1^k, t_2^k)$;
end

3.2.3 Main loop

The satellite with the minimum decision time is iteratively selected as the *active* satellite. Each time the decision time corresponds to the time instant in which a new DTO can be taken, a check for feasibility is done (*Feasible*). Suppose that the potential acquisition satisfies all constraints, if the DTO corresponds to a high or medium priority request, the *Take* function is executed, otherwise a *decision policy* is applied to decide whether to *Take* the DTO or to *Skip* it. On the other hand, if the acquisition leads to a constraints violation, the *Skip* will be the next action. The *Downlink* operation is carried out whenever the decision time of the active satellite corresponds to the transmission time of one of the two channels. This means that the decision time is certainly inside one or more DLOs (since the DLOs may overlap, it is possible that more than one station is available for transmission when the routine is executed).

/* Main loop */
repeat
  /* Select the active satellite $\bar{k}$ */
  $\bar{k} = arg\min_{k \in K}\{t^k\}$;
  if($t^{\bar{k}} = t_0^{\bar{k}}$) **then**
    if Feasible($DTOList[i^{\bar{k}}]$) **then**
      if(r(w($DTOList[i^{\bar{k}}]$)) is of high or medium priority) **then**
        Take;
      else
        Apply the decision policy to $DTOList[i^{\bar{k}}]$ and store the boolean outcome in TAKE;
        if(TAKE=TRUE)**then**
          Take;
        else;
          Skip;

else
Skip;
else
Downlink;
until ($t^{\bar{k}} < H$) /* H is the planning horizon */

Feasible

During the feasibility check the algorithm looks-ahead for DTOs corresponding to higher priority requests incompatible with the DTO to be taken for overlaps or insufficient transition time. If it finds some, the *Skip* action will be executed.

A DTO can be taken only if the following conditions are verified:

- if the DTO is associated with an image that results from a target splitting and it is not the first acquisition related to that target, its look-angle must be compatible with those of the already acquired images related to the same target;
- if the DTO is associated with an image $w \in W$ such that $w_B > BA$, channel 2 must not be in use;
- if the DTO is not associated with a high priority request, constraints on operational profiles must be satisfied;
- memory constraints must be satisfied.

Operational profiles. For the active satellite $\bar{k}$ and a DTO $d \in D^{\bar{k}}$ related to a non-high priority request $r(w(d))$, the operational profile constraints checking is done as follows. Let b_d the final acquisition time of d. We redefine the last time windows one day large and one orbit large associated with the satellite as $[b_d - 24h, b_d]$ and $\left[b_d - \frac{24h}{N}, b_d\right]$. Decreasing the values of $t^{\bar{k}}_{day}$, $s^{\bar{k}}_{day}$, $t^{\bar{k}}_{orbit}$, $s1^{\bar{k}}_{orbit}$ and $s2^{\bar{k}}_{orbit}$ as a consequence of the forward shift of the time windows, and increasing some of them according to the features of the image $w(d)$, we can check if this new acquisition leads to profile constraint violations (we can check if d is not feasible).

Let $\hat{t}^{\bar{k}}_{day}$, $\hat{s}^{\bar{k}}_{day}$, $\hat{t}^{\bar{k}}_{orbit}$, $\hat{s}1^{\bar{k}}_{orbit}$ and $\hat{s}2^{\bar{k}}_{orbit}$ be the new values describing operational profiles. If the profile constraints concerning the day are violated ($(\hat{t}^{\bar{k}}_{day} > T_{day}) or (\hat{s}^{\bar{k}}_{day} > S_{day})$), d is infeasible and cannot be taken. The same happens if the satellite $\bar{k}$ is performing a peak orbit or a triplet of peak orbits ($(b_d - t^{\bar{k}}_{PO} \leq \frac{24h}{N})$ or $(b_d - t^{\bar{k}}_{TPO} \leq 3\frac{24h}{N})$) and peak profiles are violated: $(\hat{t}^{\bar{k}}_{orbit} + (\hat{s}1^{\bar{k}}_{orbit} + \hat{s}2^{\bar{k}}_{orbit})\delta) > 2T_{orbit}$ in case of peak orbit or $((\hat{t}^{\bar{k}}_{orbit} > T_{orbit}) or (\hat{s}1^{\bar{k}}_{orbit} > S1_{orbit}) or (\hat{s}2^{\bar{k}}_{orbit} >$

$S2_{orbit}$)) in case of triplet of peak orbits. If the satellite is performing a nominal orbit the constraints checking is more complicated and it is done in the following way. If $((\hat{t}^{\bar{k}}_{orbit} \leq T_{orbit}) and (\hat{s}1^{\bar{k}}_{orbit} + \hat{s}2^{\bar{k}}_{orbit} \leq S_{orbit}))$, then d can be taken, otherwise a further check must be done concerning the possibility to start a peak orbit or a triplet of peak orbits. Four different situations can arise:

- $((b_d - t^{\bar{k}}_{PO} \leq \frac{24h}{N}) and (b_d - t^{\bar{k}}_{TPO} \leq 3\frac{24h}{N}))$: neither a peak orbit nor a triplet of peak orbits can be started, thus d is infeasible;
- $((b_d - t^{\bar{k}}_{PO} > \frac{24h}{N}) and (b_d - t^{\bar{k}}_{TPO} \leq 3\frac{24h}{N})$ or $((b_d - t^{\bar{k}}_{PO} \leq \frac{24h}{N}) and (b_d - t^{\bar{k}}_{TPO} > 3\frac{24h}{N}))$: only a peak orbit or only a triplet of peak orbits can be started, if the acquisition satisfies the relative profiles, d can be taken; otherwise d is infeasible;
- $((b_d - t^{\bar{k}}_{PO} > \frac{24h}{N}) and (b_d - t^{\bar{k}}_{TPO} > 3\frac{24h}{N}))$: both a peak orbit and a triplet of peak orbits can be started; if all peak profiles are violated, d is infeasible otherwise it can be taken.

In the last case, after a time period one orbit large (starting from b_d) in which both $t^{\bar{k}}_{PO}$ and $t^{\bar{k}}_{TPO}$ have been temporarily set to b_d, if peak profiles concerning the triplet of peak orbits have been violated, $t^{\bar{k}}_{TPO}$ is reset to its previous value, otherwise $t^{\bar{k}}_{PO}$ is reset.

Since constraints on operational profiles are very conservative, we allow to violate them whenever it is necessary to take a DTO related to a high priority request. Actually this kind of requests represents a small fraction of the set: if constraint violations arise, a feasible solution can be easily recovered in a post-processing phase.

Memory. As far as memory constraints are considered, if the amount of available memory or the number of storable files is not sufficient in any of the memory devices (internal block and external block), and $w(d)$ is not a low priority image, the algorithm *backtracks*.

Backtracking involves two phases, named *Delete* and *Restore*. In the *Delete* phase the most recently taken images are scanned in reverse chronological order and they are tentatively eliminated from the plan, until enough memory resources become available to store $w(d)$ in one of the two memory devices. During this phase an acquisition cannot be eliminated if it has already been partially transmitted. Moreover images can be deleted only if they are related to requests whose priority level is strictly lower than $r(w(d))$. The *Delete* phase stops going back in time when the starting time of the plan is reached. If this stop criterion prevents the backtracking routine from making enough memory resources available, the *Delete* phase fails, the DTO to be taken is considered infeasible and no modification is

made to the plan. On the contrary, if the *Delete* phase succeeds, the DTO is considered as feasible and the *Restore* phase starts. In the *Restore* phase the list of temporarily deleted acquisitions is scanned again in chronological order and all the acquisitions which do not make the DTO infeasible are restored (figure 3.4).

If backtracking succeeds, the value of v is suitable modified. On the other hand, since to maintain consistency in the state of satellites is a very complicated task, not all values describing the state of satellite $\bar{k}$ are updated. In particular $t^{\bar{k}}_{PO}$ and $t^{\bar{k}}_{TPO}$ are not modified. It may happen that a peak orbit or a triplet of peak orbits that have been started before backtracking are now useless as a consequence of the deletion of some acquistions. However, situations in which peak orbits are not well exploited after having been started are very unusual. It is for this reason that we do not care to modify the value of $t^{\bar{k}}_{PO}$ and $t^{\bar{k}}_{TPO}$. Nevertheless it must be noticed that this is a further restriction for the operational profiles. Finally, for all satellites $k \in K$, each DTO $\hat{d} \in DTOList^k$ associated with an image whose acquisition has been deleted is restored (see also the *Take* operation).

After backtracking, if the DTO has been made feasible, it is taken; otherwise it is skipped.

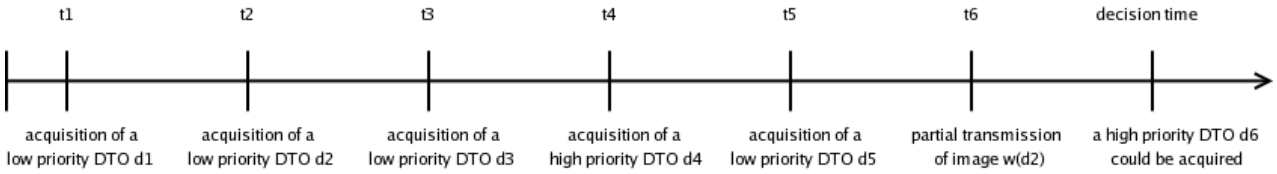
Decision policies.

Decision policies may take into account different deterministic or probabilistic criteria, depending on the value of the image, on the station to which it must be transmitted, on the state of the active satellite, etc. (later we will discuss them in more details).

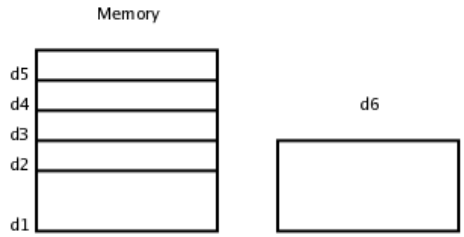
Take

Once the decision to acquire an image is taken, all pieces of information needed to update the active satellite's state are computed during the feasibility check.

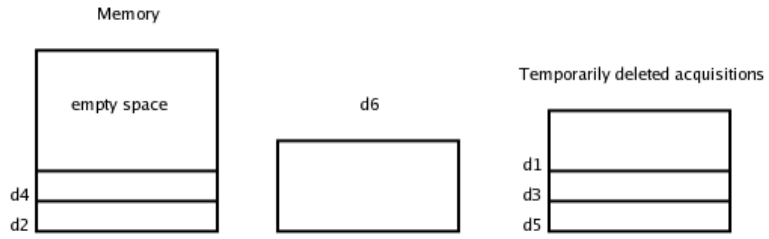
Then it is necessary to maintain consistency in the system. First of all, from the list $DTOList^{\bar{k}}$ must be deleted all DTOs incompatible with $DTOList^{\bar{k}}[i^{\bar{k}}]$ (for overlaps or insufficient transition time) and all next DTOs $\hat{d}$ such that $w(\hat{d}) = w(DTOList^{\bar{k}}[i^{\bar{k}}])$. To this purpose, satellite unavailabilities must be taken into account. Consider the pair of DTOs d_1 (equals to $DTOList^{\bar{k}}[i^{\bar{k}}]$) and d_2 separated by an unavailability period. Let t_{start} and t_{end} be the starting and ending time of the unavailability period considered. If $b_{d_1} + t_{d_1 d_2} > t_{start}$ and $t_{end} + t_{d_1 d_2} > a_{d_2}$, DTO d_2 is incompatible with the acquisition of DTO d_1. Second, for each satellite $k \in K$, the DTO list $DTOList^k$ must be updated removing all next DTOs $\hat{d}$ such that $w(\hat{d}) = w(d_1)$ and when $DTOList^k[i^k]$ changes, the acquisition time t^k_0 must be updated too. After the deletion process, $i^{\bar{k}}$ is incremented and $t^{\bar{k}}_0$ is updated accordingly.



There in no more memory available at decision time.



Delete phase: acquisitions are considered in reverse chronological order (from ``decision time" downto t1) and are tentatively eliminated. The acquisition of DTO d4 cannot be deleted as d4 is associated with a high priority request; since image w(d2) has been partially tranmitted, the acquisition of DTO d2 cannot be deleted too.



Restore phase: temporarily deleted acquisitions are considered in chronological order and all the acquisitions which do not make infeasible the acquisition of DTO d6 are restored.

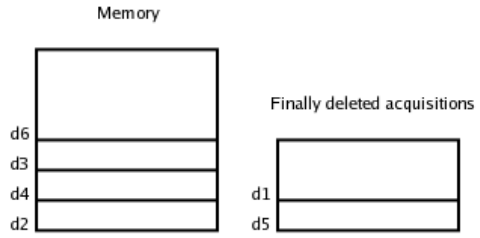


Figure 3.4: qualitative example of a succeeding backtracking process caused by an insufficient amount of available memory.

As a consequence of the previous operations, for a satellite $k \in K$ it may be necessary to modify its decision time t^k together with the values in its state regarding operational profiles.

Routine Take
Input: DTO index $i^{\bar{k}}$.
Output: a new state for the active satellite $\bar{k}$ and possibly also for other satellites $k \neq \bar{k}$.

begin
$d = DTOList^{\bar{k}}[i^{\bar{k}}]$
$v := v + v_{w(d)}$

/* Update active satellite's state */
if $((m_I^{\bar{k}} + m_{w(d)}) \leq M_{b_I}^{\bar{k}})$ and $(f_I^{\bar{k}} + f_{w(d)}) \leq F_{b_I}^{\bar{k}}))$
then
$m_I^{\bar{k}} += m_{w(d)}$; $f_I^{\bar{k}} += f_{w(d)}$;
else
$m_I^{\bar{k}} += m_{w(d)}$; $f_I^{\bar{k}} += f_{w(d)}$;
$L^{\bar{k}} \leftarrow \hat{L}^{\bar{k}}$;
$op^{\bar{k}} \leftarrow \widehat{op}^{\bar{k}}$;

/* Clean DTO lists for the active satellite*/
Delete from $DTOList^{\bar{k}}$ all DTOs incompatible with $DTOList^{\bar{k}}[i^{\bar{k}}]$ for overlaps or insufficient transition time;
Delete from $DTOList^{\bar{k}}$ all next DTOs of the same image as $w(DTOlist^{\bar{k}}[i^{\bar{k}}])$;

/* Clean DTO lists for the other satellites*/
for each $k \in K, k \neq \bar{k}$
Delete from $DTOList^k$ all next DTOs of the same image as $w(DTOlist^{\bar{k}}[i^{\bar{k}}])$;
If $DTOList^k[i^k]$ changes, update acquisition time t_0^k;

/* Find next DTO for the active satellite */
$i^{\bar{k}} = i^{\bar{k}} + 1$;

/* Update acquisition time $t_0^{\bar{k}}$ for the active satellite */
$d = DTOList^{\bar{k}}[i^{\bar{k}}]$
$t_0^{\bar{k}} = a_d$;

/* Update decision times for all satellites */
for each $k \in K$
$t^k = min(t_0^k, t_1^k, t_2^k)$;
If t^k changes, update values in op^k;
end

Skip

If DTO $DTOList^{\bar{k}}[i^{\bar{k}}]$ is infeasible, this routine simply skips it incrementing the value of $i^{\bar{k}}$ and updating the value of $t_0^{\bar{k}}$. Again, if it is necessary to change the value of the decision time, the values in $op^{\bar{k}}$ must be properly updated too.

Routine Skip
Input: DTO index $i^{\bar{k}}$.
Output: a new state for the active satellite $\bar{k}$.
begin
/* Find next DTO for the active satellite */
$i^{\bar{k}} = i^{\bar{k}} + 1$;
/* Update acquisition time $t_0^{\bar{k}}$ */
$d = DTOList^{\bar{k}}[i^{\bar{k}}]$;
$t_0^{\bar{k}} = a_d$;
/* Update decision time for the active satellite */
$t^{\bar{k}} = min(t_0^{\bar{k}}, t_1^{\bar{k}}, t_2^{\bar{k}})$;
If $t^{\bar{k}}$ changes, update values in $op^{\bar{k}}$;
end

Downlink

Each time a downlink operation must be executed, the decision time is equal to the minimum between $t_1^{\bar{k}}$ and $t_2^{\bar{k}}$, the transmission times associated with the channels of the active satellite. At this time, a set $G_{t^{\bar{k}}}$ of DLOs is available for the active satellite, and each DLO in $G_{t^{\bar{k}}}$ can be characterized by the remaining time available for downlink to the corresponding ground station.

The routine searches for a data file that can be transmitted to one of the ground stations available at that moment. Precedence is given to GPS data files, then stored image segmented files are considered. The file must have a size such that its transmission ends before the DLO of the corresponding station. When $t^{\bar{k}} = t_2^{\bar{k}}$ and the satellite is performing the acquisition of another image $w \in W$

such that $B_w > BA$ (the default bit-rate available to store information), no downlink operations can be done (*store-and-downlink* constraint). On the other hand, if the transmission concerns the same image that the satellite is currently acquiring, conditions regarding *pass-through* or *low-rate pass-through* must be verified (in particular, for each available satellite's channel, the last usage time is a piece of information stored in the satellite state). Finally, in order to make unlikely the cases in which a split target is only partially acquired or a high priority request is not satisfied, if $t^{\bar{k}} = t_2^{\bar{k}}$ we force the algorithm to look-ahead for DTOs associated with higher priority requests whose acquisition starts before the end of the downlink operation and requires the usage of channel 2. If some are found, transmission is prevented.

In case of downlink, if the file transmitted is associated with an image $w \in W$ such that $t_{r(w)}$ is equal to "OR", it can be deleted from $L^{\bar{k}}$. The deletion occurs also if $t_{r(w)}$ is equal to "AND" and the file has been already transmitted to all the stations listed in $X_{r(w)}$. Moreover, if the transmission is done on channel 2, every DTO overlapping with the transmission period and whose acquisition bit-rate is greater than BA is deleted from $DTOList^{\bar{k}}$. This automatically satisfies the second condition tested in the feasibility check, whereas *store-and-downlink* constraints can still be violated (for example at the beginning of a DLO). Finally, if after a downlink operation $t^{\bar{k}}$ changes, the values in $op^{\bar{k}}$ must be updated.

The downlink routine must also keep consistent the time in which a satellite can start a transmission with the times in which the available ground stations can receive data. To perform this task, with each DLO $d \in \cup_{k \in K} DLOList^k$ we associate the actual starting time a'_d in which the corresponding station can receive data (at the beginning $a'_d = a_d$ for all $d \in \cup_{k \in K} DLOList^k$) and we keep these lists sorted according to the value of this new quantity. Whenever a new set G of overlapping DLOs can be defined starting from $\hat{d} = DLOList^{\bar{k}}[j^{\bar{k}}]$, the active satellite's transmission times are equal to $a'_{\hat{d}} = a_{\hat{d}}$. Then, after each downlink operation, one or both of them are updated together with the actual starting times of each DLO $d \in G_{t^{\bar{k}}} \subseteq G$.

When $t^{\bar{k}} = min\{t_1^{\bar{k}},\ t_2^{\bar{k}}\} = a'_{\hat{d}}$ (with $\hat{d} = DLOList^{\bar{k}}[j^{\bar{k}}]$), the satellite $\bar{k}$ can transmit to one of the stations associated with a DLO contained in $G_{t^{\bar{k}}} \subseteq G$. If $t_1^{\bar{k}} = t_2^{\bar{k}}$ channel 1 is chosen and in the remainder, without loss of generality, we suppose that $t_1^{\bar{k}} \leq t_2^{\bar{k}}$. Let τ be the time used for a transmission on channel 1 ($t_1^{\bar{k}} \leq t_2^{\bar{k}}$) towards the station associated with DLO $d \in t_G^{\bar{k}}$. $t_1^{\bar{k}}$ becomes equal to $t_1^{\bar{k}} + \tau$. If $|G_{t^{\bar{k}}}| = 1$ and d is associated with a station that may receive information only from one channel, also a'_d and $t_2^{\bar{k}}$ become equal to $t_1^{\bar{k}} + \tau$ since no other transmissions can be done in the meanwhile. Otherwise all the actual starting times associated with the DLOs contained in $G_{t^{\bar{k}}}$ become equal to $t_2^{\bar{k}}$: the next time at which the satellite $\bar{k}$ can perform a transmission.

It may happen that no transmission can be done at a given decision time $t^{\bar{k}}$: there are no data to transmit towards the available stations, $t^{\bar{k}} = t_2^{\bar{k}}$ and a store-and-downlink cannot take place, a pass-through should take place but not all the constraints are satisfied, etc. In these cases a time needed for transmission $\tilde{\tau}$ is properly computed in order to set the trasmission time associated with the channel that must be used to the minimum among the following values:

- the starting time $a_{DTOList^{\bar{k}}[i^{\bar{k}}]}$ of the next available DTO for the active satellite;
- the ending time b_d associated with each DLO $d \in G_{t^{\bar{k}}}$;
- the actual starting time a'_d associated with each DLO $d \in G \setminus G_{t^{\bar{k}}}$;

the actual starting time of DLOs $d \in G_{t^{\bar{k}}}$ is then updated as previously described. In particular when $|G_{t^{\bar{k}}}| = 1$, it is like if a "dummy" transmission is performed towards the station corresponding to $d \in G_{t^{\bar{k}}}$.

After actual starting times updating, the list $DLOList^{\bar{k}}$ could require a reordering. The DLO indexed by $j^{\bar{k}}$ in the list $DLOList^{\bar{k}}$ could change. If this happens, the actual starting time of this new DLO has to be checked: it must be possibly set to the minimum between the new values of $t_1^{\bar{k}}$ and $t_2^{\bar{k}}$ (the next time at which the satellite $\bar{k}$ can perform a transmission).

Moreover $a'_{DLOList^{\bar{k}}[j^{\bar{k}}]}$ could now be equal to $b_{DLOList^{\bar{k}}[j^{\bar{k}}]}$. In such a case $j^{\bar{k}}$ is incremented until this condition holds. If the new DLO $DLOList^{\bar{k}}[j^{\bar{k}}] \notin G$, $t_1^{\bar{k}}$ and $t_2^{\bar{k}}$ must be set to $a'_{DLOList^{\bar{k}}[j^{\bar{k}}]}$, otherwise $t_1^{\bar{k}}$ and $t_2^{\bar{k}}$ are already set to the correct values.

Routine Downlink
Input: DLO index $j^{\bar{k}}$.
Output: a new state for the active satellite $\bar{k}$.

begin
$\hat{c} = arg\ min_{\hat{c} \in \{1,2\}}\{t_{\hat{c}}^{\bar{k}}\}$;
$G_{t^{\bar{k}}}$: the set of DLOs available for the active satellite at decision time;
G: the set of DLOs that overlap with $DLOList^{\bar{k}}[j^{\bar{k}}]$; $G \supseteq G_{t^{\bar{k}}}$;
F: the set of data files that could be transmitted towards a ground station associated with a DLO $d \in G_{t^{\bar{k}}}$;

/* If possible, execute a downlink */
Look for the first file $f \in F$ that can be transmitted;
if (found) **then**
Compute the time τ needed for transmission;

Update the list of stored files $L^{\bar{k}}$;
else
$\tau = \tilde{\tau}$;

Update the transmission times associated with the active satellite;
Update the actual starting time a'_d associated with each DLO $d \in G_{t^{\bar{k}}}$;

/* Update DLO list */
$d = DLOList^{\bar{k}}[j^{\bar{k}}]$;
Reorder the list $DLOList^{\bar{k}}$ according to the actual starting time;
if $(d \neq DLOList^{\bar{k}}[j^{\bar{k}}])$ **then**
$a'_{DLOList^{\bar{k}}[j^{\bar{k}}]} = min(t_1^{\bar{k}}, t_2^{\bar{k}})$;

/* Update $j^{\bar{k}}$ and possibly $t_1^{\bar{k}}$ and $t_2^{\bar{k}}$ */
repeat
$j^{\bar{k}} = j^{\bar{k}} + 1$;
until $a'_{DLOList^{\bar{k}}[j^{\bar{k}}]} = b_{DLOList^{\bar{k}}[j^{\bar{k}}]}$;

if $(DLOList^{\bar{k}}[j^{\bar{k}}] \notin G)$ **then**
$t_1^{\bar{k}} = t_2^{\bar{k}} = a'_{DLOList^{\bar{k}}[j^{\bar{k}}]}$;

/* Update decision time for the active satellite */
$t^{\bar{k}} = min(t_0^{\bar{k}}, t_0^{\bar{k}}, t_0^{\bar{k}})$;
If $t^{\bar{k}}$ changes, update values in $op^{\bar{k}}$;
end

3.2.4 Finalization

After the planning, the final states of the satellites are stored. Moreover, for each satellite, a log file describing the sequence of activities it should perform is given in output.

Here you can see a log file passage: two DTO acquisitions and two downlink operations are listed.

```
...
TIME: 2003- 0- 1T 1:46:49.140Z
Satellite's setup changes:
SAR Operating Mode:  SPOTLIGHT2 - unchanged
```

```
SAR Incidence Angle:  HIGH EXTENDED - unchanged
Looking Direction:  RIGHT - unchanged
Satellite 1 has started to get DTO 101829...
TIME: 2003- 0- 1T 1:46:55.240Z
Satellite 1 has terminated to get DTO 101829.
DTO 101829 splitted in 1 files.
Memory block used:  1.
Actual memory available on block 1:  85560.00 Mbit
Number of files that can be still stored on block 1:  215.
Actual memory available on block 2:  70580.00 Mbit
Number of files that can be still stored on block 2:  206.

TIME: 2003- 0- 1T 1:52:24.428Z
Satellite's setup changes:
SAR Operating Mode:  from SPOTLIGHT2 to PINGPONG
SAR Incidence Angle:  from HIGH EXTENDED to NOMINAL to LOW EXTENDED
Looking Direction:  RIGHT - unchanged
Satellite 1 has started to get DTO 65648...
TIME: 2003- 0- 1T 1:52:29.428Z
Satellite 1 has terminated to get DTO 65648.
DTO 65648 splitted in 1 files.
Memory block used:  1.
Actual memory available on block 1:  84212.00 Mbit
Number of files that can be still stored on block 1:  214.
Actual memory available on block 2:  70580.00 Mbit
Number of files that can be still stored on block 2:  206.

TIME: 2003- 0- 1T 2:  3:17.690Z
Satellite 1 has started downlink at station 1 on DLO 1...
Downloading on channel 1
20977 271 1:  file 0 - transmission time:  23.900 seconds - downlink and delete
Satellite 1 has finished to download.

TIME: 2003- 0- 1T 2:  3:17.690Z
Satellite 1 has started downlink at station 1 on DLO 1...
Downloading on channel 2
60726 463 1:  file 0 - transmission time:  8.987 seconds - downlink and delete
Satellite 1 has finished to download.
...
```

3.2.5 Decision policies

The behaviour of the algorithm depends on how the decision routine is defined. We have considered 6 different algorithms. Half of them are deterministic, and the others are randomized. The first and the second decision policies are trivial, the others are defined with the aim of optimizing the performance of the system, that is to maximize the global value of taken images.

Algorithm 0 always takes DTOs whenever possible.

On the other hand, **Algorithm 1** uses a random probability to decide whether to take a DTO or not. A new value for a normal random variable P is computed each time the decision policy is executed. The value of P represents the probability with which the DTO is taken.

In defining **Algorithm 2** we suppose that the value of a DTO $d \in D$ could be proportional to the correspondent amount of information $w(d)_m$ (number of bits). d is taken with probability $P' = \frac{w(d)_m}{\hat{M}}$, where $\hat{M}$ is (an estimate of) the maximum size of an image.

In **Algorithm 3** we balance the amount of information to be transmitted to each ground station. To this purpose we need to give precedence to the DTOs requiring transmission to less used stations, rather than to the DTOs for which the ground station availability is likely to be a bottleneck. Therefore for each satellite $k \in K$ and each DTO $d \in DLOList^k$ requiring transmission to a station $x \in X$, we consider the total amount m_x^k of memory currently occupied on board of the satellite by files to be transmitted to station x and we define $P'' = 1 - \frac{m_x^k}{(M_{b_I}^k + M_{b_E}^k)}$. A DTO is taken with probability P''.

In **Algorithm 4**, for the sake of balance between the ground stations, we consider the total number n_x^k of files stored on board of satellite $k \in K$ to be transmitted to station $x \in X$ and we forbid the acquisition of images related to stations for which n_x is maximum unless all values are equal. This is the second deterministic algorithm together with Algorithm 0.

Finally in **Algorithm 5** for each satellite $k \in K$ we consider again the total amount m_x^k of memory currently occupied on board of the satellite by files to be transmitted to station $x \in X$ and we define $m^+ = \max_{x \in X}\{m_x\}$ and $m^- = \min_{x \in X}\{m_x\}$. We forbid the acquisition of images such that (a) $m^+ - m^-$ increases and (b) after the acquisition $m^+ - m^- > \Delta$, where Δ is a suitably tuned threshold.

3.2.6 Objective functions

We have explicitly taken into account the default objective (calling for the maximization of the linear quality criterion) defining the decision policies properly. On the other hand, the minimization of the time needed to satisfy high priority

requests is embedded in the algorithms. Backtracking and look-ahead capabilities allow to acquire high priority requests as soon as possible. Then the downlink routine gives precedence to their transmission.

3.3 Computational results

3.3.1 Planning and scheduling scenari

We have identified a reference scenario (scenario 0) for the sake of comparison between the different algorithms. The best algorithm has then been executed on alternative scenari to put in evidence bottleneck activities and slack in resources. The alternative scenari have been generated by changing one of the parameters defining the reference scenario.

Reference scenario

Nominal values for the parameters in scenario 0 are set as follows.

- Number of satellites = 4.
- Set of ground stations = {Kiruna, Fairbanks, Matera}.
- The set of ground stations to which each request must be transmitted includes 1 station (at random).
- Number of requests = 2000 per day.
- Position of the requests: randomly spread all over the Earth surface: the longitude is randomly generated with uniform probability distribution in the range [1..360]; the latitude is randomly generated with uniform probability distribution in the range [−60..60].
- The DTOs for each request are all those complying with a deadline generated at random with uniform probability distribution in the range [1..3] days.
- The acquisition operating mode is generated at random with uniform probability distribution in a range [1..6]; each number corresponds to one of six different operating modes (HUGEREGION, WIDEREGION, SPOTLIGHT1, SPOTLIGHT2, PINGPONG, HIMAGE).

Alternative scenari

We have considered the following alternative scenari, in which one characteristic of the reference scenario has been changed.
Scenario 1. Number of satellites equal to 2 instead of 4. The satellites are considered at 180 degrees from each other. We considered the same requests and DLOs as in scenario 0.
Scenario 2. Number of ground stations equal to 2 instead of 3: Kiruna has been eliminated, keeping only Fairbanks and Matera. The same requests of scenario 0 have been used but those requiring transmission to Kiruna have been randomly assigned to Matera and Fairbanks.
Scenario 3. Number of transmissions for each image equal to 2, instead of 1. To each request of scenario 0 another station has been added, generating it at random among the two stations not associated with the request in scenario 0. In this scenario the transmission delay is computed between the request time and the beginning of the first transmission, while the aging is computed between the acquisition time and the beginning of the last (second) transmission.
Scenario 4. Number of requests equal to 1000 per day, instead of 2000 per day. We kept half of the requests of scenario 0, corresponding to a number of DTOs equal to 374376, instead of 750687.
Scenario 5. Concentrated requests. We have considered requests concentrated in some regions instead of uniformly scattered in latitude and longitude. We have roughly modelled the continents by rectangles in latitude and longitude, to divide "ground" from "sea" areas and we have generated 2000 requests per day centered only around points belonging to "ground" areas.
Scenario 6. Larger deadlines, varying at random in the range [4...6] days instead of [1...3] days. This yielded a number of DTOs equal to 1681592, instead of 750687.
Scenario 7. Acquisition modes without SPOTLIGHT images. All SPOTLIGHT requests of scenario 0 have been converted at random to one of the other 4 modes.

3.3.2 Test results

Experiments have been done on a 1.60GHz Pentium 4 machine. We have run the randomized algorithms five times and we have considered the best solutions found. Table 3.1 summarizes the outcome of our experiments on the reference scenario defined above. Time is indicated in hours:minutes:seconds. We report the computational time of the algorithms and the value of the solutions (the total size of the taken images, measured in Gbits). We also indicate the number and the percentage of taken images, the average time between request time and acquisition time, the average time between request time and transmission time, the average time between acquisition and downlink (aging of information on board), the aver-

Table 3.1: Comparison of different policies on the reference scenario

Algorithm	**0**	**1**	**2**	**3**	**4**	**5**
Comp. time (sec.)	57	267	289	276	55	55
Value (Gbit)	61231	61393	61270	60896	60980	60928
Taken im. num	8927	8912	8993	8920	9262	8960
Taken im.(%)	27.90	27.85	28.10	27.88	28.94	28.00
Access time	59:41:18	60:45:21	59:36:38	59:56:28	59:16:51	59:54:11
Transm. time	62:45:25	63:54:39	62:41:15	63:02:19	62:04:12	62:59:22
Aging	03:16:11	03:20:57	03:17:20	03:15:29	02:58:16	03:17:11
Mem. occ. (%)	38.47	39.75	38.75	37.62	37.08	38.31
Files in mem. (%)	6.03	6.24	6.09	5.94	5.98	6.04
Fairbanks (%)	24.01	24.17	23.77	24.91	26.01	24.16
Kiruna (%)	21.28	20.45	21.05	21.11	23.36	20.45
Matera (%)	47.89	49.91	48.83	45.82	41.94	48.58
Acq. time/orbit	00:03:19	00:03:18	00:03:19	00:03:18	00:03:16	00:03:18
Set-up time/orbit	00:08:42	00:09:19	00:09:11	00:09:13	00:08:47	00:08:51

age percentage level of memory occupation, the average percentage number of files stored in memory, the average percentage of time available for transmission which is actually used for transmission for each station, the average acquisition time per satellite and per orbit and the average setup time per satellite and per orbit.

It is remarkable how the results produced by all the algorithms look so similar. The number of taken images is about 28% of the total. The images are acquired in average 2.5 days after their submission, and three hours later they are transmitted back to the Earth.

The number of files that can be stored on board of each satellite is far from being a critical resource, and even if the memory is more exploited, its average percentage level of occupation is always less than 40% for each satellite.

As far as the average percentage of time used for transmission is concerned, it is always less than 50% for Matera and always less than 24% for the other ground stations. This is a consequence of the place where ground stations are located on the Earth surface (Fairbanks is located in Alaska while Kiruna is in the north of Sweden).

Finally it must be noticed that the average time spent in acquisition per orbit is about $\frac{1}{3}$ of an orbit duration; this is partially due to the time needed to perform set-up operations among consecutive acquisitions.

These results give evidence of how the critical resource of the system (in its default configuration) is neither the memory on board of each satellite, nor the transmission time available for each ground station. Actually, as we will show in

details in the next chapter, the operational profile constraints are the tightest for the system.

As shown by computational results, Algorithm 4 allows to acquire the greatest number of images; therefore the experiments on the alternative scenari have been carried out using this algorithm. Test results on the alternative scenari are reported in table 3.2.

The analysis of scenario 1 confirms that the overall acquisition and transmission capacity of the constellation is directly proportional to the number of satellites, whenever it is less than four.

In scenario 2 with only two ground stations available, the amount of information obtained is 66% with respect to scenario 0 with three ground stations. Transmission time to Matera is more heavily exploited in percentage than transmission time to Fairbanks. Matera tends to be the bottleneck and this is a consequence of the limited time available to communicate with it. The aging of acquired images is about 5 hours, i.e. 2 hours more with respect to the reference scenario, and this leads to an increase of the average percentage level of memory occupation and of the average percentage number of files stored.

In scenario 3 the necessity of double transmission for each image results in a loss in number and value of taken images of about 17% and 27%. This suggests that the transmission activity can influence the solution quality even if transmission time is not a critical resource, as shown by the percentages of its exploitation.

In scenario 4, the number of requests per day is halved with respect to the scenario 0. With only 1000 requests/day the capabilities of the satellites are not fully exploited, actually the value and the number of taken images decrease of about 36% and 24%.

In scenario 5, where the images requested are more concentrated, so that the number of conflicts is higher, there is a loss of about 44% in value and of about 33% in number of taken images with respect to the reference scenario.

In scenario 6, deadlines have been enlarged so that there are approximately twice as many data-take opportunities for each image with respect to scenario 0. Nevertheless, we have again losses, even if they are smaller with respect to the previous scenario: about 21% in value and about 5% in number of taken images.

In scenario 7, without SPOTLIGHT images, the performances of the algorithms degrade since operational profiles are not fully exploited (this issue will be covered in details in the next chapter).

The analysis of the computing time required by the algorithms on the alternative scenari confirms the validity of the theoretical estimation of the algorithms complexity: the computing time is linear in the number of satellites and the number of DTOs. The dependency on the number of DLOs is not observable because it is dominated by the dependency on the DTOs that are much more numerous.

3.3.3 High priority requests

In scenario 0, with each request has been associated a low priority. In order to test backtracking and look-ahead capabilities of the algorithms, we have randomly set to high the priority of 6400 requests (20% of the total). Then we have run the Algorithm 4 on the modified scenario 0. 96% of high priority requests have been satisfied. This is a very good result if we consider that no pre-processing has been done to guarantee the feasibility of high priority request acquisitions. Moreover, high priority images represent the 99% of those taken.

3.3.4 Failures

The equipment can be broken off. For a satellite $k \in K$ only a limited amount of memory can be available on a block at a certain time. Defining properly the value of the parameters $M^k_{b_I}, F^k_{b_I}$ and $M^k_{b_E}, F^k_{b_E}$ these situations can be easily modeled. Moreover, only one specific transmission channel, between the two, can be available to make a plan. During a downlink, in case of a channel failure, if $\bar{c}$ is the channel available for the active satellite $\bar{k}$, all the actual starting times associated with the DLOs contained in $G_{t^{\bar{k}}}$ are set equal to $t^{\bar{k}}_{\bar{c}} + \tau$ (where τ is the time used for the transmission).

Tests have been done for different failures scenari: one of the memory device broken, one channel available for transmission towards ground stations, and a mix of the previous failures. Experiments have been done considering a single satellite on a planning horizon of 1 day. The name of each considered scenario is composed of two digits: the number of memory blocks devices that can be used to store data files and the number of channels available for transmission. Results are reported in table 3.3. The results computed for scenari 1-2 and 1-1, compared respectively to those reported for scenari 2-2 and 2-1, show how a failure in one of the memory devices does not affect the activity that can be carried out by a satellite. On the other hand, the unavailability of one of the two transmission channels causes a mean loss of 12.94% in the number of taken images. Once again, these last results suggest that the transmission activity can influence the performance of a satellite even if transmission time is not a critical resource.

3.4 Conclusions

The optimization problem presented in this chapter concerns the management of the scientific activities of four satellites equipped with SAR instruments performing multiple orbits in a maximum planning horizon of 16 days. We have considered the real problem, taking into account all the details specified by Space Software Italia.

Table 3.2: Alternative scenari, Algorithm 4

Scenario	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Comp. time (sec.)	55	26	57	63	30	59	152	56
Value (Gbit)	60980	29185	39551	43790	38905	34421	48188	33816
Taken im. num	9262	4824	7168	7643	7016	6252	8789	6189
Taken im.(%)	28.94	15.07	22.40	23.88	43.85	19.54	21.47	19.34
Access time	59:16:51	61:31:21	16:53:05	46:46:40	12:30:14	20:40:11	30:24:23	18:10:33
Transm. time	62:04:12	64:18:02	21:28:17	48:12:49	15:51:37	24:13:25	33:46:34	21:43:22
Aging	02:58:16	02:55:31	04:54:55	02:56:44	03:35:06	03:48:53	03:45:15	03:52:53
Mem. occ. (%)	37.08	29.40	86.15	44.80	70.61	68.56	81.11	82.82
Files in mem. (%)	5.98	4.79	12.12	6.43	9.79	9.47	11.33	11.18
Fairbanks (%)	26.01	12.99	50.75	35.50	36.84	32.91	39.67	41.30
Kiruna (%)	23.36	11.28	-	32.97	32.13	28.19	35.08	35.93
Matera (%)	41.94	17.20	95.57	44.43	73.53	68.67	83.04	80.56
Acq. time/orbit	00:03:16	00:01:38	00:05:04	00:07:14	00:05:47	00:05:12	00:06:55	00:06:25
Set-up time/orbit	00:08:47	00:04:17	00:04:16	00:04:36	00:05:17	00:05:40	00:05:14	00:04:45

Table 3.3: Failure scenari, Algorithm 4, $H = 24h$

Scenario	**2-2**	**1-2**	**2-1**	**1-1**
Value (Gbit)	918	920	804	806
Taken im. num	152	149	129	133
Taken im.(%)	7.60	7.45	6.45	6.65
Access time	09:13:16	09:08:08	08:28:46	09:01:47
Transm. time	11:31:56	11:52:51	10:47:45	11:06:50
Aging	03:15:57	03:27:57	03:16:28	03:19:22
Mem. occ. (%)	32.64	35.10	58.07	62.09
Files in mem. (%)	5.49	5.85	9.63	10.09
Fairbanks (%)	35.90	35.94	31.45	30.22
Kiruna (%)	34.80	35.24	33.87	31.99
Matera (%)	72.82	71.47	69.59	67.71
Acq. time/orbit	00:00:50	00:00:49	00:00:44	00:00:43
Set-up time/orbit	00:02:06	00:01:59	00:01:57	00:01:57

Among the others, we have dealt with constraints about the acquisition of polygon requests and with operational constraints regarding the transmission activity and the energy consumption on board of each satellite. To our knowledge, those kind of constraints have seldom been studied in the Operations Research literature related to this kind of problems. Moreover we have considered simultaneously two objectives: the maximization of the linear quality criterion and the minimization of the time needed to satisfy high priority requests.

We have defined greedy constructive (randomized) algorithms that compute feasible solutions with respect to the whole set of considered constraints. By means of backtracking and look-ahead capabilities they try to acquire all high priority requests as soon as possible and to terminate the acquistion of splitted requests partially taken. The minimization of the time needed to satisfy high priority requests has been taken into consideration by defining the capabilities previously mentioned and by giving precedence to high priority requests in transmission. The value of the taken images has been maximized using suitable decision policies within the algorithms.

Experimental results have been obtained running all algorithms on a reference scenario with 2000 requests per day on a planning horizon of sixteen days, corresponding to more than one million DTOs. Then the best algorithm has been tested on alternative scenari. For each scenario we could measure the percentage of use of each ground station as well as of all system resources, such as the memory on board of each satellite. We could also have a quantitative estimate of the access time and the transmission time of the taken images and the percentage amount of

time spent on acquisition and set-up operations for each orbit and each satellite. The results found show how transmission activity and energy consumption are the problem features that have the greatest impact on the solution quality. Finally we have successfully tested the robustness of the best algorithm in case of high priority requests and we have presented some results concerning failures scenari.

Owing to the features of the greedy algorithms, the observed computational times were largely inferior to the imposed limits, since they never exceeded a few minutes on the largest problem instances.

Chapter 4

An alternative solution methodology for the MORCP

In this chapter we present an alternative solution methodology for the MORCP based on Lagrangean relaxation; the aim is twofold. On one hand we look for good dual bounds. On the other hand, we try to define quasi-feasible solutions that can guide the heuristics presented in chapter 3 in finding better solutions than those computed from scratch. Because of the complexity of these tasks, we limit the planning horizon to a single day. According to the purpose of this chapter, we introduce in section 4.1 a valid relaxation for the MORCP. The resulting problem is modelled in section 4.2 through a multi-commodity flow formulation. In section 4.3 we present a decomposition approach based on Lagrangean relaxation. We describe the dynamic programming algorithm defined to solve the *Shortest Path Problem with Resource Constraints* arising for each satellite and we briefly discuss the subgradient algorithm applied to find the best dual bound. Starting from the presented approach, we illustrate in section 4.4 how quasi-feasible solutions can be obtained for the MORCP. Computational results are reported in section 4.5, followed by conclusions in section 4.6.

4.1 A relaxed problem

With respect to the features described in section 3.1, we relax the MORCP making the following assumptions.

Memory. For each satellite $k \in K$, the memory blocks can be viewed as a single big block with capacity $M^k = M^k_{b_I} + M^k_{b_E}$ in which an unlimited number of files can be stored.

Trasmission. We always allow a satellite to perform acquisition and transmission simultaneously, when the satellite is acquiring and transmitting the same image as well as different images. In particular we consider a download time τ equal to the ratio between the image file size and BT. Choosing this value for τ leads to a valid relaxation of the MORCP: actually in low-rate pass-through this value can be smaller than or equal to the real one (figures 3.2 and 3.3). Moreover, as far as transmission is concerned, for each request $r \in R$ we assume that X_r includes all the available ground stations and that t_r is always equal to "OR". Finally we assume that an image segmented file can be transmitted with interruptions, and we disregard transmission of GPS data.

Operational Profiles. We check the nominal operational profiles concerning every time window one orbit large only in some orbits (identified as described in section 4.2.2), and we disregard the nominal profiles constraints bounding the total workload in the unique time window $24h$ large. In the worst case, the additional time that can be used during a peak orbit is equal to $3T_{orbit}$. We do not restrict this additional amount of time to be available in a given time window one orbit large but we allow to consume it along the whole planning horizon, whenever required. Finally we disregard the possibility to acquire additional SPOTLIGHT1 or SPOTLIGHT2 images during a triplet of peak orbits.

4.2 Relaxed problem formulation

As well as in the case of the MOOCP, we model the relaxed problem by means of a multi-commodity flow formulation based on the unified framework described in (Desaulniers et al., 1998). In particular we adopt the task-on-arc formulation that allows to include in the network model some constraints on the path structure. In this context, a commodity and a task represent a satellite and an image respectively. Thus the set of images W, indexed by w, represents the set of tasks to cover (the set of images to acquire) and a commodity is defined for each satellite $k \in K$ (with $|K| = 4$).

In the following three sections all notation symbols defined refer to a commodity $k \in K$, but the superscript k has been dropped for simplicity; it will be reintroduced starting from section 4.2.4.

4.2.1 Commodity network

A directed acyclic graph $G(V, A)$ is related to each commodity. With the task-on-arc representation, nodes in V represent time-space sites.

In the set of nodes $V = N \cup \{o,\ d\}$, o and d represent the source and the sink nodes for the commodity. A node $i \in N$ represents the i-th DTO listed in the chronologically sorted list of DTOs that can be acquired by the satellite. When the satellite is in $i \in N$, its setup is the one needed to acquire DTO i. At the end of a plan the satellite setup in d is the one required by the last DTO taken, and this setup is going to characterize the node o in the subsequent planning. Let $\overline{ss}$ be the initial satellite setup for the actual planning, o can be viewed as a node associated with an artificial DTO o such that $ss_o = \overline{ss}$ and $[a_o,\ b_o] = [0,\ 0]$.

The arc set A is defined as follows. For each $i \in N \cup \{o\}$ and each $j \in N$, if DTOs i and j are not separated by an unavailability period for the satellite, the arc (i, j) is feasible if $b_i + t_{ij} \leq a_j$. Otherwise a further condition has to be checked. Let t_{start} and t_{end} be the starting and ending time of the unavailability period considered. If $b_i + t_{ij} > t_{start}$ and $t_{end} + t_{ij} > a_j$, DTO j is incompatible with the acquisition of DTO i and arc (i, j) cannot be inserted in A. The task covered by the arc (i, j) is the image $w(j)$ associated with the destination node. Arcs directed to d always satisfy the feasibility conditions, thus an arc (i, d) can be inserted in A for each $i \in N$. Those arcs do not cover any tasks.

Since the SPPRC is NP-hard in the strong sense, we build A reducing the number of arcs to be taken into account.

Let us consider the set S of possible satellite configurations (see section 3.1.1): for a given node $i \in N \cup \{o\}$ and a given setup $ss \in S$ we evaluate each arc (i, j) with $j \in N$ and $ss_j = ss$. If A already contains an arc (i, k) such that $ss_k = ss$ and it is possible to define a feasible arc (k, j), the arc (i, j) is disregarded since j is reachable from i through k. Otherwise if the arc (i, j) satisfies the feasibility conditions, it is inserted in A. Nodes $i \in N \cup \{o\}$ for which no leaving arcs have been defined are linked to the destination node d. Moreover for each arc $(i, j) \in A$ previously defined, we consider another arc (i, j) that allows to reach the node $j \in N$ without acquiring DTO j. Thus for each node $i \in N \cup \{o\}$ we have two sets of arcs: $A_{i,1}$ and $A_{i,0}$. The former includes all feasible arcs outgoing from i and covering the task associated with the destination nodes $j \in N$, whereas the latter includes all arcs that allow to reach the destination nodes while taking no image (figure 4.1). If $|A_{i,1}| = 0$, the set $A_{i,0}$ contains only the arc (i, d).

If arc $(i, j) \in \cup_{i \in N \cup \{o\}} A_{i,1}$, it is characterized by a profit v_{ij} equal to $v_{w(j)}$. The profit is set to 0 for the other arcs.

Furthermore each arc $(i, j) \in A$ is characterized by memory variation and operational profiles increments.

If arc $(i, j) \in \cup_{i \in N \cup \{o\}} A_{i,1}$ the memory variation is $m_{ij} = m_{w(j)} - \delta_{ij}$, otherwise $m_{ij} = -\delta_{ij}$. The parameter δ_{ij} models the amount of data that can be transmitted to ground stations in the time interval that elapses while the commodity is travelling along arc (i, j), that is $b_j - b_i$.

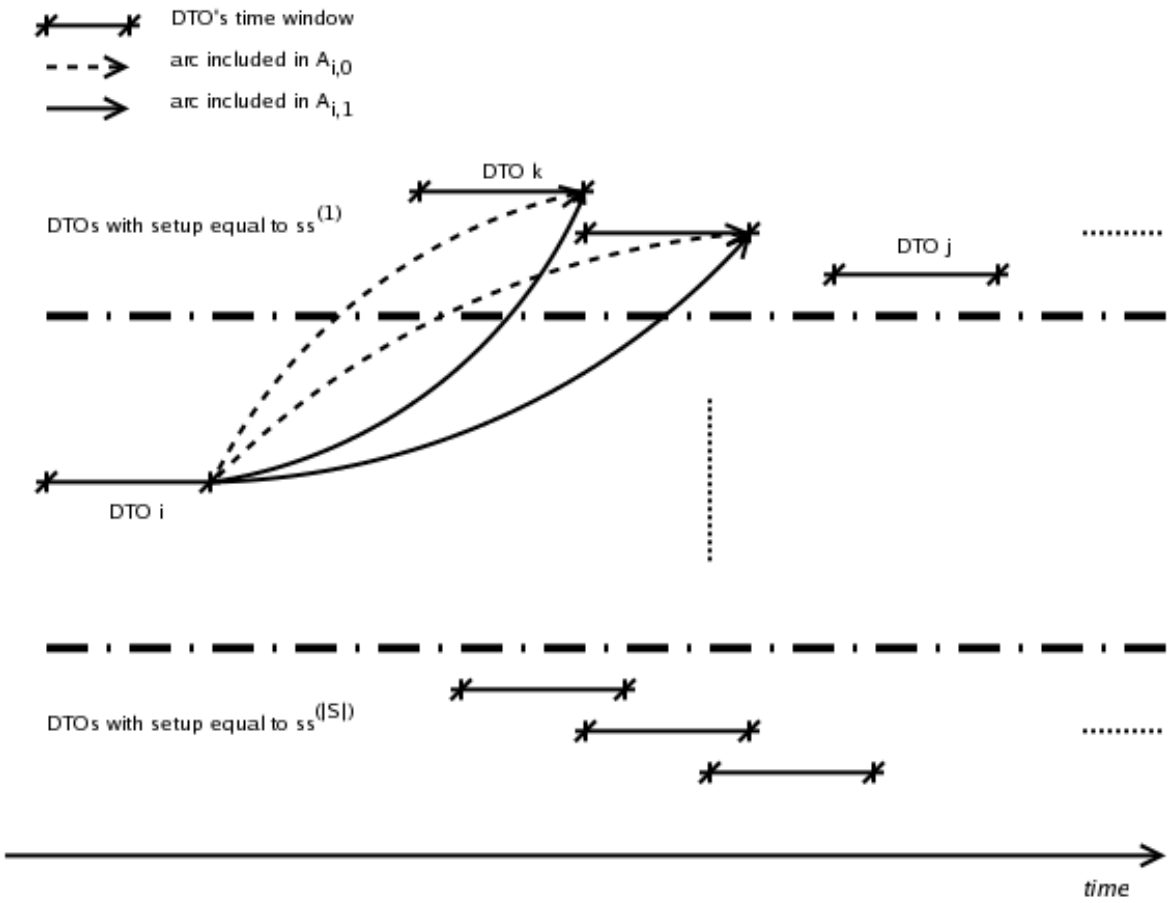


Figure 4.1: arcs included in $A_{i,1}$ and $A_{i,0}$ as far as setup $ss^{(1)} \in S$ is considered; DTO j can be acquired after DTO k.

If during this time interval no DLO is available for the satellite, then $\delta_{ij} = 0$. Otherwise different situations can arise: there can be stations with one or two channels, DLOs may overlap and also the number of available satellite's channels may vary. We distinguish between two types of available download time: Δ_s and Δ_d. The former takes into account all time intervals during which images can be downloaded to a single station with one channel. The latter specifies the amount of time during which a single station with two channels or more stations simultaneously are available. Let c be the number of transmission channels available for the satellite. If $c \geq 1$, $\delta_{ij} = [\Delta_s + (\Delta_d \cdot c)] \cdot BT$, otherwise $\delta_{ij} = 0$.

The time spent in acquiring WIDEFIELD images and the number of SPOTLIGHT1 and SPOTLIGHT2 acquired images are the two quantities constrained by operational profiles. Define ρ_{ij} and σ_{ij} respectively as the time increment and the increment in the number of SPOTLIGHT1 or a SPOTLIGHT2 acquired images. Let us consider an arc $(i, j) \in \cup_{i \in N \cup \{o\}} A_{i,1}$. If DTO j is associated with a WIDEFIELD image, then $\rho_{ij} = b_j - a_j$ and $\sigma_{ij} = 0$. On the other hand, if $w(j)$ is a SPOTLIGHT1 or SPOTLIGHT2 image, then $\rho_{ij} = 0$ and $\sigma_{ij} = 1$. For the other arcs $(i, j) \in A$, ρ_{ij} and σ_{ij} are set to 0.

4.2.2 Operational profile resources

As previously mentioned, the nominal profiles are checked only in a discrete number of orbits. To this purpose, for each commodity we fix the number Z of orbits to consider at a given time and we define them in the following way. For each $z \in \{1 \dots Z\}$ we split the time horizon in consecutive time windows one orbit large starting from the time instants $\frac{24h/N}{Z} \cdot (z - 1)$, where N is the number of orbits performed by a satellite in 24 hours. At the end of this process Z different sequences of orbits are defined and a given time instant belongs to Z orbits, one for each sequence (see figure 4.2).

To check operational profiles we define a resource set P^z, indexed by $p = 1 \dots 3$, for each commodity and for each sequence of orbits $z \in \{1 \dots Z\}$ In a given set P^z, P_1^z represents the nominal time spent in acquiring WIDEFIELD images, P_2^z represents the number of SPOTLIGHT1 and SPOTLIGHT2 acquired images and finally P_3^z represents the additional time spent (we recall that the additional time is not restricted to be available in a given time window one orbit large but it can be consumed during the planning whenever needed). Then we define T_i^{zp} as the resource variable specifying the amount of resource $p \in \{1 \dots 3\}$ for the sequence of orbits $z \in \{1 \dots Z\}$, consumed until node $i \in V$, on a path on the commodity network. A path is feasible only if $T_i^{zp} \in [a_i^{zp},\ b_i^{zp}]\ \forall z \in \{1 \dots Z\},\ \forall p \in \{1 \dots 3\}$, for all reached nodes $i \in V$, where $[a_i^{zp},\ b_i^{zp}]$ is equal to $[0,\ T_{orbit}]$, $[0,\ S_{orbit}]$ and $[0,\ 3T_{orbit}]$ respectively for $p = 1 \dots 3$, for each $z \in \{1 \dots Z\}$. That is a path is feasible only if a feasible amount of each resource has been consumed along it. For

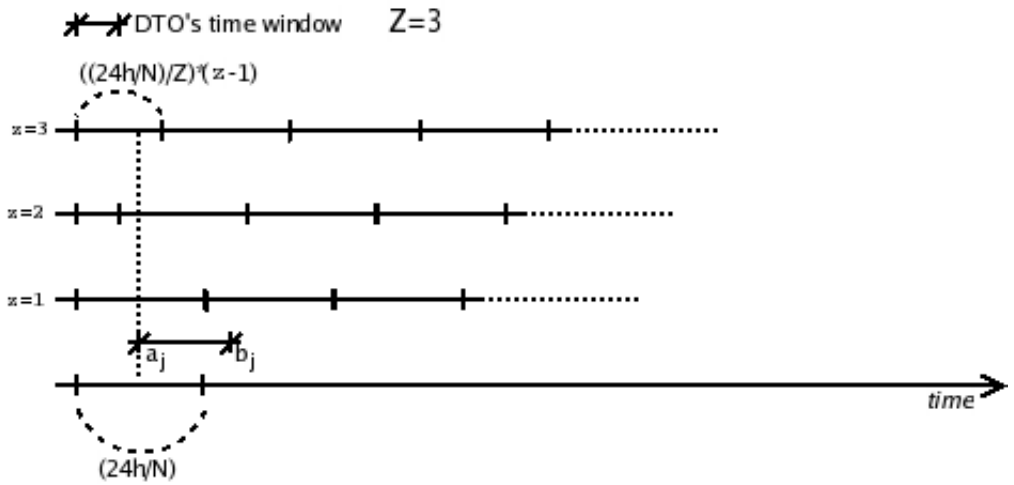


Figure 4.2: orbit identification.

each $z \in \{1 \dots Z\}$ and for each $p \in \{1 \dots 3\}$, the value of T_o^{zp} is fixed and it is equal to the corresponding value coming out at the end of the previous planning.

From node i to node j, $(i,j) \in A$, for each $z \in \{1 \dots Z\}$, the value of resource variable related to $p \in \{1 \dots 3\}$ is updated by means of the function $f_{ij}^{zp}(\mathbf{T}_i^z)$, where $\mathbf{T}_i^z = (T_i^{zp} \,|\, p \in \{1 \dots 3\})$. These functions are the so called *resource extension functions*. In the easiest cases f_{ij}^{zp} assumes the form of a linear function that depends only on the considered resource p, and it computes the consumption on the arc $(i,j) \in A$ simply by adding a value depending of the task covered by the arc. More complex extension functions have to be defined in our case.

Consider an arc $(i,j) \in \cup_{i \in N \cup \{o\}} A_{i,1}$. For a given sequence of orbits $z \in \{1 \dots Z\}$, the starting time a_j of DTO j lies in the time interval associated with the orbit $\omega^z(a_j)$ equal to $\lfloor \frac{a_j - \frac{24h/N}{Z}(z-1)}{24h/N} \rfloor + 1$ if $a_j \geq \frac{24h/N}{Z}(z-1)$ and to 0 otherwise. For example, in figure 4.2, $\omega^z(a_j)$ is equal to 1, 1 and 0 respectively for $z = 1 \dots 3$.

Let $\gamma^{\omega^z(a_j)}$ be the fraction of increments that influence the operational profiles in the orbit $\omega^z(a_j)$; $\gamma^{\omega^z(a_j)} = min\left\{\frac{b_j - \omega^z(a_j)\cdot(24h/N)}{b_j - a_j}, 1\right\}$.

Consider the case $0 < \gamma^{\omega^z(a_j)} < 1$. If $\omega^z(b_i) = \omega^z(a_j)$ like in the case depicted in figure 4.3, the value of $\mathbf{T}_i^z$ has to be modified accordingly to the fraction of increments $\gamma^{\omega^z(a_j)}$ before the extension, and has to be checked for feasibility. For a given arc, only the value of one resource between P_1^z and P_2^z can directly change. Suppose for example that $w(j)$ is a SPOTLIGHT1 or a SPOTLIGHT2 image, the new value $\overline{\mathbf{T}}_i^z$ is computed as follows. If $T_i^{z2} + \sigma_{ij}\gamma^{\omega^z(a_j)} \leq b_i^{z2}$ nominal profile costraints are satisfied: $\overline{T}_i^{z2} = T_i^{z2} + \sigma_{ij}\gamma^{\omega^z(a_j)}$ and the values of the other resource variables remain unchanged. If $T_i^{z2} + \sigma_{ij}\gamma^{\omega^z(a_j)} > b_i^{z2}$ it is necessary to consume

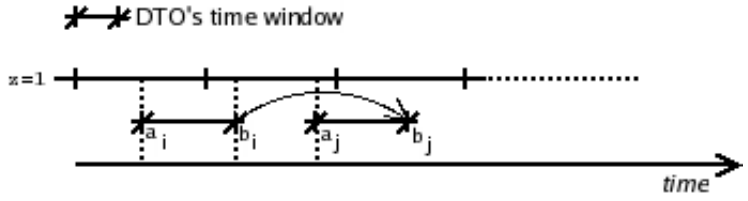


Figure 4.3: fraction of increments, case 1.

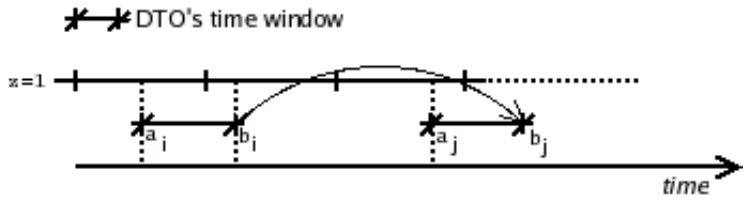


Figure 4.4: fraction of increments, case 2.

additional time: $\overline{T}_i^{z3} = T_i^{z3} + \delta(T_i^{z2} + \sigma_{ij}\gamma^{\omega^z(a_j)} - b_i^{z2})$, $\overline{T}_i^{z2} = b_i^{z2}$ and $\overline{T}_i^{z1} = T_i^{z1}$ (δ is the constant factor through which a SPOTLIGHT1 or a SPOTLIGHT2 acquired image is normalized).

On the other hand, if b_i and a_j are separated by more than one orbit as in figure 4.4, the partial increments cannot cause infeasibility in the orbit coming before $\omega^z(b_j)$, we set $\overline{\mathbf{T}}_i^z = \mathbf{T}_i^z$. If $\overline{\mathbf{T}}_i^z$ is feasible with respect to the resource windows associated with node i, the extension can take place. Since $\omega^z(b_i) \neq \omega^z(b_j)$, the consumption of resources P_1^z and P_2^z accumulated can be disregarded. The value of $\mathbf{T}_j^z$ is defined accordingly to the fraction of increments $\tilde{\gamma}^{\omega^z(a_j)} = (1 - \gamma^{\omega^z(a_j)})$.

Consider now the case $\gamma^{\omega^z(a_j)} = 1$. $\mathbf{T}_j^z$ is computed accordingly to the value of ρ_{ij} and σ_{ij}. In particular, if $\omega^z(b_i) \neq \omega^z(b_j)$ the consumption of resources P_1^z and P_2^z accumulated until node i can be disregarded. To this purpose let θ_{ij} be a binary coefficient equal to 0 if $\omega^z(b_i) \neq \omega^z(b_j)$.

For each commodity and each sequence of orbits $z \in \{1 \dots Z\}$ we can now define the extension functions associated with an arc $(i, j) \in A$. For the resources

P_1^z and P_2^z the functions look as follows:

$$f_{ij}^{z1} = \begin{cases} +\infty & \textbf{if}(0<\gamma^{\omega^z(a_j)}<1)\textbf{and}(\textbf{not}(\mathit{feasible}(\overline{\mathbf{T}}_i^z))) \\ \rho_{ij}\tilde{\gamma}^{\omega^z(a_j)} & \textbf{if}(0<\gamma^{\omega^z(a_j)}<1)\textbf{and}(\mathit{feasible}(\overline{\mathbf{T}}_i^z))\textbf{and}(\rho_{ij}\tilde{\gamma}^{\omega^z(a_j)} \leq b_j^{z1}) \\ b_j^{z1} & \textbf{if}(0<\gamma^{\omega^z(a_j)}<1)\textbf{and}(\mathit{feasible}(\overline{\mathbf{T}}_i^z))\textbf{and}(\rho_{ij}\tilde{\gamma}^{\omega^z(a_j)} > b_j^{z1}) \\ \theta_{ij}T_i^{z1}+\rho_{ij} & \textbf{if}(\gamma^{\omega^z(a_j)}=1)\textbf{and}(\theta_{ij}T_i^{z1}+\rho_{ij} \leq b_j^{z1}) \\ b_j^{z1} & \textbf{if}(\gamma^{\omega^z(a_j)}=1)\textbf{and}(\theta_{ij}T_i^{z1}+\rho_{ij} > b_j^{z1}) \end{cases}$$

$$f_{ij}^{z2} = \begin{cases} +\infty & \textbf{if}(0<\gamma^{\omega^z(a_j)}<1)\textbf{and}(\textbf{not}(\mathit{feasible}(\overline{\mathbf{T}}_i^z))) \\ \sigma_{ij}\tilde{\gamma}^{\omega^z(a_j)} & \textbf{if}(0<\gamma^{\omega^z(a_j)}<1)\textbf{and}(\mathit{feasible}(\overline{\mathbf{T}}_i^z))\textbf{and}(\sigma_{ij}\tilde{\gamma}^{\omega^z(a_j)} \leq b_j^{z2}) \\ b_j^{z2} & \textbf{if}(0<\gamma^{\omega^z(a_j)}<1)\textbf{and}(\mathit{feasible}(\overline{\mathbf{T}}_i^z))\textbf{and}(\sigma_{ij}\tilde{\gamma}^{\omega^z(a_j)} > b_j^{z2}) \\ \theta_{ij}T_i^{z2}+\sigma_{ij} & \textbf{if}(\gamma^{\omega^z(a_j)}=1)\textbf{and}(\theta_{ij}T_i^{z2}+\sigma_{ij} \leq b_j^{z2}) \\ b_j^{z2} & \textbf{if}(\gamma^{\omega^z(a_j)}=1)\textbf{and}(\theta_{ij}T_i^{z2}+\sigma_{ij} > b_j^{z2}) \end{cases}$$

and for P_3^z the extension function is:

$$f_{ij}^{z3} = \begin{cases} +\infty & \textbf{if}(0<\gamma^{\omega^z(a_j)}<1) \\ & \textbf{and}(\textbf{not}(\mathit{feasible}(\overline{\mathbf{T}}_i^z))) \\ \overline{T}_i^{z3}+(\rho_{ij}\tilde{\gamma}^{\omega^z(a_j)} - b_j^{z1})^+ + \delta(\sigma_{ij}\tilde{\gamma}^{\omega^z(a_j)} - b_j^{z2})^+ & \textbf{if}(0<\gamma^{\omega^z(a_j)}<1) \\ & \textbf{and}(\mathit{feasible}(\overline{\mathbf{T}}_i^z)) \\ T_i^{z3}+(\theta_{ij}T_i^{z1} + \rho_{ij} - b_j^{z1})^+ + \delta(\theta_{ij}T_i^{z2} + \sigma_{ij} - b_j^{z2})^+ & \textbf{if}(\gamma^{\omega^z(a_j)}=1) \end{cases}$$

4.2.3 Memory resources

To check memory constraints we define a new resource Q representing the memory consumption for each commodity. Let T_i^Q be the resource variable specifying the value of the resource Q accumulated at node $i \in V$ during a path, and let the resource window $[a_i^Q,\ b_i^Q]$ be equal to $[0,\ M]$ for each $i \in V$. For the considered resource, the extension function associated with an arc $(i,j) \in A$ can be defined as $f_{ij}^Q = T_i^Q + max\{0,\ T_i^Q + m_{ij}\}$.

4.2.4 A model

For $k \in K$, let x_{ij}^k, with $(i,j) \in A^k$, denote the flow variables which are equal to 1 for the arcs in the solution path of commodity k and 0 otherwise. Then let $\overline{W}$ be the set of images $w \in W$ such that $p_{r(w)} = 1$ and define also $\widehat{W} = \{w \in W \mid p_{r(w)} = 0\}$. The integer formulation for the relaxed version of the MORCP is the following:

$$\max \sum_{k \in K} \sum_{(i,j) \in A^k} v_{ij}^k x_{ij}^k \tag{4.1}$$

$$\text{s.t.:} \sum_{k \in K} \sum_{\substack{(i,j) \in A_{i,1}^k | \\ w(j) = w}} x_{ij}^k = 1 \qquad \forall w \in \overline{W} \tag{4.2}$$

$$\sum_{k \in K} \sum_{\substack{(i,j) \in A_{i,1}^k | \\ w(j) = w}} x_{ij}^k \leq 1 \qquad \forall w \in \widehat{W} \tag{4.3}$$

$$\sum_{j|(o^k,j) \in A^k} x_{o^k j}^k = 1 \qquad \forall k \in K \tag{4.4}$$

$$\sum_{j|(i,j) \in A^k} x_{ij}^k - \sum_{j|(j,i) \in A^k} x_{ji}^k = 0 \qquad \forall k \in K, \forall i \in N^k \tag{4.5}$$

$$\sum_{j|(j,d^k) \in A^k} x_{jd^k}^k = 1 \qquad \forall k \in K \tag{4.6}$$

$$x_{ij}^k (f_{ij}^{kzp}(\mathbf{T}_i^{kz}) - T_j^{kzp}) \leq 0 \quad \forall k \in K, \forall z \in \{1...Z^k\}, \forall p \in \{1...3\}, \forall (i,j) \in A^k \tag{4.7}$$

$$a_i^{kzp} \leq T_i^{kzp} \leq b_i^{kzp} \qquad \forall k \in K, \forall z \in \{1...Z^k\}, \forall p \in \{1...3\}, \forall i \in V^k \tag{4.8}$$

$$x_{ij}^k (f_{ij}^{Q^k}(T_i^{Q^k}) - T_j^{Q^k}) \leq 0 \qquad \forall k \in K, \forall (i,j) \in A^k \tag{4.9}$$

$$a_i^{Q^k} \leq T_i^{Q^k} \leq b_i^{Q^k} \qquad \forall k \in K, \forall i \in V^k \tag{4.10}$$

$$x_{ij}^k \in \{0,1\} \qquad \forall k \in K, \forall (i,j) \in A^k \tag{4.11}$$

The objective function (4.1) maximizes the sum of the arc profits. Linking constraints (4.2) are the set-partitioning constraints for priority requests, while linking constraints (4.3) state that each image belonging to non priority requests have to be acquired at most once. Constraints (4.4)–(4.6) define the classical network flow constraints for a path originating at source node and ending at destination node. The non-linear constraints (4.7, 4.9) express the compatibility requirements among flow and resource variables. Finally, the resource window constraints are given by (4.8, 4.10), while constraints (4.11) impose binary values for the flow variables. Constraints (4.4)–(4.11) are separable by satellite and together with (4.1) they define a SPPRC (*Shortest Path Problem with Resource Constraints*) structure for each commodity $k \in K$.

As far as dual bounds for the MORCP are concerned, those found solving a relaxation of this problem are valid except for a constant factor. Actually the profit of SPOTLIGHT1 and SPOTLIGHT2 images is equal to a fixed amount p, and in a triplet of peak orbits the number of SPOTLIGHT1 and SPOTLIGHT2

acquired images can increase up to $(S1_{orbit} + S2_{orbit})$ in each time window one orbit large. Thus an upper bound value for (4.1)–(4.11) becomes valid for the MORCP increasing it by a factor of $p \cdot 3 \cdot ((S1_{orbit} + S2_{orbit}) - S_{orbit}$.

4.3 Lagrangean relaxation

In the problem P defined by (4.1)-(4.11), the objective (4.1) and the constraints sets (4.4)–(4.11) are separable by commodity. We obtain dual bounds solving the problem by means of the Lagrangean relaxation decomposition approach (Geoffrion , 1974).

Consider the integer program:

$$(IP) \qquad \begin{aligned} z = \quad & max\ \mathbf{c}^T\mathbf{x} \\ & A\mathbf{x} \leq \mathbf{b} \\ & D\mathbf{x} \leq \mathbf{d} \\ & \mathbf{x} \in \mathbb{Z}^n_+ \end{aligned}$$

where A is an $l \times n$ matrix, D is an $m \times n$ matrix, $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^l$ and $\mathbf{d} \in \mathbb{R}^m$. Suppose that the constraints $A\mathbf{x} \leq \mathbf{b}$ are "nice" in the sense that an integer program with just these constraints is easy. Thus if one drops the "complicating constraints" $D\mathbf{x} \leq \mathbf{d}$, the resulting relaxation is easier to solve than the original problem IP. However, the resulting bound obtained from the relaxation may be weak, because some important constraints are totally ignored. One way to tackle this difficulty is by Lagrangean relaxation.

For any value of $\mathbf{u} = (u_1, \ldots, u_m) \geq \mathbf{0}$, the problem

$$(IP(\mathbf{u})) \qquad \begin{aligned} z(\mathbf{u}) = \quad & max\ \mathbf{c}^T\mathbf{x} + \mathbf{u}(\mathbf{d} - D\mathbf{x}) \\ & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \in \mathbb{Z}^n_+ \end{aligned}$$

is the *Lagrangean relaxation* of IP with parameter $\mathbf{u}$. In $(IP(\mathbf{u}))$ the complicating constraints are handled by adding them to the objective function as a penality term. The components of $\mathbf{u}$ are the *Lagrangean multipliers* (or *dual variables*) associated with the constraints $D\mathbf{x} \leq \mathbf{d}$. By solving the $(IP(\mathbf{u}))$ we obtain an upper bound on the optimal value of IP. To find the best (smallest) upper bound over the infinity of possible value for $\mathbf{u}$ it is necessary to solve the *Lagrangean dual* problem:

$$z_{LD} = min\{z(\mathbf{u}) | \mathbf{u} \geq \mathbf{0}\}.$$

We consider two different sets of Lagrangean multipliers. Linking constraints (4.2) are relaxed by means of multipliers λ_w, with $w \in \overline{W}$. Then, for each $w \in \widehat{W}$, a multiplier $\mu_w > 0$ is associated with the corresponding constraint in the set (4.3).

For given values of λ and μ, the Lagrangean relaxation $P(\lambda, \mu)$ can be defined in the following way:

$$z(\lambda, \mu) = \sum_{k \in K} z^k + \sum_{w \in \overline{W}} \lambda_w + \sum_{w \in \widehat{W}} \mu_w \tag{4.12}$$

where, for each commodity $k \in K$, z^k is the optimal value of the SPPRC modelled below.

$$\max \sum_{\substack{(i,j) \in A^k_{i,1} | \\ w(j) \in \overline{W}}} (v^k_{ij} - \lambda_w) x^k_{ij} + \sum_{\substack{(i,j) \in A^k_{i,1} | \\ w(j) \in \widehat{W}}} (v^k_{ij} - \mu_w) x^k_{ij} \tag{4.13}$$

$$\text{s.t.:} \quad \sum_{j | (o^k, j) \in A^k} x^k_{o^k j} = 1 \tag{4.14}$$

$$\sum_{j | (i,j) \in A^k} x^k_{ij} - \sum_{j | (j,i) \in A^k} x^k_{ji} = 0 \qquad \forall i \in N^k \tag{4.15}$$

$$\sum_{j | (j,d^k) \in A^k} x^k_{jd^k} = 1 \tag{4.16}$$

$$x^k_{ij}(f^{kzp}_{ij}(\mathbf{T}^{kz}_i) - T^{kzp}_j) \leq 0 \qquad \forall z \in \{1...Z^k\}, \forall p \in \{1...3\}, \forall (i,j) \in A^k \tag{4.17}$$

$$a^{kzp}_i \leq T^{kzp}_i \leq b^{kzp}_i \qquad \forall z \in \{1...Z^k\}, \forall p \in \{1...3\}, \forall i \in V^k \tag{4.18}$$

$$x^k_{ij}(f^{Q^k}_{ij}(T^{Q^k}_i) - T^{Q^k}_j) \leq 0 \qquad \forall (i,j) \in A^k \tag{4.19}$$

$$a^{Q^k}_i \leq T^{Q^k}_i \leq b^{Q^k}_i \qquad \forall i \in V^k \tag{4.20}$$

$$x^k_{ij} \in \{0, 1\} \qquad \forall (i,j) \in A^k \tag{4.21}$$

Solving $P(\lambda, \mu)$, we obtain a valid dual bound for the MORCP. In order to find the best dual bound we need to solve the Lagrangean dual problem defined as:

$$z_{LD} = min\{z(\lambda, \mu) | \mu \geq 0\}. \tag{4.22}$$

4.3.1 A dynamic programming algorithm for the SPPRC

For each commodity $k \in K$, the SPPRC is defined on an acyclic graph $G^k(V^k, A^k)$. Dynamic programming solution approaches for the SPPRC systematically build new paths from the source of the graph, by extending them from node to node into all feasible directions. Their efficiency depends on the ability to identify and

discard paths which are not useful either to build a Pareto-optimal set of paths or to be extended into Pareto-optimal paths. Discarding useless paths is achieved by applying dominance rules, which will be illustrated in the remainder.

To illustrate the dynamic programming algorithm implemented, some concepts have to be introduced; first of all, the concept of a *label*. Define $\mathcal{R}^k$ as the set of all resources considered for a given commodity $k \in K$, thus $\mathcal{R}^k = \{P_1^{k1}, P_2^{k1}, P_3^{k1}, \ldots, P_1^{kZ^k}, P_2^{kZ^k}, P_3^{kZ^k}, Q^k\}$. With each path $\mathcal{P}_i$ from the origin o^k to i satisfying the resource windows is associated a label $(\mathcal{T}_i, P_i)$. $\mathcal{T}$ is a vector $(T_i^1, T_i^2, \ldots, T_i^{|\mathcal{R}^k|})$ storing the quantity of each resource used by the path, and P_i is the path profit. These labels are iteratively computed along the path $\mathcal{P}_i = (i_0, i_1, \ldots, i_H)$ as:

$$(\mathcal{T}_{i_0}, P_{i_0}) = (T_{o^k}^{k11}, T_{o^k}^{k12}, T_{o^k}^{k13}, ..., T_{o^k}^{kZ^k1}, T_{o^k}^{kZ^k2}, T_{o^k}^{kZ^k3}, T_{o^k}^{Q^k}, 0)$$
$$(\mathcal{T}_{i_h}, P_{i_h}) = (f_{i_{h-1},i_h}^{kzp}(\mathbf{T}_{h-1}^{kzp}) \ \forall z \in \{1...Z^k\}, \forall p \in \{1...3\}; f_{i_{h-1},i_h}^{Q^k}(T_{h-1}^{Q^k}), P_{i_{h-1}} + v_{i_{h-1}i_h})$$

where $i_0 = o^k$ and $i_H = i$.

Let $(\mathcal{T}_i^1, P_i^1)$ and $(\mathcal{T}_i^2, P_i^2)$ be two different labels for two paths from o^k to i. The first label *dominates* the second, i.e., $(\mathcal{T}_i^1, P_i^1) \succ (\mathcal{T}_i^2, P_i^2)$ if and only if $(\mathcal{T}_i^2 - \mathcal{T}_i^1) \geq 0$ and $(P_i^1 - P_i^2) \geq 0$. A label $(\mathcal{T}_i, P_i)$ at a given node i is said to be *efficient* if no other labels at i dominate it. By extension, also the corresponding path is said to be *efficient*. This dominance relation defines a partial order on the labels and this implies the possibility of several efficient paths for each node.

Define now Q_i, $i \in V^k$ to be the set of labels of node i and let $EFF(Q_i)$ denote the set of efficient labels among the set of labels Q_i of node i. The set of efficient labels at each node can be computed by dynamic programming. The longest path from o^k to d^k satisfying the resource window constraints is obtained directly from the set $EFF(Q_{d^k})$: it is represented by the label with the greatest profit.

We implement a *label setting* algorithm. The defining property of a label setting algorithm is that those labels chosen to be extended (in the path extension step) are kept without modification until the end of the labelling process. They will not be identified as discardable in subsequent checks for dominance. Labelling algorithms that do not guarantee this behaviour are called *label correcting* algorithms. The general ideas of label setting as well as label correcting algorithms in the context of the one-dimensional Shortest Path Problem (SPP) are, for instance, explained in the book of Ahuja et. al. (1993). The algorithm proposed is an extension of the Dijkstra algorithm (Dijkstra , 1959). Actually even if the profit of the arcs may be negative due to the presence of dual variables, for each commodity $k \in K$ the graph $G^k(V^k, A^k)$ is acyclic and it is defined in such a way that an arc $(i, j) \in A^k$ if and only if $i < j$.

Let $\Gamma(i) = \{j | (i, j) \in A^k\}$ be the set of successors of node i. A basic operation in label setting or label correcting algorithms is the *extension* of a label $(\mathcal{T}_i, P_i)$.

It consists in creating new labels at nodes $j \in \Gamma(i)$ by adding arc (i,j) to the path from o^k to i associated with labels $(\mathcal{T}_i, P_i)$. In our case, the new label for a given $j \in \Gamma(i)$ is:

$$g_{ij}(\mathcal{T}_i, P_i) = \begin{cases} (f_{ij}^{kzp}(\mathbf{T}_i^{kzp})\ \forall z \in \{1...Z^k\}, \forall p \in \{1...3\}; f_{ij}^{Q^k}(T_i^{Q^k}), P_i + v_{ij}) & \textbf{if}(feasible(\mathcal{T}_j)) \\ \emptyset & \textbf{otherwise} \end{cases}$$

The *extension* of node i is the extension of all labels in Q_i. The set of new labels at each node $j \in \Gamma(i)$ is $\bigcup_{q \in Q_i} g_{ij}(\mathcal{T}_i^q, P_i^q)$. Since the definition of the resource extension functions, not all new labels will be efficient. Moreover, some new labels may dominate or may be dominated by some labels already in Q_i. Hence, the new set of efficent labels at node j is given by $EFF(\bigcup_{q \in Q_i} g_{ij}(\mathcal{T}_i^q, P_i^q) \cup Q_j)$. For a given commodity $k \in K$, the algorithm can be described as follows:

Initialization

$Q_o^k = \{(\mathcal{T}_{o^k}^1 = (T_{o^k}^{k11}, T_{o^k}^{k12}, T_{o^k}^{k13}, ..., T_{o^k}^{kZ^k1}, T_{o^k}^{kZ^k2}, T_{o^k}^{kZ^k3}, T_{o^k}^{Q^k}, 0),\ P_{o^k}^1 = 0)\};$

$Q_i = \{(\mathcal{T}_i^1 = (b_i^{kzp}\ \forall z \in \{1...Z^k\}, \forall p \in \{1\dots\}; b_i^{Q^k}),\ P_i^1 = -\infty)\};$ $\quad \forall i \in V^k / \{o^k\}$

Extension

for $i = 1$ **to** $|V^k|$
  for each $j \in \Gamma(i)$
    for each $q \in Q_i$
      $Q_j = EFF(g_{ij}(\mathcal{T}_i^q, P_i^q) \cup Q_j);$

The efficiency of the dynamic programming algorithm outlined above heavily relies upon the data structures used to represent the set of labels.

Consider for example the case in which label sets are implemented by using a generic list. For a given $j \in V^k$, to keep Q_j efficient, each time a new feasible label $(\mathcal{T}_j, P_j)$ can be possibly inserted in the set, it is necessary to verify if $(\mathcal{T}_j^q, P_j^q) \succ (\mathcal{T}_j, P_j)$ for all $q \in Q_j$. Moreover if $(\mathcal{T}_j, P_j)$ is Pareto-optimal, all labels must be considered again to check if some of them are dominated. Insertion can thus be very costly with a high number of labels, and most of the times this is the case.

For a given commodity $k \in K$, we implement the label set Q_i associated with each $i \in V^k$ by defining a three-dimensional matrix M of $(m \times n \times p)$ elements, where $m = \lceil \frac{T_{orbit}}{\alpha} \rceil$, $n = \lceil \frac{S_{orbit}}{\beta} \rceil$, $p = \lceil \frac{3T_{orbit}}{\gamma} \rceil$, and α, β, γ are parameters suitably chosen. Each element of the matrix is a doubly linked list in which labels are kept in non-increasing order according to the value of their profit. For a given $q \in Q_i$, the label $(\mathcal{T}_i^q, P_i^q)$ is listed in the matrix element $M[\lceil \frac{T_i^{k11^q}}{\alpha} \rceil][\lceil \frac{T_i^{k12^q}}{\beta} \rceil][\lceil \frac{T_i^{k13^q}}{\gamma} \rceil]$. Moreover with each matrix element $M[x][y][z]$, with $x \in \{1\dots m\}$, $y \in \{1\dots n\}$ and $z \in \{1\dots p\}$, we associate two variables, $\underline{M}[x][y][z]$ and $\overline{M}[x][y][z]$, whose value

is respectively the minimum and the maximum profit associated with the labels in the corresponding list. Thanks to this implementation, when a new label $(\mathcal{T}_j, P_j)$ has to be inserted in Q_j, we know that $(\mathcal{T}_j, P_j)$ can be dominated only by a label listed in a matrix element $M[x][y][z]$ such that $1 \leq x \leq \lceil \frac{T_j^{k11}}{\alpha} \rceil$, $1 \leq y \leq \lceil \frac{T_j^{k12}}{\beta} \rceil$, $1 \leq z \leq \lceil \frac{T_j^{k13}}{\gamma} \rceil$ and $\overline{M}[x][y][z] \geq P_j$. On the other hand we know also that $(\mathcal{T}_j, P_j)$ can dominate only labels listed in a matrix element $M[x][y][z]$ such that $\lceil \frac{T_j^{k11}}{\alpha} \rceil \leq x \leq m$, $\lceil \frac{T_j^{k12}}{\beta} \rceil \leq y \leq n$, $\lceil \frac{T_j^{k13}}{\gamma} \rceil \leq z \leq p$ and $\underline{M}[x][y][z].min \leq P_j$. When we are looking for a label that can dominate $(\mathcal{T}_j, P_j)$, we scan the list associated with a matrix element forward from the head, until the profit of the considered label is less than P_j. In the other case we scan the list backward from the tail, and we stop when the profit of the considered label is greater than P_j.

4.3.2 Solving the Lagrangean dual

We solve the Lagrangean dual by applying the iterative *subgradient* method. As far as problem (4.22) is concerned, at each iteration i this algorithm computes the dual bound $z(\lambda^i, \mu^i)$ by keeping the best value found so far. Then it adjusts the Lagrangean multipliers by moving from the present point (λ^i, μ^i) in the direction opposite to a subgradient of $z(\lambda, \mu)$ at (λ^i, μ^i). In our case, updating is done in the following way:

$$\mu_w^{i+1} = max\{0, \mu_w^i - \Delta_i(1 - \sum_{k \in K} \sum_{\substack{(i,j) \in A_{i,1}^k | \\ w(j) = w}} \bar{x}_{ij}^k)\} \qquad \forall w \in \overline{W}$$

$$\lambda_w^{i+1} = \lambda_w^i - \Delta_i(1 - \sum_{k \in K} \sum_{\substack{(i,j) \in A_{i,1}^k | \\ w(j) = w}} \bar{x}_{ij}^k) \qquad \forall w \in \widehat{W}$$

where $\bar{\mathbf{x}}^k$, with $k \in K$, represents the optimal solution of the Lagrangean problem $P(\lambda^i, \mu^i)$, and Δ_i is the step length.

The algorithm has been implemented as described in (Beasley, 1992), where the rules used to update the step length from iteration to iteration are also explained. In particular, at each iteration i, we compute a lower bound by removing from the optimal solution of $P(\lambda^i, \mu^i)$ all the acquisitions that lead to a violation of constraints (4.2) and (4.3).

4.4 Defining quasi-feasible solutions

To compute quasi-feasible solutions, we consider the problem (4.1)–(4.7), (4.11) taking into account only nominal operation profiles in the SPPRC arising for each commodity $k \in K$ ($[a_i^{kz3},\ b_i^{kz3}]$=[0, 0] for all $z \in \{1 \dots Z^k\}$).

We randomly define a permutation $\pi_K = \{k_1, \ldots, k_K\}$ over the set K. Then, according to this permutation, we consider sequentially the K subproblems.
For a given $k_t \in \pi_K$, we do not take into consideration all arcs $(i,j) \in \cup_{i \in V^{k_t}} A^{k_t}_{i,1}$ such that $w(j)$ is an image already acquired by a commodity k_s with $1 \leq s \leq t$. Then we solve the resulting SPPRC by means of a modified version of the dynamic programming algorithm described in section 4.3.1 that prevents the multiple acquisitions of an image along a path. The difference lies in an additional checking during the extension of a label. The creation of a new label at node $j \in \Gamma(i)$, by adding arc $(i,j) \in A^{k_t}_{i,1}$ to the path $\mathcal{P}_i = (i_0, i_1, \ldots, i_H)$ associated with the label $(\mathcal{T}_i, P_i)$, is prevented if $\exists h, 1 \leq h \leq H \mid w(i_h) = w(j) \wedge (i_{h-1}, i_h) \in A^{k_t}_{i_{h-1},1}$.
At the end of this process we obtain a feasible solution for the problem (4.1)–(4.7), (4.11): a path $\mathcal{P}^{(k_t)}$ for each $k_t \in \pi_K$.

Then, we apply a post-processing phase to exploit for each satellite the possibility to perform a peak orbit and a triplet of peak orbit.
For a satellite $k_t \in \pi_K$ we identify first the peak orbit. For each acquired DTO i_j in $\mathcal{P}^{(k_t)} = (i_0, i_1, \ldots, i_H)$ we consider a time window one orbit large starting at a_{i_j} and we evaluate the increase in the solution value that can be obtained by performing new acquisitions within the time window. The new acquisitions are selected by considering sequentially all DTOs that can be taken by the satellite during the orbit: a DTO can be taken if it is compatible with those already listed in $\mathcal{P}^{(k_t)}$, if it does not lead to violations of the peak profile constraints and if it is associated with an image not yet acquired. The orbit associated with the best increase becomes the peak orbit and $\mathcal{P}^{(k_t)}$ is modified accordingly.
In a similar way we next identify the triplet of peak orbits not overlapping with the peak orbit.

The solution finally obtained identifies a subset of DTOs to acquire and it can be used as a guide for a heuristic. It is quasi-feasible in the sense that the constraints not explicitly taken into account can be still violated.

4.5 Computational results

The solution methodology proposed has been tested on 5 problem instances given by Space Software Italia. Each of them concerns 4 satellites performing about 15 orbits in a time horizon of one day. In these instances, none of the considered requests is of high priority or splitted and with each image is associated a profit equal to 1 (i.e. the objective is to maximize the number of taken images). In table 4.1 we summarize the main characteristics of the instances. $|W|$ is the cardinality of the image set (one image for each request coming from the user). The remaining three columns indicate respectively the percentage of SPOTLIGHT1, SPOTLIGHT2 and WIDEFIELD images in the set.

Table 4.1: Test instances

Instance	**|W|**	**SP1**	**SP2**	**WF**
1	1327	0	1064	263
2	1000	119	130	751
3	2000	74	295	1631
4	1000	95	103	802
5	1600	153	166	1281

Table 4.2: Commodity network size

Instance	$\|\overline{\mathbf{V}}\|$	$\|\overline{\mathbf{A}}\|$
1	1573.50	158476.50
2	1192.75	220034.75
3	3977.75	544543.50
4	2206.00	243825.00
5	3432.25	587219.75

For all the instances, the number of requests is greater than 1000, with a maximum of considered requests equal to 2000. Except for the first instance where there are no SPOTLIGHT1 images and the SPOTLIGHT2 images are 80% of the total, in average SPOTLIGHT1 and SPOTLIGHT2 images represent together 21% of the considered images. The average size of the commodity networks generated from each instance is listed in table 4.2. $|\overline{V}|$ is the average number of nodes, whereas $|\overline{A}|$ represents the average number of arcs.

Experiments have been done on a 1.60GHz Pentium 4 machine. For each instance we have found dual bounds by solving different relaxations. We define $P^{(1)}(\lambda, \mu)$ by considering a set of resources that includes only the memory. Then, taking into account only operational profile resources for a number of orbit sequences equal to 1 for each commodity, we define $P^{(2)}(\lambda, \mu)$. Finally, starting from this latter relaxation, we derive $P^{(3)}(\lambda, \mu)$, where we allow each discrete orbit to be a peak orbit, and $P^{(4)}(\lambda, \mu)$, where WIDEFIELD taken images are combined linearly with SPOTLIGHT1 and SPOTLIGHT2 acquisitions in evaluating nominal profiles. In $P^{(3)}(\lambda, \mu)$, for each commodity $k \in K$, the resource set $P^{k1} = \{P_1^{k1}\}$ and $[a_i^{k11},\ b_i^{k11}] = [0,\ 4T_{orbit}]$ for each $i \in V^k$. Whereas in $P^{(4)}(\lambda, \mu)$, $P^{k1} = \{P_1^{k1}, P_3^{k1}\}$ and $[a_i^{k11},\ b_i^{k11}]$, $[a_i^{k13},\ b_i^{k13}]$ are equal to $[0,\ 2T_{orbit}]$, $[0,\ 3T_{orbit}]$

Table 4.3: Dual bounds

Instance	$z_{LD}^{(1)}$	$z_{LD}^{(2)}$	$z_{LD}^{(3)}$	$z_{LD}^{(4)}$
1	898.504	**673.941**	895.476	954.490
2	703.504	876.619	**701.398**	821.559
3	2472.972	-	**1996.071**	-
4	1017.995	1456.481	**1002.718**	1505.500
5	2085.116	-	**1589.840**	2164.000

for each $i \in V^k$. In both relaxations, SPOTLIGHT1 and SPOTLIGHT2 acquisitions are converted in time consumptions through a constant factor. We solve the Lagrangean dual (4.22) associated with the four mentioned relaxations with a maximum CPU time of 6 hours. For each instance, the dual bound $z_{LD}^{(i)}$, with $i = 1 \dots 4$, is reported in table 4.3 and the best dual bound is bolded. In particular, the values in the columns $z_{LD}^{(2)}$ and $z_{LD}^{(4)}$ are already increased by the needed constant factor (see section 4.2.4) and are valid dual bounds for the MORCP.

The dual bounds found by using operational profiles resources are always better than those found by considering the memory resource: the percentage of improvement ranges from 0.14% to 33.32%. This shows how the operational profile constraints are the most conservative. Increasing the cardinality of the resource set, the number of labels per node evaluated by the dynamic programming algorithm increases exponentially. Thus more time is required to solve the SPPRC associated with each commodity and fewer iterations can be performed by the subgradient algorithm within 6 hours. In particular, due to the size of the considered instances, only for the relaxation $P^{(3)}(\lambda, \mu)$ we have been able to approximate the optimal value of the associated Lagrangean dual. This is why the value listed in $z_{LD}^{(3)}$ are in general better than those reported in $z_{LD}^{(2)}$ and $z_{LD}^{(4)}$.

We have tried to improve the value obtained for the relaxation $P^{(3)}(\lambda, \mu)$ by considering a number of orbit sequences equal to 2 and 3. Results are reported in table 4.4.

Not surprisingly, only for the smallest instances we have been able to obtain slightly better dual bounds. Actually a greater number of orbits give rise to an increase in the number of resources to consider, with the drawbacks previously discussed.

Then we have computed primal bounds by guiding the Algorithm 4 described at the end of the previous chapter by means of quasi-feasible solutions. For each instance, we report in column $z^{(G)}$ of table 4.5 the results computed giving in input to the algorithm the subset of DTOs selected in the quasi-feasible solution found as described in section 4.4. Column $z^{(S)}$ contains the results computed by the

Table 4.4: Dual bounds, $z_{LD}^{(3)}$, multiple orbit sequences

Instance	$Z^k = 1\ \forall k \in K$	$Z^k = 2\ \forall k \in K$	$Z^k = 3\ \forall k \in K$
1	895.476	894.793	**892.807**
2	701.398	701.471	**700.700**
3	**1996.071**	2328.281	3022.500
4	**1002.718**	1214.509	1649.000
5	**1589.840**	2261.000	2260.000

Table 4.5: Primal bounds

Instance	$\underline{z}^{(G)}$	$\underline{z}^{(S)}$	gap(%)
1	548	513	6.82
2	644	588	9.52
3	1384	1506	-8.10
4	783	822	-4.74
5	1033	1034	-0.09

algorithm from scratch. In the last column of the table we report the gap between the two values.

For the smallest instances, 1 and 2, we improve the solution respectively of 6.82% and 9.52%, whereas for the other instances the quality of the solutions computed from scratch is better. This suggests that the choice of the acquisitions to schedule becomes less important proportionately to the increase of the number of possible acquisitions. Since the time needed to compute the quasi-feasible solutions for instances 1 and 2 is less than 2 minutes, the solution methodology proposed represents a valid alternative to solve instances of the MORCP that are relatively small.

Table 4.6: Bounds comparison

Instance	$\underline{z}^*$	$\overline{z}^*$	gap(%)
1	548	673.941	18.69
2	644	700.700	8.09
3	1506	1996.071	24.55
4	822	1002.718	18.02
5	1039	1589.840	34.65

Finally in table 4.6 we report for each instance the best values found for primal and dual bounds. We recall that, due to the size of the considered problem instances, only for the relaxation $P^{(3)}(\lambda, \mu)$ we have been able to approximate the optimal value of the associated Lagrangean dual by means of the subgradient algorithm. This relaxation allows each discrete orbit to be a peak orbit. Thus, with respect to the real operational profile constraints, the additional time available increases by about a factor equal to 3. This is why for the biggest instance, the dual bounds computed do not represent a significant term of comparison for the corresponding heuristic solution value.

4.6 Conclusions

In this chapter we have considered the MORCP for a planning horizon of 1 day. To find dual bounds, we have modelled a relaxed version of the problem through a multy-commodity flow formulation. Then we have applied to this last the Lagrangean relaxation approach. The problem features that have the greatest impact on the solution quality, that is acquisition and transmission activities, as well as energy consumption, have been modelled in the relaxed problem by means of resources. In particular, nominal operational profiles constraints are checked at a given time for a discrete number of orbits and the model has been parameterized according to this number. A SPPRC can be identified in the formulation for each commodity and by applying the decomposition approach each subproblem can be solved indipendently. To this purpose, we have efficiently implemented a label setting algorithm. Dual bounds have been computed by applying the subgradient method to the Lagrangean dual. Finally, starting from this approach, we have illustrated how quasi-feasible solutions can be obtained to guide heuristics previously defined in finding better solutions than those presented from scratch.

Experimental results have been obtained for five problem instances given by Space Software Italia. The biggest instance considers 2000 requests; the corresponding subproblems have been solved on a graph involving about 4000 nodes and 500000 arcs. We have computed valid dual bounds with different settings; the results found show how the operational profiles constraints are the most conservative. Moreover, for the smallest instances, guiding the Algorithm 4 described at the end of the previous chapter by means of quasi-feasible solutions, we have been able to improve in average the solution value of 8.17%. Nevertheless, due to the size of the problem instances considered, some of the dual bounds computed do not represent a significant term of comparison for the corresponding heuristic solution value.

Chapter 5

Conclusions

In this thesis, by means of Operations Research techniques, we have tackled two planning and scheduling problems arising for AEOS: the *Multi-Orbit Optical Constellation Problem* (MOOCP) and the *Multi-Orbit Radar Constellation Problem* (MORCP). The MOOCP concerns the management of the scientific activities for the French *PLEIADES* constellation of optical satellites, whereas the MORCP regards the management of four satellites equipped with SAR instruments with respect to the payload utilization: the Italian *COSMO-SkyMed* constellation is the reference scenario. For both problems we have dealt with the richest model presented in the literature so far. In particular in the MORCP we have considered all the real problem details specified by Space Software Italia.

Bianchessi et. al (2005) defined a tabu search heuristic to solve the MOOCP with Satellites Sharing that can be applied also to our MOOCP. Then, as far as this problem is concerned, in this thesis we have focused on the definition of an upper bounding procedure with the aim to find tight dual bounds. On the other hand we have defined for the MORCP a procedure to find dual bounds as well as heuristic algorithms.

The heuristics defined for the MORCP are greedy constructive (randomized) algorithms. They compute feasible solutions with respect to all technical constraints and try to satisfy constraints about high priority request acquisitions by means of backtracking and look-ahead capabilities.

As far as dual bounds are considered, to compute them we have modelled the two problems through a multi-commodity flow formulation inspired from the unified framework for vehicle routing and scheduling problems described by Desaulniers et al. (1998). Optical satellites can acquire images only during their enlightened revolution, thus we have identified a commodity for each orbit while modelling the MOOCP; on the other hand, a relaxed MORCP has been formulated by defining a commodity for each satellite. Then we have solved the two problems using equivalent primal and dual decomposition approaches: Dantzig-Wolfe de-

composition approach (Dantzig and Wolfe, 1960) and Lagrangean relaxation (Geoffrion , 1974). These approaches allow to find the same dual bound for a given problem. Nevertheless, solving approximately the Lagrangean dual by means of a subgradient algorithm allows a faster convergence to the best dual bound value with no guarantee of optimality. Since the subproblems arising in the relaxed MORCP requires much more time to be solved with respect to those arising in the MOOCP, we have applied Lagrangean relaxation to the former problem and Dantzig-Wolfe decomposition to the latter. Bounds obtained for large scale MOOCP test instances are at most 3.09% greater than heuristic solution values. On the other hand, due to the size of the MORCP instances considered, the dual bounds computed for them do not represent a significant term of comparison for the corresponding heuristic solution value. Nevertheless, for the smallest MORCP instances, starting from the approach used to compute dual bounds, we have illustrated how quasi-feasible solutions can be obtained to guide the greedy algorithms in finding better solutions than those presented from scratch. This suggests that we can successfully apply the same mechanism to the bigger instances by defining quasi-feasible solutions by means, for example, of sophisticated local search based heuristics, i.e. tabu search heuristics.

The work of Bianchessi et. al (2005) is the only one presented in the literature so far concerning the specific problems considered in this thesis. Thus, further comparisons about the results obtained are not possible.

Finally, focusing on future developments, apart from the main objective function to be optimized, it may be convenient to take into account secondary objective functions such as response time, balance in the use of the ground stations, and others. In such case, an optimization algorithm is not enough; rather a *decision support system* is recommendable, allowing for a trade-off analysis among non-dominated (Pareto-optimal) solutions, according to dynamically adjustable criteria that can vary according to the specific needs or operating conditions of the moment. Moreover, besides optimizing the operations of a system, in the respect of technological constraints, Operations Research can also be used to optimize the management of a system. *Revenue management* is concerned with the maximization of the revenue that can be obtained when selling a product or a service whose availability has already been established, so that costs are more or less fixed. A typical example of successful application of revenue management is that of flights pricing: the same seat on board of the same airplane on the same leg has a cost that can vary considerably over time, depending on a number of parameters such as the number of passengers already booked on the same flight, the advance with which the seat is reserved and many others. This allows airline companies to reduce the number of empty seats, that are sold at a very low price, as well as to exploit at the maximum extent the service provided in highly requested days to

highly requested destinations. In the case of the problems arising for AEOS, the negotation with the customers, the definition of a price for each image, the management of users' quotae are very important aspects that need to be empowered with revenue management tools. This is especially important in consideration of the very high fixed cost necessary to build and operate a satellite constellation.

Bibliography

R. Ahuja, T. Magnanti and J. Orlin. "Network Flows: Theory, Algorithms, and Applications", Prentice Hall, Englewood Cliffs, New Jersey (1993).

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, The Netherlands (1999).

J.E. Beasley. "Lagrangean Relaxation." *Technical report*, The Management School, Imperial College, London (May, 1992).

E. Bensana, G. Verfaillie, J.C. Agnèse, N. Bataille and D. Blumstein. "Exact and approximate methods for the daily management of an Earth observation satellite." In *Proceedings of the 4th International Symposium on Space Mission Operations and Ground Data System (SpaceOps-96)*, Munich, Germany (1996).

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

N. Bianchessi, J.-F Cordeau, J. Desrosiers, G. Laporte and V. Raymond. "A Heuristic for the Management of Multiple-Orbit Earth Observation Satellites." *Technical Report GERAD-2005-45*, HEC Montréal (2005).

J.-F. Cordeau and G. Laporte. "Maximizing the Value of an Earth Observation Satellite Orbit." *Journal of the Operational Research Society*, (2005). To appear.

G.B. Dantzig and P. Wolfe. "Decomposition Principle for Linear Programs." *Operations Research*, **8**:101–111 (1960)

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.

J. Desrosiers, Y. Dumas, M.M. Solomon and F. Soumis "Time Constrained Routing and Scheduling.", In Network Routing, M.O. Ball et al. editors, *Handbooks in Operations Research and Management Science*, vol. 8, pages 35–139. Elsevier Science B.V., 1995.

E. Dijkstra. "A note on two problems in connection with graphs." *Numer. Math.*, **1**:269–271 (1959)

J. Frank, A. Jonsson, R. Morris and D.E. Smith. "Planning and scheduling for fleets of Earth observing satellites." NASA Ames Research Center (2001).

V. Gabrel, A. Moulet, C. Murat and V.T. Paschos. "A new single model and derived algorithms for the satellite shot planning problem using graph theory concepts." *Annals of Operations Research* **69**:115–134 (1997)

V. Gabrel and C. Murat. "Mathematical programming for Earth observation satellite mission planning." In T.A. Ciriani, G. Fasano, S. Gliozzi and R. Tadei, editors, *Operations Research in Space and Air*, pages 103–122, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003.

M.R. Garey and D.S. Johnson. "Computers and Intractability, a Guide to the Theory of NP-Completeness." Freeman, 1979.

A.M. Geoffrion. "Lagrangian Relaxation and its Uses in Linear Programming." *Mathematical Programming Study*, **2**:82–112 (1974).

A. Globus, J. Crawford, J. Lohn and A. Pryor. "Scheduling Earth observing satellites with evolutionary algorithms." *International Conference on Space Mission Challenges for Information Technology*, Pasadena, California, (2003).

N.G. Hall and M.J. Magazine. "Maximizing the value of a space mission." *European Journal of Operations Research*, **78**:224–241 (1994).

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).

E.J. Kuipers. "Algorithm for the management of the missions of Earth observation satellites." *Booklet of abstracts*, Challenge ROADEF 2003.

M. Lemaître, G. Verfaillie and N. Bataille "Exploiting a common property resource under a fairness constraint: A case study." In *Proc. 18th IJCAI-99*, pages 206–211, Stockholm, Sweden (1999).

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, Japan. Morgan Kaufmann (1997).

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, E. Bensana, C. Michelon-Edery, and N. Bataille. "Dealing with uncertainty when managing and Earth observation satellite." In *Proceedings of the 5th International Symposium on Artificial Intelligence, Robotics and Automation for Space*, pages 205–207, Noordwijk, The Netherlands (1999).

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.

G. Verfaillie, M. Lemaître, and T. Schiex. " Russian doll search for solving constraints optimization problems." In *Proc. AAAI-96*, pages 181–187, Portland, OR (1996).

T. Schiex, H. Fargier and G. Verfaillie. "Valued constraint satisfaction problems: hard and easy problems." In *Proc. IJCAI-95*, pages 631–639, Montréal, Québec (1995).

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