Thèse

Préparée au
Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS

en vue de l'obtention du grade de
Docteure de l'Institut National des Sciences Appliquées de Toulouse

Spécialité : **Systèmes Industriels**

par

Catherine Mancel

Ingénieure ISIMA

MODÉLISATION ET RÉSOLUTION DE PROBLÈMES D'OPTIMISATION COMBINATOIRE ISSUS D'APPLICATIONS SPATIALES

Soutenue le 25 juin 2004 devant le jury :

Présidente	**C. MERCÉ**
Rapporteurs	**P. MAHEY** **P. MICHELON**
Examinateur	**N. BATAILLE**
Directeurs de thèse	**P. LOPEZ** **R. VALETTE**
Invité	**J.-C. HOCHON**

Avant-propos

Le travail présenté dans ce mémoire a été réalisé au Laboratoire d'Automatique et d'Analyse des Systèmes (LAAS) du CNRS, dans le cadre d'une Convention Industrielle de Formation par la Recherche. Je remercie Messieurs Jean-Claude Laprie et Malik Ghallab, directeurs succesifs du laboratoire pendant mon séjour au LAAS, de m'avoir accueillie dans leur structure. Je remercie également Monsieur Christophe Lansade, alors Directeur Régional de la société IXI, devenue par la suite GFI-Consulting, de m'avoir donné la possibilité d'engager cette thèse CIFRE.

Pour l'honneur qu'il me fait d'avoir accepté d'être rapporteur de ma thèse et avant cela, pour m'avoir enseigné mes premiers cours de Recherche Opérationnelle à l'Institut Supérieur d'Informatique, de Modélisation et leurs Applications (ISIMA), je remercie Monsieur Philippe Mahey, Professeur à l'Université de Clermont-Ferrand. Je suis très reconnaissante à Monsieur Philippe Michelon, Professeur à l'Université d'Avignon et des Pays de Vaucluse, qui m'a également fait l'honneur d'étudier mes travaux et d'en être rapporteur.

J'exprime ma gratitude à Madame Colette Mercé, Professeure à l'Institut National des Sciences Appliquées (INSA) de Toulouse, pour avoir présidé le jury de cette thèse et pour sa lecture attentive de mon manuscrit. Je remercie très sincèrement Monsieur Nicolas Bataille, Ingénieur au Centre National des Études Spatiales (CNES), pour sa participation au jury, mais aussi pour l'intérêt qu'il a porté à mes travaux depuis mon DEA et tout au long de ma thèse, et pour l'aide qu'il m'a apportée dans l'étude de différentes problématiques liées au domaine spatial.

Je tiens à remercier Monsieur Jean-Claude Hochon, mon responsable industriel à GFI-Consulting pendant ces années de thèse, qui, tout en me laissant une grande liberté de travail, s'est toujours montré disponible pour répondre à mes questions.

Je veux exprimer ici mon extrême reconnaissance envers mes deux directeurs de thèse, Monsieur Pierre Lopez, Chargé de Recherche au LAAS et Monsieur Robert Valette, Directeur de Recherche au LAAS. Leur compétence, les conseils et critiques dont ils m'ont fait bénéficier, ainsi que leur soutien constant et la disponibilité exemplaire dont ils ont fait preuve à mon égard, pendant toute la durée de ma thèse, m'ont permis de mener ces travaux à leur terme. Je remercie Robert pour sa gentillesse au quotidien et pour s'être toujours attaché à suivre au plus près mon travail, qui pourtant s'est très vite éloigné de ses domaines de prédilection. Je remercie Pierre de m'avoir donné goût à ses domaines de recherche et à son métier lors de mon premier stage au LAAS, puis d'avoir continué à m'encadrer dans ce climat de travail appréciable, à la fois rigoureux et détendu. Notre collaboration a été, pour moi, très formatrice et je souhaite qu'elle puisse se poursuivre encore longtemps.

Je n'oublie pas mes collègues et amis du LAAS, qui contribuent par leur bonne humeur, leur

sympathie, par leur compétence également, à faire de ce lieu un cadre privilégié de travail et d'échanges. J'adresse une pensée particulière aux doctorantes (et docteure) d'*ex*-OCSD, avec en tête, Emmanuelle et Stéphanie, dont le soutien moral et logistique, tout particulièrement dans les derniers moments, m'a été très précieux.

Je remercie enfin, pour l'accueil chaleureux qu'ils m'ont réservé, les membres du Laboratoire d'Informatique d'Avignon (LIA), où j'ai passé mes derniers mois de thèse en tant qu'ATER. Je remercie en particulier mes collègues chercheurs opérationnels et optimiseurs, notamment Dominique Feillet qui a consacré du temps à la lecture de certaines parties de mon manuscrit, ainsi que Christian Artigues et Cristian Oliva, dont la présence amicale dans les périodes difficiles de rédaction m'a beaucoup aidée.

Table des matières

Introduction **9**

I Applications spatiales et optimisation combinatoire **13**

1 Problèmes d'optimisation combinatoire dans les applications spatiales **15**
1.1 Introduction . . . 15
1.2 Problèmes classiques d'optimisation combinatoire . . . 15
1.2.1 Problème du sac-à-dos . . . 16
1.2.2 Problème d'affectation . . . 17
1.2.3 Problème du voyageur de commerce . . . 18
1.2.4 Problème d'ordonnancement . . . 19
1.3 Problèmes d'optimisation combinatoire issus d'applications spatiales . . . 21
1.3.1 Spécificité des problèmes posés . . . 21
1.3.2 Modélisation et résolution : revue des méthodes utilisées . . . 23
1.3.2.1 Heuristiques et algorithmes dédiés . . . 23
1.3.2.2 Approches par contraintes, méthodes d'Intelligence Artificielle 25
1.3.2.3 Méthodes exactes . . . 26
1.3.2.4 Approches par simulation : utilisation des Réseaux de Petri . . 27
1.4 Conclusion . . . 28

2 Techniques de résolution retenues **31**
2.1 Introduction . . . 31
2.2 Propagation de contraintes . . . 31
2.2.1 Propagation de contraintes temporelles . . . 32
2.2.1.1 Problèmes temporels simples . . . 32
2.2.1.2 Problèmes temporels généraux . . . 33
2.2.2 Propagation de contraintes de partage de ressources . . . 33
2.2.2.1 Opérations locales, opérations globales . . . 34
2.2.2.2 Raisonnement énergétique . . . 34
2.3 Programmation linéaire . . . 37

2.3.1 Définitions . . . 37
2.3.2 Résolution des programmes linéaires . . . 39
2.3.2.1 Principaux algorithmes . . . 39
2.3.2.2 Méthode du simplexe . . . 40
2.3.3 Programmation linéaire en nombres entiers . . . 41
2.4 Génération de colonnes pour les problèmes de grande dimension . . . 44
2.4.1 Principe de résolution par génération de colonnes . . . 44
2.4.2 Convergence de la méthode . . . 46
2.4.3 Décomposition de Dantzig-Wolfe (1960) . . . 47
2.4.4 Résolution de PLNE par génération de colonnes . . . 50
2.4.4.1 Décomposition des PLNE . . . 50
2.4.4.2 Algorithme générateur . . . 50
2.4.4.3 Recherche de solutions entières : “branch and price” . . . 51
2.5 Conclusion . . . 52

II Étude de trois projets spatiaux : analyse et modélisation des problèmes de planification 55

3 Skybridge : une constellation basse altitude pour la communication 57
3.1 Présentation du projet Skybridge . . . 57
3.1.1 Constellations basse altitude de télécommunication . . . 57
3.1.2 Architecture de la constellation Skybridge . . . 58
3.2 Problématique . . . 59
3.3 Analyse du problème . . . 61
3.4 Modélisation du problème . . . 62
3.5 Travaux connexes . . . 65
3.6 Bilan de l’étude sur Skybridge . . . 65

4 Netlander : projet d’exploration de la planète MARS 67
4.1 Présentation de la mission . . . 67
4.2 Problématique . . . 68
4.3 Analyse du problème . . . 71
4.3.1 Revue de la littérature . . . 71
4.3.2 Proposition d’une décomposition . . . 72
4.3.3 Sous-problème de planification des communications orbiteur/sondes . . . 74
4.3.3.1 Analyse du problème . . . 74
4.3.3.2 Modélisation . . . 75
4.3.4 Sous-problème de planification des expériences . . . 77
4.3.4.1 Analyse du problème . . . 77

4.3.4.2 Modélisation 79
4.4 Conclusion 80

5 Pléiades : Satellites d'observation de la Terre 81
5.1 Présentation du projet 81
5.2 Problématique 83
5.3 Analyse du problème 86
5.3.1 Revue de la littérature 86
5.3.2 Modélisation 87
5.4 Étude de la complexité du problème Pléiades 91
5.5 Conclusion 92

III Résolution de problèmes de planification dans les projets Netlander et Pléiades 93

6 Planification de communications et d'expériences dans le projet Netlander 95
6.1 Introduction 95
6.2 Planification des communications orbiteur/sondes 95
6.2.1 Description des orbites testées et des outils utilisés 96
6.2.2 Résultats du modèle initial 97
6.2.3 Généralisation du problème 99
6.2.4 Conclusion 101
6.3 Planification des expériences 101
6.3.1 Courbe de charge de la mémoire de masse d'une sonde 102
6.3.1.1 La mémoire de masse d'une sonde 102
6.3.1.2 Élaboration de la courbe de charge de la mémoire de masse . . 103
6.3.1.3 Analyse des courbes, règles de déduction 108
6.3.2 Courbe de consommation de l'énergie électrique d'une sonde 109
6.3.2.1 L'énergie électrique d'une sonde 109
6.3.2.2 Élaboration de la courbe d'utilisation de l'énergie électrique . . 109
6.3.2.3 Analyse des courbes, règles de déduction 115
6.3.3 Exemple 117
6.3.4 Conclusion 122

7 Planification de prises de vues dans le projet Pléiades 129
7.1 Introduction 129
7.2 Positionnement bibliographique 129
7.3 Décomposition du problème 131
7.4 Description du problème de génération des pseudo-séquences de prises de vue . 133

7.5 Algorithme de génération de colonnes pour le problème Pléiades 135
7.5.1 Génération des pseudo-séquences de prises de vue 135
7.5.1.1 Chemins élémentaires, algorithme PLPSE 135
7.5.1.2 Chemins avec cycles autorisés, algorithme PLPSC 140
7.5.2 Résolution de la relaxation linéaire de PM 141
7.5.2.1 Fonction `Construction_Graphe`(SP) 142
7.5.2.2 Construction de Ω_0 143
7.6 Expérimentations 144
7.6.1 Modèle linéaire initial 144
7.6.1.1 Calcul d'une valeur de N_{sup} 144
7.6.1.2 Bornes supérieures obtenues par programmation linéaire 145
7.6.2 Génération de séquences solutions de prises de vue 146
7.6.3 Procédure de génération de colonnes 147
7.6.3.1 Prétraitement des données : réduction du graphe G 147
7.6.3.2 Procédure GC(Ω_0) 148
7.7 Conclusions et perspectives 149

Conclusion **151**

Bibliographie **154**

Introduction

Initialisées avec le lancement du satellite Spoutnik en 1957, les missions spatiales furent d'abord développées dans un but d'exploration de l'Univers et pour le rayonnement scientifique et militaire des principales puissances mondiales de l'après-guerre. L'étude de la Terre et de l'Espace fait aujourd'hui encore l'objet de nombreuses missions scientifiques par l'envoi de satellites, sondes ou vols habités. Parallèlement à cela, les progrès technologiques rapides et l'apparition de besoins nouveaux dans les domaines des télécommunications et de l'observation de la Terre ont conduit, essentiellement à partir des années 80, au développement d'un autre type de missions dont le but principal est de proposer des services (communication de différents types d'informations sur toute la planète, localisation pour la navigation, surveillance de la Terre, cartographie, etc.).

Dans tous les cas, ces missions spatiales, à vocation scientifique, militaire ou commerciale, mettent en jeu des systèmes d'une grande complexité technologique et nécessitent toujours d'importants investissements en termes de moyens, d'études scientifiques et de temps de travail. Ce sont ainsi des projets de grande envergure, qui posent des problèmes d'optimisation des décisions dans de nombreux domaines (pour la définition du système, la mise et le maintien à poste, la planification des opérations de la mission).

En particulier, des problèmes combinatoires complexes relatifs à la gestion des ressources du système – telles que l'énergie, la mémoire, les antennes de communication, divers instruments de mesure – émergent lors des phases de planification d'une mission spatiale, avec un niveau croissant d'exigence sur la fiabilité et l'optimisation des résultats. Ces problèmes présentent des caractéristiques communes en termes de problématiques et de contraintes, liées en particulier à une forte limitation des ressources disponibles et du temps et à la dynamique des systèmes satellitaires. De plus en plus, les approches de résolution de ces problèmes font appel aux méthodes et techniques issues des domaines de la Recherche Opérationnelle et de l'Intelligence Artificielle, avec, en général, le développement d'heuristiques pour fournir des solutions.

Dans cette thèse, nous nous intéressons à la modélisation et à la résolution de ces problèmes d'optimisation combinatoire issus de la planification de missions spatiales. Nous étudions les apports de méthodes exactes issues de la Recherche Opérationnelle – en particulier la Programmation Linéaire – et de techniques de réduction des espaces de recherche – par Propagation de Contraintes par exemple –, pour la résolution de ces problèmes.

Les travaux présentés dans ce manuscrit ont été réalisés dans le cadre d'une Convention Industrielle de Formation par la Recherche entre la société GFI-Consulting et le groupe MO-

GISA [1] du LAAS-CNRS [2] et en collaboration avec le CNES [3].

Cette collaboration a débuté en juin 2000, au cours du stage de DEA, qui portait sur le projet Skybridge (constellation de satellites basse altitude pour la communication) mené par Alcatel Space Industries et auquel participait le CNES. Ces premiers travaux ont permis principalement d'établir une comparaison de diverses techniques d'optimisation envisageables pour un problème d'allocation de liens de communication et ont motivé la poursuite de la collaboration dans la voie de recherche d'outils efficaces pour la résolution de différents problèmes de gestion des ressources dans les sytèmes spatiaux.

Suite au gel du projet Skybridge fin 2000 et à l'abandon de plusieurs autres constellations satellitaires de communication à cette même époque, notre travail s'est porté sur des problèmes issus de deux autres projets développés au CNES : Netlander, qui a pour but l'exploration de Mars et Pléiades, pour l'observation de la Terre par satellites.

Le projet Netlander pose un problème de planification d'expériences sur un ensemble de sondes posées à la surface de Mars et de planification des communications entre les sondes et un orbiteur martien. Le projet Pléiades pose un problème de planification des prises de vue que doit réaliser un satellite "agile" d'observation de la Terre. Notons que ce dernier problème a également été proposé comme sujet du challenge ROADEF'03 [4].

Le manuscrit est divisé en trois parties. La partie I pose le cadre de travail en proposant un tour d'horizon des problèmes d'optimisation combinatoire qui émergent dans les missions spatiales et en décrivant les principaux concepts et méthodes utilisés dans la suite des travaux : la Programmation Linéaire et les techniques de Propagation de Contraintes. Cette partie fait l'objet de deux chapitres. Le chapitre 1 introduit d'abord la problématique de l'optimisation combinatoire en présentant, du point de vue de leur modélisation et de leur résolution, quatre problèmes classiquement étudiés en Recherche Opérationelle : un problème de sac-à-dos, un problème d'affectation, un problème de voyageur de commerce et un problème d'ordonnancement. Dans ce chapitre, nous caractérisons ensuite les problèmes d'optimisation combinatoire spécifiques aux applications spatiales, en nous appuyant sur les points communs qui existent dans les problématiques et les contraintes de ces problèmes. Nous dressons un bref état de l'art des approches utilisées pour résoudre ces problèmes. Les méthodes développées puisent essentiellement dans les concepts de la Recherche Opérationnelle et de l'Intelligence Artificielle. Si en pratique les heuristiques sont le plus souvent utilisées pour la recherche de solutions, une approche par des méthodes exactes, en particulier la Programmation Linéaire, présente plusieurs intérêts, en dehors d'une résolution optimale de certains problèmes. Notons que nous complétons cet état de l'art au cours de la partie suivante, lorsque nous revenons plus en détail sur certains problèmes. Le chapitre 2 décrit les concepts de base de la Propagation de Contraintes, ainsi que les principes de la Programmation Linéaire, de la Programmation Linéaire en Nombres Entiers et de la Génération de Colonnes. Ce sont les méthodes que nous avons privilégiées par la suite pour l'étude des différents problèmes d'optimisation soumis par le CNES. Les techniques de Propagation de Contraintes permettent de réduire l'espace de recherche des solutions d'un problème. Couplées à l'utilisation d'une méthode exacte comme la Programmation Linéaire ou ses extensions aux problèmes en variables entières (PLNE) ou aux problèmes de grande dimension (GC), elle peuvent permettre de renforcer l'efficacité de

1. Modélisation, Optimisation et Gestion Intégrée de Systèmes d'Activités
2. Laboratoire d'Analyse et d'Architecture des Systèmes – Centre National de la Recherche Scientifique
3. Centre National d'Études Spatiales
4. ROADEF : Société Française de Recherche Opérationnelle et d'Aide à la Décision

la résolution.

La partie II présente une analyse des problématiques issues des trois projets spatiaux que nous avons étudiés et propose pour chacune un modèle mathématique. Elle est composée des chapitres 3 à 5. Le chapitre 3 est consacré à l'étude d'un problème d'allocation des créneaux de communication dans la constellation Skybridge. La particularité de ce problème est essentiellement due à la contrainte liée aux "handovers" qui consistent à transférer la prise en charge d'une communication d'un satellite à un autre, sans interrompre la communication. En effectuant des hypothèses simplificatrices sur la demande de service, nous proposons un modèle linéaire en variables mixtes pour ce problème. Nous terminons ce chapitre en présentant quelques résultats de validation du modèle. Du fait de l'abandon du projet Skybridge, l'étude de ce problème n'a pas été poursuivie et nous n'y revenons plus dans la suite du manuscrit. Dans le chapitre 4, nous présentons le projet d'exploration martienne, Netlander. Le système considéré est essentiellement composé de quatres sondes posées à la surface de Mars et d'un orbiteur martien qui relaie les communications avec la Terre. Dans ce projet, un problème d'"optimisation des boucles de programmation" nous a été soumis par le CNES. L'analyse de cette problématique nous conduit à définir un problème mixte de planification de communications entre les sondes martiennes et l'orbiteur, et de planification des expériences des sondes. Nous proposons une décomposition en deux sous-problèmes. Nous modélisons le sous-problème de planification des communications orbiteur/sondes par un programme linéaire en nombres entiers. Le sous-problème de planification des expériences des sondes est mal défini et se pose plutôt comme un problème d'aide à la décision. Le chapitre 5 concerne le projet Pléiades. Il s'agit du programme d'observation de la Terre du CNES destiné à remplacer les satellites SPOT actuellement en place. Du fait des innovations technologiques apportées au satellite Pléiades par rapport à ses prédécesseurs et qui en font un satellite "agile", il se pose un nouveau problème de planification des prises de vue, complexe et très fortement combinatoire. En posant une hypothèse simplificatrice sur la définition du critère d'optimisation, nous proposons un modèle linéaire en variables mixtes de ce problème.

Enfin, la partie III qui regroupe les sixième et septième chapitres, expose les méthodes développées pour la résolution des problèmes issus des projets Netlander et Pléiades. Dans le chapitre 6, nous résolvons le problème de planification des communications orbiteur/sondes par PLNE, sur des instances d'orbites fournies par le CNES comme orbites candidates pour le projet Netlander. Nous proposons ensuite une approche d'aide à la planification des expériences des sondes, en calculant des courbes d'évaluation de la charge des principales ressources en jeu. Ces évaluations se basent sur des raisonnements énergétiques (techniques de Propagation de Contraintes). Le chapitre 7 développe une méthode de Génération de Colonnes pour le calcul de bornes du problème de planification des prises de vue d'un satellite Pléiades. Cette approche se base sur la décomposition du modèle linéaire du problème, en un problème (esclave) de recherche de "pseudo-séquences" de prises de vue, d'une part et, d'autre part, en un programme linéaire (problème maître), qui contient certaines contraintes particulières du problème. Nous décrivons des algorithmes pour la recherche de pseudo-séquences, en incluant des techniques de filtrage des données et nous décrivons le processus de résolution par Génération de Colonnes du programme linéaire relâché. Nous terminons par la validation de cette approche et par des perspectives d'amélioration de la méthode et de poursuite des travaux.

Première partie

Applications spatiales et optimisation combinatoire

Chapitre 1

Problèmes d'optimisation combinatoire dans les applications spatiales

1.1 Introduction

Les systèmes spatiaux, de leur conception à leur maintenance, impliquent toujours des investissements lourds en termes de moyens, de compétences scientifiques et de temps de travail. Les missions spatiales utilisant ces systèmes doivent satisfaire des critères de performance et de qualité de service de plus en plus élevés.

De la gestion de ces systèmes spatiaux découlent des problèmes d'optimisation de différentes natures. D'une part, on a des problèmes qui sont fortement liés à une étude complexe de la mécanique spatiale et des technologies employées, où les domaines de variables sont principalement continus (par exemple, recherche d'une géométrie optimale d'un système spatial, étude du maintien à poste d'une constellation à moindre coût [Brochet 99]). D'autre part, on a des problèmes qui concernent plus précisément la planification des missions d'un point de vue de la gestion des ressources et qui font principalement intervenir des variables à domaines discrets.

C'est ce deuxième type de problèmes, fortement combinatoires, auquel nous nous intéressons dans ce travail. Après une présentation de problèmes classiques d'optimisation combinatoire, ce premier chapitre passe en revue les méthodes de modélisation et de résolution (issues le plus souvent des communautés de la Recherche Opérationnelle et de l'Intelligence Artificielle) proposées pour traiter les problèmes d'optimisation combinatoire émergeant dans le cadre de la planification de missions spatiales.

1.2 Problèmes classiques d'optimisation combinatoire

Un problème d'optimisation consiste à chercher une instanciation d'un ensemble de variables soumises à des contraintes, de façon à maximiser ou minimiser un critère. Lorsque les domaines de valeurs des variables sont discrets, on parle alors de problèmes d'optimisation combinatoire.

Nous présentons rapidement ici quatre problèmes classiques d'optimisation combinatoire : le problème du sac-à-dos, le problème d'affectation, le problème du voyageur de commerce et le problème d'ordonnancement.

1.2.1 Problème du sac-à-dos

Le "problème du sac-à-dos" est un problème de sélection qui consiste à maximiser un critère de qualité sous une contrainte linéaire de capacité de ressource. Il doit son nom à l'analogie qui peut être faite avec le problème qui se pose au randonneur au moment de remplir son sac-à-dos : il lui faut choisir les objets à emporter de façon à avoir un sac le plus "utile" possible, tout en respectant son volume.

Plus formellement, on peut le décrire de la façon suivante. Soit un ensemble de n éléments et une ressource disponible en quantité limitée, b. Pour $j = 1$ à n, on note p_j le profit associé à la sélection de l'élément j et on note a_j la quantité de ressource que nécessite l'élément j, s'il est sélectionné. Les coefficients p_j et a_j prennent des valeurs positives pour tout $j = 1$ à n. Le problème du sac-à-dos consiste à choisir un sous-ensemble des n éléments qui maximise le profit total obtenu, en respectant la quantité de ressource disponible [Nemhauser & Wolsey 88].

On associe à chaque élément j une variable de sélection, x_j, binaire, égale à 1 si j est sélectionné, égale à 0 sinon. Le profit total obtenu peut alors s'écrire comme la somme : $\sum_{j=1}^{n} p_j.x_j$ et la quantité totale de ressource utilisée comme la somme : $\sum_{j=1}^{n} a_j.x_j$. Le problème du sac-à-dos se modélise donc sous la forme :

$$\begin{array}{ll} Max & \sum_{j=1}^{n} p_j.x_j \\ \text{s.c.} & \sum_{j=1}^{n} a_j.x_j \leq b \\ & x_j \in \{0,1\} \quad \forall\, j = 1..n \end{array}$$

La contrainte de ressource est appelée "contrainte de sac-à-dos" ; on la retrouve dans des problèmes d'optimisation, issus de nombreux domaines d'application, qui mettent en jeu des ressources à capacité limitée.

Dans le cas où l'on a plusieurs contraintes de ce type (par exemple, le randonneur peut considérer non seulement un volume maximal, mais également un poids maximal, que son sac peut supporter), on parle de problème de sac-à-dos "multidimensionnel".

Le problème du sac-à-dos a fait l'objet de différents travaux proposant des méthodes exactes de résolution. Un état de l'art détaillé de ces approches est présenté dans [Martello *et al.* 00]. Les algorithmes proposés relèvent de trois principaux types de méthodes. Premièrement, des algorithmes de type *séparation et évaluation* [1] ont été proposés dans les années 70, permettant de traiter efficacement des instances de petite taille. Ces performances ont par la suite été améliorées par l'adjonction de contraintes supplémentaires pour renforcer les bornes dans l'arbre de recherche. Deuxièmement, des algorithmes se basant sur l'identification d'une variable critique et d'un sous-ensemble associé de variables, sur lequel on applique une recherche arborescente tronquée, ont permis, à partir des années 80, d'augmenter la taille des instances pouvant être résolues (jusqu'à $n = 100000$). Troisièmement, des algorithmes efficaces de programmation dynamique ont été proposés. En particulier, dans [Martello *et al.* 99],

1. Nous renvoyons à la lecture du paragraphe 2.3.3, chapitre 2, pour une présentation de la résolution de problèmes par séparation et évaluation.

la programmation dynamique est combinée avec l'identification d'une variable critique et l'utilisation de techniques de renforcement des bornes.

Concernant le problème de sac-à-dos multidimensionnel, nous renvoyons à la lecture de [Oliva *et al.* 01] qui détaille les différentes approches existantes.

1.2.2 Problème d'affectation

Le "problème d'affectation" consiste à établir des liens entre les éléments de deux ensembles distincts, de façon à minimiser un coût et en respectant des contraintes d'unicité de lien pour chaque élément.

On considère m tâches et n agents, avec $n \geq m$. Pour tout couple (i,j) ($i = 1$ à m, $j = 1$ à n), l'affectation de la tâche i à j entraîne un coût de réalisation noté $c_{i,j}$ ($c_{i,j} \geq 0$). Chaque tâche doit être réalisée exactement une fois et chaque agent peut réaliser au plus une tâche. Le problème consiste à affecter les tâches aux agents, de façon à minimiser le coût total de réalisation et en respectant les contraintes de réalisation des tâches et de disponibilité des agents [Nemhauser & Wolsey 88].

À tout couple tâche/agent (i,j), on associe une variable d'affectation, $x_{i,j}$, binaire, qui prend la valeur 1 si la tâche i est affectée à l'agent j et 0 sinon. Le coût total de réalisation des tâches s'exprime alors par la somme : $\sum_{i=1}^{m} \sum_{j=1}^{n} c_{i,j}.x_{i,j}$. Le nombre d'agents réalisant la tâche i est donné par : $\sum_{j=1}^{n} x_{i,j}$, pour tout $i = 1$ à m et le nombre de tâches réalisées par l'agent j est donné par : $\sum_{i=1}^{m} x_{i,j}$, pour tout $j = 1$ à n. On peut donc modéliser le problème d'affectation sous la forme :

$$
\begin{array}{lll}
Min & \sum_{i=1}^{m} \sum_{j=1}^{n} c_{i,j}.x_{i,j} & \\
\text{s.c.} & \sum_{j=1}^{n} x_{i,j} = 1 & \forall\, i = 1..m \\
 & \sum_{i=1}^{m} x_{i,j} \leq 1 & \forall\, j = 1..n \\
 & x_{i,j} \in \{0,1\} & \forall\, i = 1..m, \forall\, j = 1..n
\end{array}
$$

Les contraintes de ce problème se retrouvent dans de nombreuses applications mettant en jeu des problèmes d'allocation de ressources. Elles sont généralement appelées "contraintes d'affectation".

En théorie des graphes, on peut se ramener à un "problème de couplage dans un graphe biparti" [Laburthe 98]. On dit d'un graphe G qu'il est biparti si l'on peut diviser les sommets en deux ensembles X_1 et X_2 de telle sorte que toutes les arêtes dans le graphe joignent un sommet de X_1 à un sommet de X_2. Un "couplage" dans un graphe biparti est un ensemble d'arêtes qui n'ont, 2 à 2, aucun sommet commun dans G.

En associant X_1 à l'ensemble des tâches, de cardinalité m et X_2 à l'ensemble des agents, de cardinalité n, une arête (i,j) dans le graphe G (avec $i \in X_1$ et $j \in X_2$) représente la possibilité d'affecter la tâche i à l'agent j ; on associe le poids $c_{i,j}$ à chaque arête (i,j) de G. Le poids d'un couplage étant défini comme la somme des poids de ses arêtes, le problème d'affectation revient alors à chercher un couplage de cardinalité m de poids minimal dans le graphe G.

Le cas particulier où X_1 et X_2 sont de même cardinalité (correspondant au cas $n = m$ pour le problème d'affectation) est fréquemment étudié ; on s'intéresse alors à la recherche d'un couplage de cardinalité maximale. Si on considère des ensembles X_1 et X_2 de cardinalité n et s'il existe n^2 arêtes dans le graphe G (le graphe biparti est complet), alors le couplage

maximal est de cardinalité n et il est appelé "couplage parfait". On peut étendre ce problème à celui de la recherche d'un couplage maximal de poids minimal dans G.

Le problème d'affectation, ou de couplage dans un graphe biparti, peut être modélisé comme un problème de flot maximal à coût minimal dans lequel les capacités des arcs sont toutes égales à 1. C'est un problème classique de la théorie des graphes qui revient à chercher à faire passer un débit maximal à travers un réseau, pour un moindre coût [2]. Des algorithmes simples et efficaces existent pour résoudre ce problème ; en particulier l'algorithme de *Busaker et Gowen* qui part d'un flot nul et qui l'augmente progressivement par recherche de "chaînes augmentantes" (*i.e.,* chemins sur les arcs desquels on peut systématiquement augmenter le flot), de coût minimal.

La "méthode Hongroise", proposée par *Kuhn* en 1955, est un algorithme dual qui s'appuie sur une modélisation du problème d'affectation sous forme d'un programme linéaire, mais qui peut être vu comme une variante de l'algorithme de *Busaker et Gowen*, spécialisée pour la structure biparti du graphe. Du fait de sa grande efficacité sur ce type de problème, c'est l'algorithme de référence en Recherche Opérationnelle pour résoudre le problème d'affectation. Son principe est basé sur le fait que les couplages de poids minimal dans le graphe du problème primal sont exactement les couplages de cardinalité maximale dans le graphe du problème dual (voir par exemple [Laburthe 98] ou [Gondran & Minoux 95] pour une présentation détaillée de la méthode).

1.2.3 Problème du voyageur de commerce

Le "problème du voyageur de commerce", ou TSP (pour *Traveling Salesman Problem*), est le suivant : un représentant de commerce ayant n villes à visiter souhaite établir une tournée qui lui permette de passer exactement une fois par chaque ville et de revenir à son point de départ pour un moindre coût, c'est-à-dire en parcourant la plus petite distance possible. C'est un des problèmes les plus anciennement et largement étudiés en optimisation combinatoire. Ses applications sont nombreuses. Par exemple, des problèmes de séquencement de processus de fabrication ou d'optimisation de parcours en robotique peuvent s'exprimer directement sous forme d'un TSP et certains problèmes, comme les problèmes de transport, sont plus complexes que le TSP mais présentent une structure sous-jacente de type TSP.

Soit $G = (X,U)$, un graphe dans lequel l'ensemble X des sommets représente les villes à visiter, ainsi que la ville de départ de la tournée, et U, l'ensemble des arcs de G, représente les parcours possibles entre les villes. À tout arc $(i,j) \in U$, on associe la distance de parcours $d_{i,j}$ de la ville i à la ville j. La longueur d'un chemin dans G est la somme des distances associées aux arcs de ce chemin. Le TSP se ramène alors à la recherche d'un circuit hamiltonien (*i.e.,* un chemin fermé passant exactement une fois par chacun des sommets du graphe) de longueur minimale dans G. Dans le cas où il existe certains arcs $(i,j) \in U$ pour lesquels $d_{i,j} \neq d_{j,i}$, on parle de TSP asymétrique [Nemhauser & Wolsey 88].

On peut formuler le TSP de manière équivalente en associant à chaque couple (i,j) de villes à visiter ($i = 1$ à n, $j = 1$ à n et $i \neq j$) une distance $\delta_{i,j}$ égale à $d_{i,j}$ s'il existe un moyen d'aller directement de i à j (*i.e.,* $(i,j) \in U$ dans G), fixée à ∞ sinon et une variable de succession, $x_{i,j}$, binaire, qui prend la valeur 1 si la ville j est visitée immédiatement après la

2. Pour un exposé formel du problème de flot maximal à coût minimal et, de façon plus générale, pour une présentation détaillée des différents éléments de la théorie des graphes, nous renvoyons à la lecture d'ouvrages spécialisés comme par exemple [Gondran & Minoux 95].

ville i dans la tournée et qui prend la valeur 0 sinon. Le TSP est alors modélisé par :

$$\begin{array}{lll} Min & \sum_{i=1}^{n}\sum_{j=1}^{n}\delta_{i,j}.x_{i,j} & \\ \text{s.c.} & \sum_{j=1}^{n} x_{i,j} = 1 & \forall\, i = 1..n \\ & \sum_{i=1}^{n} x_{i,j} = 1 & \forall\, j = 1..n \\ & \sum_{i\in S, j\notin S} x_{i,j} \geq 2 & \forall\, S \subset X, S \neq \emptyset \\ & x_{i,j} \in \{0,1\} & \forall\, i = 1..n, \forall\, j = 1..n \end{array}$$

Les deux premières contraintes traduisent le fait que chaque ville doit être visitée exactement une fois ; la troisième contrainte interdit les solutions composées de sous-tours disjoints, elle est généralement appelée contrainte d'élimination des sous-tours.

Des algorithmes de Programmation Linéaire en Nombres Entiers ont été développés pour résoudre de façon exacte le TSP (nous reviendrons au chapitre 2 sur la PLNE). En particulier, des méthodes de recherche par séparation et évaluation sont renforcées efficacement par l'adjonction de coupes (algorithmes de *branch and cut*) et de techniques de propagation de contraintes dans l'arbre de recherche. Des algorithmes de programmation dynamique pour la recherche de circuits hamiltoniens dans un graphe, ainsi que la Programmation Par Contraintes (PPC) fournissent également de bons résultats pour des problèmes allant jusqu'à une centaine de nœuds [Laburthe 98]. De plus, on dispose de procédures heuristiques et de techniques d'optimisation locale efficaces (par exemple, l'algorithme de *Lin et Kernighan*, voir [Laburthe 98]), ainsi que de calculs de bornes inférieures, basés par exemple sur une relaxation lagrangienne du TSP, permettant de fournir un bon encadrement de l'optimum.

Il existe de nombreuses variantes du TSP, obtenues soit par adjonctions de contraintes – comme le TSP avec fenêtres de temps (TSPTW pour *Traveling Salesman Problem with Time Windows*) dans lequel la visite de chaque ville doit se faire dans un intervalle de temps donné –, soit par modification, comme par exemple les problèmes de tournée de véhicules (VRP pour *Vehicule Routing Problem*) dans lesquels on ne considère plus un unique représentant de commerce pour visiter les villes mais une équipe (une flotte de véhicules) et qui peuvent être vu comme des problèmes de flot. Pour une présentation détaillée des différentes variantes du TSP, ainsi que des méthodes de résolution existantes, on peut se référer à un ouvrage spécialisé comme [Gutin & Punnen 02], ou à [Laporte 92] pour une présentation plus synthétique.

1.2.4 Problème d'ordonnancement

Le "problème d'ordonnancement" consiste à séquencer et placer dans le temps un ensemble d'activités (entités élémentaires de travail), compte tenu de contraintes temporelles (délais, contraintes d'enchaînement, ...) et de contraintes portant sur l'utilisation et la disponibilité des ressources requises par les activités [Esquirol & Lopez 99]. Posé ainsi, il s'agit d'un problème de satisfaction de contraintes qui trouve ses applications dans divers domaines (gestion de projets, ateliers de production, ...) et qui fait l'objet de travaux de recherche d'un point de vue de l'aide à la décision, notamment par des approches par contraintes [Lopez 03]. Dans un contexte d'optimisation, on cherche de plus à minimiser (ou maximiser) un critère, comme par exemple la durée totale de réalisation des activités (minimisation du *Makespan*).

Le terme "problème d'ordonnancement", contrairement aux trois problèmes vus précédemment, ne fait pas référence à un problème totalement défini pour lequel il existe une formulation

mathématique directe, mais plutôt à une famille de problèmes. En effet, un problème d'ordonnancement est défini par la donnée des activités et des ressources qui le constituent et ces éléments peuvent être de natures très variées.

Par exemple, une ressource est "disjonctive" (ou non partageable) si elle ne peut pas exécuter plus d'une activité à la fois (on parle alors de problème d'ordonnancement disjonctif) ; dans le cas contraire, la ressource est dite "cumulative" (et conduit à un problème d'ordonnancement cumulatif). Une ressource est "renouvelable" si, après avoir été utilisée par une ou plusieurs activités, elle est à nouveau disponible en même quantité (machine, processeur, ...) ; la quantité de ressource utilisable à chaque instant est limitée. Si la réalisation d'une activité l'ampute d'une certaine quantité, la ressource est au contraire "consommable" (matières premières, budget, ...) et dans ce cas, la consommation globale au cours du temps est limitée. Une ressource est "doublement contrainte" si son utilisation instantanée et sa consommation globale sont, toutes deux, limitées (le financement de projets en est l'exemple le plus typique).

Concernant les activités, on peut considérer qu'il est possible de les interrompre puis de les reprendre plus tard, ou au contraire qu'elles doivent être exécutées sans interruption. On parle alors respectivement de problème "préemptif" et "non préemptif". Différentes contraintes peuvent porter sur les activités ; par exemple des contraintes de "fenêtres" d'exécution, avec des dates de "début au plus tôt" et de "fin au plus tard", des "contraintes de précédence" qui imposent que certaines activités soient finies avant que d'autres ne puissent débuter, ou encore des contraintes de durée entre deux activités (temps d'attente limité ou, au contraire, temps de préparation nécessaire entre deux tâches). On peut se référer à des ouvrages spécialisés sur les problèmes d'ordonnancement, comme [Blazewicz *et al.* 01] ou [Lopez & Roubellat 01], pour plus de détails sur ces notions, ainsi que la présentation d'autres critères de classification.

Plusieurs types de variables de décision peuvent être utilisés pour modéliser un problème d'ordonnancement. Par exemple, dans le cas d'un problème d'ordonnancement à une machine, disjonctif et non-préemptif, [Van Den Akker 94] présente une revue des différentes formulations mathématiques proposées dans la littérature. On relève quatre types de variables de décision : des variables réelles indiquant le début ou la fin de l'exécution d'une activité sur la machine (t_i est la date de début d'exécution de l'activité i), des variables binaires indiquant une relation de succession immédiate entre deux tâches ($f_{i,j} = 1$ si l'activité j est exécutée juste après i sur la machine, $f_{i,j} = 0$ sinon), des variables binaires indiquant la position d'une activité dans la séquence ($s_{i,j} = 1$ si l'activité i est la $j^{\text{ème}}$ activité exécutée par la machine, $s_{i,j} = 0$ sinon), des variables binaires, indicées sur le temps discrétisé et indiquant le début d'une activité à un certain pas de temps ($x_{i,t} = 1$ si l'activité i débute au pas de temps t, $x_{i,t} = 0$ sinon).

Les diverses contraintes s'expriment plus ou moins aisément dans les différents formalismes. Par exemple, une formulation basée uniquement sur les dates de début des activités permet de formuler des contraintes de précédence du type "l'activité i est réalisée avant l'activité j" ($t_j - t_i > p_i$, où p_i est la durée d'exécution de l'activité i), mais ne permet pas d'exprimer certaines relations, comme les temps de préparation entre deux activités. Il est nécessaire dans ce cas d'introduire une variable indiquant la succession des activités entre elles (variable de type "successeur" ou "position dans la séquence").

On obtient ainsi des modèles linéaires pour un grand nombre de problèmes d'ordonnancement. La relaxation linéaire de ces modèles fournit des bornes qui peuvent être utilisées directement à l'intérieur d'un algorithme de recherche arborescente par séparation et évaluation,

ou qui peuvent être affinées par différentes procédures (relaxation Lagrangienne, adjonction de coupes). De façon générale, les techniques purement linéaires ne donnent pas de bons résultats sur les problèmes d'ordonnancement. Les algorithmes les plus efficaces, que ce soit pour une résolution exacte du problème d'optimisation ou pour la recherche de solutions admissibles de bonne qualité, reposent le plus souvent sur une utilisation de techniques poussées de propagation de contraintes [Laburthe 98].

1.3 Problèmes d'optimisation combinatoire issus d'applications spatiales

1.3.1 Spécificité des problèmes posés

La planification des activités d'une mission spatiale – scientifique, militaire ou commerciale – pose des problèmes d'optimisation combinatoire complexes, qui, tout en se rapprochant de certains des problèmes classiques que nous venons d'évoquer, présentent un certain nombre de spécificités [Hall & Magazine 94, Muscettola *et al.* 95].

Tout d'abord, une fois mis en place, un système spatial n'est quasiment plus accessible pour d'éventuelles opérations de maintenance ou de réparation. Ceci implique donc, d'une part, que le projet doit être totalement défini et prêt à fonctionner avant son lancement et, d'autre part, qu'il est nécessaire d'avoir une gestion de l'ensemble du système extrêmement fiable. En particulier, les différentes opérations réalisées ne doivent pas amener le système à se trouver dans des situations de blocage (par exemple, par la destruction d'un instrument de prise de vue suite à une orientation directe vers le soleil, par l'immobilisation d'un rover dans une ornière, ...).

De plus, les différentes ressources d'une mission spatiale sont généralement "précieuses et rares" [Muscettola *et al.* 95]. Précieuses, comme par exemple un instrument de mesure scientifique, car c'est en grande partie sur leur bon fonctionnement et sur leur judicieuse exploitation que repose le succès de la mission, elles utilisent le plus souvent des technologies de pointe ayant impliqué le travail de nombreuses personnes, sur de longues périodes et la part du budget global de la mission allouée à leur conception et leur fabrication est importante. Rares, car pour des raisons de coût, de limitation de la charge utile embarquée, de l'impossibilité de les réparer ou les remplacer, les ressources sont disponibles en quantité toujours strictement limitée et sont pourtant énormément sollicitées, amenant ainsi de nombreux conflits à arbitrer.

Les ressources considérées dans les problèmes de planification de missions spatiales sont de diverses natures :

- instruments scientifiques embarqués (téléscopes, radar, caméras, ...), qui le plus souvent peuvent être assimilés à des ressources disjonctives en ordonnancement,
- ressources énergétiques (combustibles – ergol –, batteries électriques), place mémoire pour le stockage des données, qui peuvent être vues comme des ressources cumulatives.

Le temps peut également être considéré comme une ressource, d'autant qu'il est généralement fortement contraint. En effet, d'une part l'exécution d'une activité n'est possible, le plus souvent, que dans une certaine fenêtre temporelle, pour des raisons de réalisabilité ou d'intérêt de la tâche. D'autre part, en particulier lorsque des satellites défilants sont impliqués dans le problème, les ressources ne sont pas disponibles de façon continue mais seulement sur

certains intervalles. De plus, les temps de reconfiguration des différents instruments entre deux exécutions sont rarement négligeables et ils induisent de nombreuses contraintes de type "temps de préparation".

Aux différentes contraintes récurrentes, évoquées jusqu'à présent, peuvent s'ajouter, suivant le problème, des contraintes complexes d'exclusion mutuelle, ou au contraire des contraintes de synchronisation ou de conjonction entre des activités (par exemple, nous présenterons dans les chapitres 3 à 5 les contraintes de "handover", qui forcent deux activités successives à se chevaucher pendant une certaine durée, ou encore les contraintes "stéréoscopiques" qui imposent que certaines activités soient réalisées par paire ou pas du tout). Ces contraintes se rencontrent également en dehors du contexte des applications spatiales, en particulier dans certains problèmes d'ordonnancement, mais elles restent exotiques et font l'objet de relativement peu de publications de travaux.

On relève deux principaux types de problèmes de planification dans les missions spatiales : les problèmes d'allocation de liens de communications station(s)/satellite(s) et les problèmes de sélection et ordonnancement d'activités à bord d'engins spatiaux.

Les problèmes d'allocation de liens entre des stations au sol et un ou plusieurs satellites se posent non seulement dans les constellations de communication de type Globalstar[3] ou Iridium[4], mais également dans le cadre de missions d'exploration, pour assurer la communication entre les centres de contrôle terrestres et les systèmes spatiaux (par exemple, les récentes missions d'exploration de Mars). Ces problèmes sont généralement soumis à de très fortes contraintes de qualité de service (par exemple, assurer la continuité des liens de communication établis).

Les problèmes de sélection et ordonnancement d'activités se posent plutôt dans le cadre de la gestion d'instruments embarqués, dans les missions à caractère scientifique nécessitant la réalisation d'une grande variété d'expériences, ou bien dans des missions plus routinières, comme l'observation de la Terre par satellites. Dans ces problèmes, du fait notamment de la "rareté" des ressources et du grand nombre de parties – scientifiques, pays ou clients – se trouvant engagées dans la mission, le nombre d'activités demandées au système est toujours largement supérieur à ce qui peut en être attendu. Ainsi, la difficulté de ces problèmes réside tout autant dans la sélection des activités qui pourront être réalisées que dans leur ordonnancement et la gestion des conflits de ressources.

La planification des activités d'une mission spatiale permet de valoriser les investissements financiers, temporels et humains qui lui sont alloués. Les horizons de calcul considérés pour ces problèmes de planification sont typiquement de l'ordre de la journée ou de la semaine, ce qui est peu en regard de la durée de la mission (plusieurs mois ou plusieurs années), mais permet d'être réactif à l'évolution du système. Les problèmes d'optimisation combinatoire sous-jacents se posent donc de façon récurrente au cours de la mission. Les méthodes et stratégies utilisées pour résoudre ces problèmes doivent ainsi être étudiées et testées en amont de la phase de lancement de la mission, compte tenu des connaissances du système, de façon à fournir des algorithmes fiables et performants pour la planification des activités, au cours de la mission.

Plusieurs publications font état de travaux de recherche sur ces problématiques, dans le cadre d'études spécifiques à certaines applications spatiales ou d'un point de vue des méthodologies de la Recherche Opérationnelle ou de l'Intelligence Artificielle. Dans la suite, nous

3. http://www.globalstar.com
4. http://www.iridium.com

présentons les différentes approches utilisées pour la résolution des problèmes de planification des activités d'applications spatiales, que nous avons relevées dans ces publications.

1.3.2 Modélisation et résolution : revue des méthodes utilisées

1.3.2.1 Heuristiques et algorithmes dédiés

Compte tenu de la complexité des problèmes à traiter et du temps imparti à leur résolution en pratique, les algorithmes proposés pour la planification d'activités dans les sytèmes spatiaux font très souvent appel à des méthodes approchées.

C'est le cas en particulier des problèmes de planification des missions d'observation de la Terre. Dans [Wolfe & Sorensen 00], trois algorithmes sont proposés pour résoudre un problème de planification des prises de vue d'un satellite d'observation de la Terre de type Landsat (satellites développés par la NASA [5]). Le problème est formulé par un programme linéaire en variables entières où le critère d'optimisation est la maximisation de la somme des valeurs associées aux prises de vue réalisées. Un algorithme glouton est d'abord proposé, basé sur l'utilisation de règles de priorités pour sélectionner et ordonnancer les prises de vue. Cet algorithme est ensuite amélioré par l'adjonction d'une technique de "look ahead" qui permet d'évaluer les conséquences de différents choix possibles, avant de prendre une décision de sélection. Enfin, un algorithme génétique est développé. La population considérée est composée de permutations de prises de vue, tirées des solutions obtenues par les algorithmes précédents et on fait appel à des stratégies de croisements et de mutation classiques. Le deuxième algorithme semble donner le meilleur compromis entre temps de calcul et qualité des solutions obtenues.

Sur un problème très similaire, [Bresina *et al.* 97] propose de générer une première solution par un algorithme glouton stochastique, puis de l'améliorer par une heuristique basée sur des mesures de la criticité des ressources pour guider la recherche. Ces algorithmes sont repris plus tard dans un contexte d'approche par contraintes dans [Frank *et al.* 01]. On note qu'en pratique il semble que les algorithmes utilisés dans un système tel que Landsat sont de type glouton, basés sur des règles de priorité simples et avec très peu de stratégies d'amélioration des solutions. Ceci est justifié dans [Frank *et al.* 01] par le fait que la combinatoire des problèmes considérés est trop grande pour être traitée par d'autres méthodes dans les temps impartis et parce que les incertitudes qui pèsent en général sur l'évolution du système sont telles qu'elles rendent les tentatives d'optimisation quasiment inutiles (par exemple, la présence des nuages est particulièrement difficile à prendre en compte dans les systèmes d'observation de la Terre et entraîne cependant la perte d'une partie importante des prises de vues réalisées – jusqu'à 70% sur certaines régions –). Dans [Pemberton 00], le problème de planification des prises de vue d'un satellite est présenté comme un problème d'ordonnancement "sur-contraint" (*i.e.,* problème soumis à un ensemble de contraintes tel qu'il n'existe pas de solution les vérifiant toutes ; ce type de problème fait l'objet de travaux spécifiques, voir par exemple [Schiex *et al.* 95]). Un algorithme hybride, combinant une descente gloutonne avec des procédures de recherche arborescente, permet d'améliorer sensiblement les résultats obtenus par des algorithmes gloutons classiques, dans des temps de calculs assez courts. Dans cette approche, l'algorithme glouton est utilisé pour partitionner les variables du problème suivant le niveau de priorité des requêtes associées et chaque sous-problème ainsi obtenu est résolu de façon exacte par une recherche arborescente.

5. *National Aeronautics and Space Administration*

La gestion des Satellites Pour l'Observation de la Terre (SPOT), menée par le CNES[6], soulève également des problèmes de sélection et ordonnancement des prises de vue (voir [Bensana *et al.* 99]), pour lesquels des approches heuristiques sont proposées. Des premiers résultats sont présentés dans [Verfaillie & Bensana 95], sur le projet SPOT5. Des méta-heuristiques classiques (recherche Tabou, recuit simulé) sont testées, ainsi que des algorithmes gloutons et des recherches arborescentes tronquées. Par la suite, sur la même problématique, [Vasquez & Hao 01] formule le problème de planification des prises de vue du satellite comme un problème de sac-à-dos multidimensionnel et obtient de meilleures solutions avec un algorithme de recherche Tabou avancé, adapté aux particularités du problème et qui intègre des techniques de propagation de contraintes. Ces résultats sont améliorés ensuite dans [Vasquez & Hao 03], avec une méthode qui utilise une recherche Tabou pour partitionner le problème puis qui résout chaque sous-problème de façon exacte par programmation dynamique. Cette méthode est reprise en partie et adaptée au cas de la gestion de satellites de nouvelle génération, pour la mission Pléiades qui va succéder au programme SPOT dans les prochaines années [Habet & Vasquez 03]. Nous reviendrons par la suite sur ce problème (chapitres 5 et 7).

La planification des observations du téléscope spatial Hubble de la NASA est une problématique très proche, abordée par exemple dans [Aktur & Kiliç 99] par des algorithmes gloutons stochastiques associés à des stratégies d'optimisation locale *ad hoc* ainsi qu'un algorithme de recuit simulé.

Pour résoudre des problèmes d'allocation de liens de communication pour des constellations de satellites dédiées à la communication, on fait également appel à des méthodes approchées. Dans ces problèmes, trouver une solution réalisable, c'est-à-dire le plus souvent, assurer un lien de communication continu entre différents clients, alors que la configuration de la constellation de satellites évolue au cours du temps, est en soi un problème difficile. Les critères d'optimisation qui peuvent s'y ajouter concernent alors le nombre de clients que l'on peut satisfaire, ou bien la quantité de ressources (nombre de satellites, nombre d'antennes, ...) utilisée, compte tenu d'un certain niveau de demande. Dans [Ferreira *et al.* 01a], différentes stratégies de décision sont passées en revue, basées sur des règles de priorité ou sur des approches probabilistes, et répondant à des variantes du problème d'allocation de liens de communication dans une constellation qui autorise les liens inter-satellites. [Landes 02] traite un problème d'allocation de liens de communication dans une constellation n'autorisant pas les liens inter-satellites. L'auteur propose un algorithme glouton et plusieurs algorithmes de recherche locale pour aborder ce problème, où l'objectif est de maximiser la satisfaction de la demande.

Enfin, [Hall & Magazine 94] traite un problème plus général de maximisation de la valeur d'une mission spatiale. Un ensemble de règles sont présentées, pour la sélection et l'ordonnancement des activités, puis sont intégrées dans une procédure heuristique de génération de solutions ou comme stratégie de branchement dans un algorithme de programmation dynamique. Les algorithmes qui en découlent sont testés sur des instances générées aléatoirement, compte tenu des caractéristiques classiques, discutées au paragraphe précédent, des problèmes issus d'application spatiales.

En pratique, les méthodes approchées sont donc largement utilisées pour la planification des différentes activités d'une mission spatiale. Cependant, elles sont souvent améliorées par

6. Centre National d'Études Spatiales

des techniques élaborées de PPC (Programmation Par Contraintes) et sont développées dans un contexte d'aide à la décision qui doit permettre au système d'être réactif face aux évolutions possibles de son environnement.

1.3.2.2 Approches par contraintes, méthodes d'Intelligence Artificielle

L'efficacité de la gestion d'un système spatial dépend en grande partie de la bonne coordination des différents sous-ensembles qui le composent et qui sont liés par de multiples contraintes. Plusieurs publications discutent de l'intérêt de développer des outils génériques et homogènes pour planifier et contrôler les différents aspects d'une mission spatiale, en s'appuyant sur des approches par contraintes pour modéliser les problèmes et réduire l'espace de recherche des solutions [Muscettola *et al.* 95, Bresina *et al.* 97, Aktur & Kiliç 99, Harrison & Price 99, Smith *et al.* 00].

Dans ces approches, les problèmes considérés sont le plus souvent modélisés sous forme de Problèmes de Satisfaction de Contraintes (CSP). Ceci permet de proposer des schémas génériques de développement de méthodes de résolution qui prennent en compte les contraintes récurrentes des problèmes de planification issus des applications spatiales (fenêtres temporelles, disponibilité et partage des ressources, ...) et auxquels on peut intégrer facilement des contraintes supplémentaires. Par exemple, le système HSTS (pour *Heuristic Scheduling Testbed System*) décrit dans [Muscettola *et al.* 95], permet la planification automatique de commandes pour un engin spatial et est utilisé dans le cadre des missions Cassini (orbiteur de la NASA associé à la sonde européenne Huygens, pour l'exploration de Saturne et Titan) et EUVE (téléscope embarqué utilisant les ultra-violets). Le système ASPEN (*Automated Scheduling/Planning Environment*), décrit dans [Fukunaga *et al.* 97], est présenté comme une structure modulaire et générique pour la planification d'une grande variété de problèmes. Il est mis en œuvre dans différents projets menés par la NASA (satellites d'observation de la Terre EO-1 et UFO-1 actuellement en orbite, sondes pour un projet d'exploration de Pluton,...).

Dans une approche d'aide à la décision, le système MEXAR, présenté dans [Cesta *et al.* 02b] est un support interactif développé pour la planification des échanges de données entre la Terre et un système d'exploration distant (utilisé pour la mission Mars-Express). Le système utilise des techniques de propagation de contraintes et propose différents algorithmes paramétrables pour guider l'utilisateur dans la recherche d'une solution satisfaisante [Oddi *et al.* 03].

En Intelligence Artificielle, les problèmes de planification font l'objet de travaux bien spécifiques, avec le développement de systèmes théoriques de production de plans – tel que Graphplan – [Russel & Norwig 95]. Les problèmes classiquement étudiés consistent à chercher un ensemble d'actions qui permettent de faire passer un système d'un état initial à un état final requis. Dans les applications spatiales, ces travaux trouvent particulièrement leurs applications lorsque qu'on cherche à rendre certains systèmes spatiaux autonomes. C'est le cas des atterrisseurs mobiles (rovers), tels que Spirit et Opportunity pour l'exploration de la planète Mars, qui sont périodiquement contrôlés depuis la Terre mais qui ont des calculateurs intégrés pour planifier leurs mouvements en fonction de leur environnement et compte tenu d'objectifs divers [Estlin & Gaines 02, Jónsson *et al.* 00].

Notons enfin que les heuristiques vues au chapitre précédent sont généralement renforcées par des techniques de PPC qui permettent de réduire efficacement l'espace de recherche (en particulier, [Harrison & Price 99, Frank *et al.* 01, Pemberton 00]).

1.3.2.3 Méthodes exactes

Les méthodes exactes telles que la programmation linéaire en nombres entiers, les procédures de recherche arborescente ou la programmation dynamique, offrent en théorie la garantie de trouver une solution optimale d'un problème et d'en prouver l'optimalité. Sur les problèmes issus d'applications spatiales, elles sont efficaces en pratique sur des instances de petite taille, mais sont rapidement confrontées à la combinatoire des problèmes considérés.

Cependant, l'étude des problèmes de planification des activités d'une mission spatiale par le biais de méthodes exactes présente plusieurs intérêts.

D'une part, la résolution exacte d'une version relâchée d'un problème (en simplifiant ou éliminant certaines contraintes) permet d'obtenir des bornes (supérieures ou inférieures, suivant que l'on considère une fonction de maximisation ou de minimisation) qui servent à évaluer la qualité des différentes méthodes heuristiques proposées pour la recherche de solutions admissibles. Ainsi, parmi les travaux cités précédemment, [Hall & Magazine 94] et [Vasquez & Hao 03] proposent des calculs de bornes, utilisant respectivement des algorithmes de relaxation lagrangienne et de programmation dynamique.

D'autre part, dans une optique d'étude "hors-ligne" des problèmes, les méthodes exactes permettent d'évaluer les capacités d'un système. Ainsi par exemple, la programmation mathématique est utilisée pour résoudre des problèmes de dimensionnement de systèmes spatiaux [Brochet 99, Lebbar 00] et des méthodes issues de la théorie des graphes sont proposées pour étudier les performances que l'on peut attendre de certaines ressources d'un système (antennes des satellites dans une constellation pour la communication, [Landes 02]).

L'étude de la modélisation d'un problème, de sa structure et des possibilités de décomposition peuvent permettre de ramener la résolution d'un problème globalement très complexe à celle d'un ensemble de sous-problèmes plus faciles à résoudre, et/ou mieux connus. On peut alors faire appel à des méthodes exactes pour résoudre tout ou partie de ces sous-problèmes. Dans le second cas, la résolution exacte de certains aspects du problème global, peut être intégrée efficacement dans une méthode de résolution heuristique.

Différents travaux font état de méthodes faisant appel à la décomposition de problèmes de planification de missions spatiales, exprimés sous forme de programmes mathématiques. Par exemple, [Brochet 99] propose un algorithme de résolution exacte pour un problème d'optimisation des manœuvres de satellites pour leur maintien à poste, basé sur une décomposition de Benders d'un modèle mathématique non-linéaire du problème. Dans [Lebbar 00], le problème d'allocation des communications Terre/satellite dans la constellation Skybridge est modélisé par un programme linéaire en nombre entiers. Ce modèle est décomposé par la méthode de Dantzig-Wolfe et résolu par un algorithme de génération de colonnes. [Lee & Park 01] montre qu'un problème d'allocation de communications dans une constellation autorisant les liens inter-satellite peut être vu comme un problème de coloration sur un multigraphe biparti. La relaxation linéaire du problème est résolue par génération de colonnes et une solution entière est ensuite obtenue par une recherche arborescente de séparation et évaluation. Pour un problème proche, un modèle linéaire est proposé dans [Ferreira *et al.* 00] et validé par programmation linéaire.

Le problème de sélection et ordonnancement des prises de vue du satellite SPOT5, déjà évoqué au paragraphe 1.3.2.1 pour avoir fait l'objet de différentes approches heuristiques, est traité dans [Gabrel 99] par un algorithme de génération de colonnes qui permet de calculer des bornes supérieures de bonne qualité. Par la suite, [Paschos *et al.* 01] étudie deux variantes de

ce même problème. Les auteurs proposent une formulation en termes de recherche de plus long chemin et d'ensemble stable dans un graphe et développent des algorithmes exacts et approchés issus de la théorie des graphes pour résoudre ces problèmes. Dans [Gabrel & Vanderpooten 02], une approche multicritère du problème est proposée. L'approche est basée sur l'utilisation d'un algorithme exact de recherche de plus court chemin dans un graphe sans circuit, intégré dans un processus interactif d'évaluation multicritère des chemins.

1.3.2.4 Approches par simulation : utilisation des Réseaux de Petri

Dans certains cas, il est difficile d'élaborer un modèle exprimant fidèlement toutes les contraintes du système tout en restant analytique. On se tourne alors vers une description de type procédurale, décrivant des enchaînements de causes et d'effets de façon détaillée. Les propriétés du système et ses paramètres significatifs sont alors obtenus par simulation.

Cette approche est principalement utilisée dans les phases de préparation d'une mission spatiale pour valider différentes prises de décisions. C'est un moyen de résolution de problèmes de décision par essai/évaluation/correction.

Dans ce contexte, les réseaux de Petri sont utilisés pour décrire les défaillances de composants, les stratégies de reconfiguration et les politiques de maintenance ou d'entretien. Un simulateur comme MISS-RdP développé par la société GFI Consulting, permet, par simulation de Monte Carlo, d'obtenir des évaluations de paramètres de la sûreté de fonctionnement, comme la disponibilité, dans des cas où les méthodes analytiques ne peuvent pas être utilisées ou, plus exactement, dans le cas où les méthodes analytiques nécessiteraient trop d'approximations et donneraient ainsi des résultats trop imprécis pour être réellement utiles.

Cet outil est utilisé au CNES pour simuler la disponibilité, en fonction du temps, de constellations de satellites et évaluer différentes stratégies pour remplacer les satellites défaillants.

Une question se pose alors : la description de nature procédurale et implicite des contraintes de séquencement apportée par les réseaux de Petri peut-elle être utilisée dans un cadre plus général pour la résolution de problèmes d'optimisation combinatoire? Cette question est posée clairement pour la première fois dans [Richard 97]. Lorsque le système comporte un ensemble de contraintes de séquencement nombreuses et trop complexes pour être décrites de façon simple par un ensemble de contraintes linéaires impliquant des variables booléennes, une approche possible consiste à commencer par la recherche de comportements consistants avec ces contraintes (il s'agit en quelque sorte d'une relaxation des autres contraintes).

Cela peut être vu comme une phase de recherche arborescente précédant une mise en équation et une résolution, par exemple par programmation mathématique. Le principe de la recherche de ces comportements (ou d'un sous-ensemble de ces comportements) est tout à fait analogue à de la propagation de contraintes. Chaque comportement est caractérisé par un scénario de franchissements de transitions consistant avec les contraintes non relaxées. Même si les contraintes ne sont pas exprimées de façon analytique par des équations, il s'agit bien d'une résolution par contraintes puisque l'on alterne des choix de transition à franchir (analogie avec l'instanciation d'une variable) avec des constructions de liste de transitions franchissables (propagation pour délimiter les nouveaux domaines admissibles pour les variables). Pour chaque comportement, un système d'inéquations linéaires dans lequel les variables sont les dates de franchissement des transitions est alors obtenu (sous la forme d'un "Simple Temporal Problem" – STP) [Mancel *et al.* 02].

En présence de fortes contraintes de séquencement, comme par exemple celles découlant de gammes de fabrication en productique, la démarche ci-dessus peut permettre de réduire fortement l'espace de recherche. Dans le cas où ces contraintes sont moins fortes, le nombre des comportements consistants avec les contraintes peut être trop grand pour autoriser une énumération complète. L'approche peut, dans ce cas, s'intégrer dans le cadre d'utilisation d'heuristiques. On n'explore alors que les comportements cohérents avec l'heuristique choisie [Mancel *et al.* 03a].

Cette démarche a été testée dans le cadre de l'étude de la constellation Skybridge [Mancel *et al.* 01b, Mancel *et al.* 01c], mais nous ne l'avons pas poursuivie pour les raisons suivantes. Dans les systèmes spatiaux que nous avons étudiés, les contraintes de séquencement de nature purement logique et imposées initialement (relations de précédence imposées par la nature des tâches et non par le fait qu'elles partagent des ressources communes) sont peu nombreuses ; il n'est donc pas judicieux de s'appuyer sur de telles contraintes pour une phase initiale de réduction de l'espace de recherche. D'autre part, tout au long de notre travail, nous avons privilégié les méthodes exactes et celles permettant l'obtention de bornes par rapport à des heuristiques permettant simplement d'obtenir de bonnes solutions. L'approche fondée sur les réseaux de Petri ne semblait donc pas prometteuse dans ce cadre qui concernait des systèmes spatiaux avec peu de contraintes de séquencement initialement données.

1.4 Conclusion

Ce chapitre propose un tour d'horizon des problèmes de planification qui se posent dans le cadre de missions spatiales.

Ces problèmes présentent des similitudes avec les problèmes d'optimisation combinatoire classiquement étudiés en Recherche Opérationnelle, mais également certaines spécificités en termes de problématiques, de contraintes et d'exigence sur la fiabilité des solutions. De plus, issus d'applications réelles, ces problèmes sont le plus souvent complexes et de grande taille.

La résolution de ces problèmes combinatoires fait l'objet de divers travaux de recherche dans les domaines de la Recherche Opérationnelle et de l'Intelligence Artificielle. En pratique, lorsque les solutions de planification ne sont pas effectuées "à la main", les algorithmes mis en œuvre font souvent appel à des méthodes approchées, ainsi qu'à des techniques de Propagation de Contraintes.

Les méthodes exactes, en particulier la Programmation Linéaire, offrent quant à elles un cadre formel pour la modélisation et l'analyse de la structure de ces problèmes. De plus, dans une étude "hors-ligne" d'une mission spatiale, c'est-à-dire en amont de son lancement effectif, ces méthodes peuvent être utilisées pour borner les problèmes d'optimisation combinatoire qui se posent et ainsi évaluer la qualité des algorithmes développés pour la recherche de solutions. Couplées à une approche par décomposition des problèmes, elles peuvent également être intégrées dans les processus de résolution, en traitant de façon exacte certains "sous-problèmes".

Dans ces travaux, nous étudions, du point du vue de leur modélisation et de leur résolution, des problèmes d'optimisation combinatoire qui émergent dans différents projets de missions spatiales développés au CNES. Nous nous intéressons particulièrement à l'apport de la Programmation Linéaire pour ces problèmes, associée à des méthodes de simplification de l'espace de recherche, par décomposition des problèmes, ou grâce à des techniques de Propagation de

Contraintes.

Ainsi, dans le chapitre suivant, nous présentons les fondements des méthodes que nous retenons pour la suite de nos travaux : les concepts des techniques de Propagation de Contraintes, les principes de la Programmation Linéaire, ainsi que ceux de la Génération de Colonnes, qui est une extension de la Programmation Linéaire au traitement des problèmes de grande taille, basée sur une décomposition des modèles linéaires.

Chapitre 2

Techniques de résolution retenues

2.1 Introduction

Nous avons vu au chapitre précédent que de nombreuses méthodes peuvent être utilisées pour aborder les problèmes de planification de missions spatiales.

Dans ces travaux, nous nous intéressons principalement à l'apport des méthodes exactes pour ces problèmes et en particulier à la Programmation Linéaire qui fournit un formalisme de modélisation et des méthodes de résolution permettant d'aborder un large éventail de problèmes d'optimisation.

Face à des problèmes complexes et fortement combinatoires, les algorithmes issus de la Programmation Linéaire perdent cependant rapidement de leur efficacité. Des techniques visant à réduire ou simplifier l'espace de recherche des solutions peuvent alors être associées à ces algorithmes afin de les renforcer. À ce titre, les techniques de Propagation de Contraintes et certaines méthodes de décomposition peuvent être envisagées.

Ainsi dans ce chapitre, nous présentons en premier lieu un bref tour d'horizon des concepts et techniques liés à la Propagation de Contraintes. Nous détaillons ensuite les principes de la Programmation Linéaire, son adaptation au cas des problèmes en variables entières et sa forme généralisée, appelée Génération de Colonnes, basée sur une décomposition du modèle en deux problèmes interdépendants.

2.2 Propagation de contraintes

La propagation de contraintes consiste à simplifier la résolution d'un problème en affinant la connaissance que l'on a de ses variables, ou en démontrant l'absence de solution. Elle opère sur des problèmes de satisfaction de contraintes (CSP) qui regroupent des problèmes classiques d'Intelligence Artificielle, ainsi que des problèmes issus de la Recherche Opérationnelle, comme les problèmes d'ordonnancement [Alliot & Schiex 94].

Un *problème de satisfaction de contraintes* P est défini par un triplet $P =< X,D,C >$ où $X = \{x_1, \ldots, x_n\}$ est l'ensemble des variables de P, $D = \{D_1, \ldots, D_n\}$ est l'ensemble des domaines de valeurs des variables et $C = \{C_1, \ldots, C_m\}$ est l'ensemble des contraintes liant les variables.

La propagation de contraintes désigne un ensemble de techniques de réécriture des

contraintes et de règles de déduction logiques qui permettent de réduire les domaines de valeurs des variables, ou de simplifier l'expression des contraintes, par exemple en identifiant et en éliminant celles qui sont redondantes. Dans un contexte d'aide à la décision, ces techniques permettent également la caractérisation de solutions admissibles en dégageant des degrés de liberté pour la prise de décision effective. La propagation peut enfin dans certains cas apporter la preuve d'une inconsistance globale du problème, et ce avant toute tentative de résolution.

Nous présentons ici une synthèse des techniques classiques de propagation de contraintes.

2.2.1 Propagation de contraintes temporelles

Un CSP dont les variables (X) correspondent à des entités temporelles (instants, intervalles, durées) et dont les contraintes (C) sont binaires et représentent des relations numériques ou logiques entre les variables, est appelé un CSP *temporel* (TCSP) [Dechter *et al.* 91].

Un TCSP est *numérique* si les contraintes n'expriment que des relations numériques entre les variables. Dans ce cas, chaque domaine D_i d'une variable x_i est l'union d'un ensemble d'intervalles disjoints. De plus, chaque contrainte est une relation binaire, notée C_{ij}, limitant les valeurs possibles d'une distance entre deux variables, $x_j - x_i$, par un domaine D_{ij}, union d'intervalles disjoints [Dechter *et al.* 91]. Notons que les contraintes de domaine d'une variable peuvent être mises sous la forme de contraintes binaires en introduisant une variable artificielle x_0, représentant l'origine.

Une incohérence apparaît dans le problème lorsque l'ensemble des intervalles associé au domaine d'une variable ou d'une contrainte est vide.

2.2.1.1 Problèmes temporels simples

Dans le cas particulier où chaque domaine D_{ij} associé à une contrainte est réduit à un unique intervalle de valeurs, on parle de *problème temporel simple* (STP). On peut alors exprimer l'ensemble des contraintes du problème (précédences, dates limites, durées fixes) par un système d'inégalités linéaires entre deux variables et on peut obtenir le problème *minimal* (les domaines associés aux variables et aux contraintes ne comportent aucune valeur inconsistante) correspondant, de façon polynomiale.

En effet, les contraintes d'un STP peuvent être représentées sur un graphe potentiels (voir, par exemple, [Esquirol & Lopez 99] pour la représentation de problèmes d'ordonnancement). On peut alors appliquer une procédure de recherche de plus longs chemins entre tout couple de sommets sur ce graphe. Dans le formalisme des TCSP, la propagation correspondante est basée sur l'utilisation de deux opérations entre contraintes (*intersection* et *composition*) qui amène à déterminer la borne inférieure et la borne supérieure du domaine des contraintes.

L'algorithme ainsi obtenu, PC-0, permet d'exprimer des conditions de consistance locale entre tout triplet de variables d'un problème P. Appliqué à un STP, il est *complet* (toutes les valeurs inconsistantes sont éliminées). Il est alors équivalent à l'algorithme de Floyd-Warshall appliqué au graphe potentiels correspondant. Cet algorithme permet d'obtenir le domaine minimal des contraintes d'un STP. Si on recherche uniquement les domaines minimaux des variables d'un STP, on peut utiliser un algorithme de complexité moindre, comme l'algorithme de Bellman-Ford [Esquirol *et al.* 01].

2.2.1.2 Problèmes temporels généraux

Pour des problèmes temporels généraux, PC-0 est sain (les dates supprimées sont bien inconsistantes), mais il est incomplet (il ne garantit pas la détection d'un problème inconsistant). Une variante de cet algorithme, PC-1, consiste à effectuer plusieurs passes de PC-0 sur un TCSP numérique, jusqu'à parvenir à un *point fixe* (problème pour lequel plus aucun ajustement des domaines des contraintes n'est possible). PC-1 améliore PC-0 mais il reste incomplet ; de plus, son application est limitée à cause d'un phénomène de fragmentation excessive des intervalles associés aux contraintes [Schwalb & Dechter 97].

Afin de limiter ce phénomène, deux méthodes sont proposées dans [Schwalb & Dechter 97]. La première (ULT pour *Upper-Lower Tightening*) s'appuie sur une relaxation des contraintes permettant de se ramener à un STP. Elle détermine les contraintes minimales et conserve les "trous" dans les domaines de valeurs. La seconde méthode (LPC pour *Loose Path Consistency*) fait appel à la définition d'une nouvelle opération d'intersection qui permet de ne pas augmenter le nombre d'intervalles par contrainte mais qui agrandit les trous dans les domaines de valeurs.

Dans [Schwalb & Dechter 97], les auteurs montrent que ces différents algorithmes déterminent les mêmes bornes minimales et maximales pour chaque contrainte. La différence entre ces algorithmes tient uniquement à la gestion des trous dans les domaines de valeurs.

De nombreux problèmes d'ordonnancement peuvent être – au moins en partie – formulés comme des TCSP ou des STP. En effet, les contraintes temporelles telles que les dates limites ou les contraintes de précédences, se traduisent, dans le formalisme des TCSP, en contraintes binaires associées à un seul intervalle. Les contraintes de partage de ressources disjonctives peuvent également être exprimées comme des contraintes binaires, mais dont le domaine comporte deux intervalles. Les algorithmes vus précédemment peuvent donc être utilisés sur ces différents types de contraintes. En revanche, certaines contraintes de ressources, comme les ressources cumulatives, ne peuvent s'exprimer dans le formalisme des TCSP ; on a alors recours à des techniques de propagation spécifiques aux contraintes de partage de ressources.

2.2.2 Propagation de contraintes de partage de ressources

Dans un problème d'ordonnancement, la prise en compte des seules contraintes de partage des ressources définit un *problème de séquencement*. Il consiste à ordonner partiellement ou totalement, suivant que les ressources sont cumulatives ou disjonctives, les tâches nécessitant une même ressource, de façon à garantir que la consommation des tâches ne dépasse à aucun instant la quantité de ressource disponible.

La prise en compte des fenêtres de réalisation des tâches permet, grâce à l'utilisation de règles et d'algorithmes de propagation efficaces, de réduire l'ensemble des solutions du problème de séquencement : certaines configurations se révèlent interdites, d'autres deviennent obligatoires, ce qui peut conduire à resserrer les fenêtres d'exécution des tâches.

Il existe plusieurs types de règles de propagation des contraintes de partage des ressources : les *opérations locales* et les *opérations globales* sont deux premiers types de règles principalement dédiés à des problèmes dont les ressources sont disjonctives. Un troisième type de règles, lié au concept d'*énergie*, est particulièrement adapté au cas des ressources cumulatives.

Dans les paragraphes suivants, nous présentons les principaux aspects des opérations lo-

cales et opérations globales, puis nous développons plus en détail les raisonnements énergétiques.

2.2.2.1 Opérations locales, opérations globales

Les techniques appelées *opérations locales* sont basées sur l'application de règles d'inférence locales à une machine. Ces techniques opèrent sur des *ensembles critiques de tâches en conflit* (plus petits sous-ensembles de tâches qui ne peuvent s'exécuter simultanément compte tenu de la disponibilité insuffisante des ressources) et prennent leur source dans les travaux de J. Carlier et J. Erschler [Carlier 75, Erschler 76] qui proposaient des premières conditions de séquencement entre tâches permettant de simplifier la recherche de solutions de problèmes d'ordonnancement.

Dans un problème de séquencement à ressources disjonctives, il faut ordonner totalement les tâches qui nécessitent les mêmes machines. On cherche notamment à démontrer qu'une tâche ne peut pas — ou doit obligatoirement — être placée avant — ou après — toutes les tâches d'un ensemble (cf. les ensembles non postérieurs/non antérieurs dans [Erschler 76], les règles de " edge-finding " dans [Applegate & Cook 91], les sélections immédiates dans [Carlier & Pinson 94, Brucker *et al.* 94, Nuijten 94], et les règles " not-first/not-last " exposées dans [Baptiste & Le Pape 96, Torres & Lopez 00]). Ces déductions ont comme résultat de resserrer, ou *ajuster*, les fenêtres d'exécution des tâches.

Les *opérations globales*, applelées également techniques de *shaving* (cf. [Carlier & Pinson 94, Péridy 96, Martin & Shmoys 96]) sont basées sur la nature spécifique des contraintes disjonctives : une contrainte locale est posée et les ajustements correspondants sont propagés sur le problème global. La détection d'une inconsistance entraîne la pose de la contrainte inverse. Ce mécanisme se rapproche de certaines techniques de renforcement de la consistance dans les réseaux de contraintes, basées sur un principe de réfutation : une variable est instanciée ; le résultat est propagé ou utilisé dans un arbre de recherche ; si une inconsistance apparaît, la valeur impliquée dans l'instanciation est retirée du domaine de la variable (algorithmes SAC, singleton consistance d'arc [Debruyne & Bessière 97] et MAC, maintien de la consistance d'arc [Sabin & Freuder 94]).

Ce dernier type de règles de propagation permet d'approfondir la connaissance des domaines de valeurs (pouvant aller parfois jusqu'à une résolution du problème) mais est beaucoup plus coûteux en temps de calcul que les opérations locales.

2.2.2.2 Raisonnement énergétique

L'énergie correspondant au produit d'un temps par une quantité de ressources, les règles de propagation de contraintes basées sur ce concept permettent de raisonner simultanément sur les contraintes de temps et de ressources [Lopez 91, Lopez *et al.* 92]. Comme les opérations locales, ces raisonnements peuvent produire des conditions de séquencement, sans toutefois nécessiter une analyse du problème en termes d'ensembles critiques de tâches en conflit, mais il permettent aussi d'interdire la localisation d'une tâche sur certains intervalles de temps. Ils prennent tout leur intérêt lorsque certaines caractéristiques de réalisation, comme par exemple la durée des tâches, ne sont pas entièrement connues.

Les déductions liées au raisonnement énergétique font appel à des bilans d'utilisation des ressources sur certains intervalles temporels, ce qui entraîne l'identification et le calcul de

différentes énergies. Sur un intervalle de temps, l'énergie peut en effet être produite par une ressource ou consommée par une tâche. Dans ce dernier cas, on distinguera sa consommation minimale de sa consommation maximale sur l'intervalle.

Pour présenter ces différents éléments, on introduit les notations suivantes : Cap_k est la capacité d'une ressource k (la quantité maximale instantanée disponible de la ressource) ; int_{ik} est l'intensité d'une tâche i (la quantité nécessaire requise à chaque instant pour l'exécution de la tâche). Pour alléger la notation, nous faisons l'hypothèse qu'une tâche i ne requiert qu'une ressource et son intensité sera notée int_i.

Énergie produite, énergie consommée

L'énergie maximale disponible qu'une ressource k produit sur un intervalle $\Delta = [t_1,t_2]$ correspond au produit de la longueur de l'intervalle par la capacité (constante) Cap_k de la ressource :

$$W_k^{\Delta} = Cap_k \times (t_2 - t_1)$$

Pour une tâche i localisée dans le temps (st_i et ft_i, les dates de début et fin de i, sont fixés), l'énergie requise — ou *consommation* — par une tâche i sur un intervalle $\Delta = [t_1,t_2]$, notée w_i^{Δ}, est donnée par le produit de son intensité par l'intersection de Δ avec l'exécution de i :

$$w_i^{\Delta} = int_i \times \max[0, \min(ft_i,t_2) - \max(st_i,t_1)]$$

soit, sachant que $ft_i - st_i = p_i$ (la durée de i) :

$$w_i^{\Delta} = int_i \times \max[0, \min(p_i,t_2 - t_1,ft_i - t_1,t_2 - st_i)]$$

La formule précédente est illustrée par la figure 2.1 où les zones hachurées représentent la consommation d'une tâche entre t_1 et t_2 pour différentes positions dans sa fenêtre d'exécution $[r_i,d_i]$.

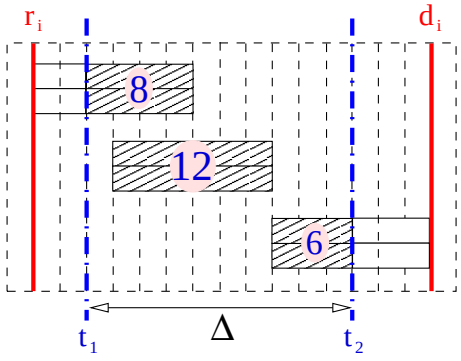


FIG. 2.1 – *Consommation d'une tâche pour trois de ses positionnements ($int_i = 2$)*

Les bornes inférieure et supérieure de la consommation d'une tâche sur un intervalle sont appelées respectivement *consommation obligatoire* et *consommation maximale*.

La consommation obligatoire, notée $\underline{w_i^{\Delta}}$, est obtenue en considérant les positions de la tâche pour lesquelles son intersection avec Δ est minimale :

$$\underline{w_i^{\Delta}} = int_i \times \max[0, \min(p_i,t_2 - t_1,r_i + p_i - t_1,t_2 - d_i + p_i)]$$

De manière duale, on obtient la formule de l'énergie maximale, notée $\overline{w}_k^\Delta$:

$$\overline{w}_i^\Delta = int_i \times \max[0, \min(p_i, t_2 - t_1, d_i - t_1, t_2 - r_i)]$$

Sur la figure 2.1, la consommation obligatoire de i est obtenue pour son calage à droite, soit $\underline{w}_i^\Delta = 6$; la consommation maximale est obtenue lorsque i est exécutée totalement entre t_1 et t_2 : $\overline{w}_i^\Delta = 12$.

Règles de déduction

Ces règles, basées sur des bilans énergétiques, permettent de déterminer des conditions d'admissibilité d'un ordonnancement, le séquencement entre tâches ou l'ajustement de fenêtres d'exécution.

Condition suffisante d'inconsistance globale

$$\exists \Delta \text{ tq } W_k^\Delta < \sum_{i \in T_k} \underline{w}_i^\Delta \Longrightarrow \text{ inconsistance}$$

En effet, dans un ordonnancement admissible, quel que soit l'intervalle considéré, la consommation globale de chaque ressource sur cet intervalle doit être inférieure à l'énergie produite par la ressource sur cet intervalle.

Ajustements (inconsistance locale) Pour une tâche i et un intervalle Δ donnés, soit $S_i^\Delta = W_k^\Delta - \sum_{j \in T_k \setminus \{i\}} \underline{w}_j^\Delta$ l'énergie maximale disponible pour exécuter i sur Δ compte tenu de la consommation obligatoire des autres tâches de T_k :

$$\exists i, \Delta \text{ tq } S_i^\Delta < \overline{w}_i^\Delta \Longrightarrow \text{ il existe des dates interdites pour } i$$

Les dates conduisant à une consommation plus grande que S_i^Δ sont inconsistantes et doivent être supprimées. La suppression de ces valeurs dans le domaine initialement permis, peut créer un ajustement des dates de début au plus tôt et/ou de fin au plus tard de i (avec éventuellement création d'un trou dans le domaine de st_i).

Relations de séquencement Pour une paire de tâches (i,j) et un intervalle Δ donnés, soit $S_{(i,j)}^\Delta = W_k^\Delta - \sum_{l \in T_k \setminus \{i,j\}} \underline{w}_l^\Delta$ l'énergie maximale disponible pour exécuter i et j sur Δ, compte tenu de la consommation obligatoire des autres tâches de T_k. On a la règle suivante :

$$\exists i, j, \Delta \text{ tq } S_{(i,j)}^\Delta < \overline{w}_i^\Delta + \overline{w}_j^\Delta \Longrightarrow \text{ il existe des couples de dates interdites pour } i \text{ et } j$$

En d'autres termes, si i et j consomment toutes deux sur Δ, elles doivent être éloignées d'une distance minimale.

On montre que le raisonnement énergétique fournit des règles dont la formulation recouvre celle des règles performantes de propagation dans le cas de ressources disjonctives (paires de disjonction, sélections immédiates, "not-first/not-last"). Il est d'une grande généralité puisqu'il permet aussi de fournir des règles déduisant des dates de début intermédiaires interdites (dans son expression la plus générale, la règle d'inconsitance locale conduit à la création de trous dans le domaine de st_i).

2.3 Programmation linéaire

2.3.1 Définitions

Programme linéaire

Un *Programme Linéaire* (PL) est un problème d'optimisation consistant à maximiser (ou minimiser) une *fonction objectif linéaire* de variables soumises à un ensemble de contraintes exprimées sous forme d'*équations ou d'inéquations linéaires*.

La forme générale d'un programme linéaire est la suivante :

$$(P_0)\begin{cases} Max(Min)\ z = & \sum_{j=1}^{n} c_j x_j & & & \text{(fonction objectif)} \\ \text{s.c.} & \sum_{j=1}^{n} a_{ij} x_j & \leq & b_i \quad i \in I^- \subseteq \{1,...,m\} & \text{(contraintes inégalité)} \\ & \sum_{j=1}^{n} a_{ij} x_j & \geq & b_i \quad i \in I^+ \subseteq \{1,...,m\} & \text{(contraintes inégalité)} \\ & \sum_{j=1}^{n} a_{ij} x_j & = & b_i \quad i \in I^0 \subseteq \{1,...,m\} & \text{(contraintes égalité)} \\ & x_j \geq 0 & & j = 1,...,n & \text{(contraintes de non-négativité)} \end{cases}$$

où les ensembles I^-, I^+ et I^0 sont disjoints, $I = I^- \cup I^+ \cup I^0 = \{1,...,m\}$ et où c_j, a_{ij} et b_i $(i = 1,...,m\ j = 1,...,n)$ sont des constantes supposées connues.

P_0 est un programme linéaire à n variables et m contraintes.

Notons qu'il n'est pas restrictif de poser des contraintes de non-négativité pour les variables x_j $(j = 1,...,n)$. En effet, toute variable x_j qui peut prendre une valeur réelle quelconque, positive, négative ou nulle, peut être remplacée par la différence $(x_j^+ - x_j^-)$ de deux variables x_j^+ et x_j^- astreintes, elles, à être non-négatives (voir par exemple [Minoux 83a]).

Forme standard

On dit qu'un programme linéaire est mis sous *forme standard* si sa fonction objectif est une fonction de *maximisation* et si toutes ses contraintes – en dehors des contraintes de non-négativité – sont des *égalités*. Un programme linéaire sous forme standard se présente donc comme suit :

$$(P_1)\begin{cases} Max\ z = & c.x \\ \text{s.c.} & A.x = b \\ & x \geq 0 \end{cases}$$

avec :

- n : nombre de variables $(x = (x_1,...,x_n)^\top)$,
- m : nombre de contraintes $(m \leq n)$,
- $A = (a_{ij})_{\substack{i=1,...,m \\ j=1,...,n}}$: Matrice des contraintes $(m \times n,\ rang(A) = m)$,
- $c = (c_1,...,c_n)$: vecteur ligne des profits (ou gains),
- $b = (b_1,...,b_m)^\top$: vecteur colonne des seconds membres.

Notons que l'on peut toujours considérer que l'on a $rang(A) = m$. En effet, $rang(A) < m$ signifie qu'une ou plusieurs lignes de la matrice A sont une combinaison linéaire des autres

lignes. Ceci revient à dire que les contraintes correspondantes sont soit redondantes, soit incompatibles avec les autres (suivant la valeur des seconds membres b_i) ; dans le premier cas, on peut éliminer ces contraintes, dans le second cas, le système $A.x = b$ n'a pas de solution.

Tout programme linéaire peut s'exprimer sous forme standard (P_1), en ajoutant éventuellement certaines variables appelées *variables d'écart* (par exemple, une inéquation $a.x \leq b$ devient l'équation $a.x + s = b$, $s \geq 0$).

Dans la suite, nous nous intéressons uniquement aux programmes linéaires exprimés sous forme standard.

Solutions d'un programme linéaire

Une *solution* d'un programme linéaire est une affectation de valeurs aux variables du problème. Une solution est *réalisable* si elle satisfait toutes les contraintes du problème : $x = (x_1,...,x_n)^\top$ est une solution réalisable de (P_1) si et seulement si $A.x = b$ et $x \geq 0$. Une *solution optimale* est une solution réalisable, notée x^* pour laquelle la fonction objectif z prend sa valeur maximale z^*.

Considérons $A.x = b$, le système d'équations linéaires de (P_1). On appelle *base* (de l'espace des colonnes de A) toute sous-matrice carrée $m \times m$, inversible de A. Soit B une telle sous-matrice de A et N la sous-matrice de A telle que $A = (B|N)$ après permutation des colonnes de A ; on peut écrire :

$$A.x = b \iff (B|N).\begin{pmatrix} x_B \\ x_N \end{pmatrix} = b$$

où x_B sont les *variables de base* (associées à la base B) de (P_1) et x_N sont les *variables hors-base* de (P_1).

En multipliant ce système par B^{-1}, on obtient :

$$B^{-1}.(B|N).\begin{pmatrix} x_B \\ x_N \end{pmatrix} = B^{-1}.b \iff x_B + B^{-1}.N.x_N = B^{-1}.b$$

On peut alors déterminer l'ensemble des solutions du système $A.x = b$ en fixant arbitrairement les valeurs de x_N et en calculant les valeurs résultantes de x_B :

$$\begin{cases} x_N = & v \in \mathbb{R}^{n-m} \\ x_B = & B^{-1}.b - B^{-1}.N.v \end{cases}$$

La solution particulière obtenue en fixant à 0 les variables hors-base x_N est appelée *solution de base* (ou *solution basique*) associée à la base B ; elle est donc égale à :

$$x_B = B^{-1}.b \text{ et } x_N = 0$$

Une base est dite *dégénérée* si le vecteur $x_B = B^{-1}.b$ a une de ses composantes nulle.

Une solution de base (associée à une base B) est une *solution de base réalisable* si elle vérifie de plus les contraintes de non-négativité $x_B \geq 0$. Une base est *réalisable* si et seulement si sa solution basique l'est.

On montre (voir par exemple [Minoux 83a]) que, si (P_1) admet une solution optimale à valeur finie, alors il existe une base réalisable B^* telle que la solution de base associée x^* est optimale.

Géométriquement, l'ensemble X des solutions réalisables de (P_1) est un ensemble convexe $X = \{x|A.x = b,x \geq 0\}$, appelé *polyèdre des solutions* (ou polyèdre des contraintes), dont les sommets (ou points extrêmes) correspondent aux solutions de base réalisables. Si (P_1) admet une solution optimale à valeur finie, cette solution correspond donc à un sommet du polyèdre X.

Dualité

Considérons le programme linéaire (P_1) ; on associe à chaque contrainte i de (P_1) (i = 1 à m) une variable u_i positive, négative ou nulle, appelée *variable duale*. Le programme linéaire D_1 défini par :

$$(D_1) \begin{cases} Min\ w = & u.b \\ \text{s.c.} & u.A \geq c \\ & u \lesseqgtr 0 \end{cases}$$

où u est le vecteur ligne : $(u_1,...,u_m)$, est appelé le *dual* du programme (P_1) (par opposition, (P_1) est appelé le *primal*).

La matrice des contraintes de (D_1) est la transposée de la matrice des contraintes de (P_1), le vecteur des seconds membres de (D_1) est le vecteur des coûts de (P_1) et vice-versa.

De cette notion découle le **théorème de la dualité** (extrait de [Minoux 83a]) :

Étant donné un programme linéaire (P) et le programme dual (D) associé :

– Si (P) et (D) ont des solutions, alors chacun d'eux a une solution optimale et :

$$z^* = \ Max(P) = \ Min(D) = w^*$$

– Si l'un d'eux a un optimum non borné, l'autre n'a pas de solution.

Ce théorème est de grande portée théorique et pratique. D'autres propriétés et théorèmes liés à la dualité sont largement utilisés en programmation linéaire. Nous renvoyons à la lecture d'ouvrages spécialisés, par exemple [Minoux 83a, Teghem 96, Goldfarb & Todd 89], pour une présentation complète de ces notions et résultats.

2.3.2 Résolution des programmes linéaires

2.3.2.1 Principaux algorithmes

Aujourd'hui encore la plus utilisée des méthodes de résolution en programmation linéaire, la méthode du simplexe, développée en 1947 par G. Dantzig, a longtemps été à la base de la plupart des algorithmes de résolution des programmes linéaires. Cependant, s'il se révèle très efficace *en moyenne*, Klee et Minty ont mis en évidence en 1972 des exemples pour lesquels le simplexe nécessite un nombre d'opérations croissant exponentiellement avec la taille du problème [Klee & Minty 72]. Cette possibilité théorique d'avoir un mauvais comportement du simplexe a suscité un regain d'intérêt pour la recherche d'algorithmes polynomiaux [1] de résolution des programmes linéaires.

1. Un algorithme est polynomial si on peut borner le temps maximal de son exécution par une fonction polynomiale de la taille du problème considéré.

Le premier algorithme de ce type, appelé *méthode ellipsoïde*, a été développé en 1979 par Khachian, montrant alors que les problèmes de programmation linéaire sont polynomiaux [Khachian 79]. Il s'agit d'une variante des méthodes de dilatation d'espace dues à Shor (cf. [Minoux 83a], chap.4) ; il s'inspire également de travaux antérieurs sur des algorithmes dits *ellipsoïdaux*. L'idée est d'utiliser une suite d'ellipsoïdes de volume décroissant mais qui contiennent à chaque itération la solution optimale du programme linéaire à résoudre. On peut trouver une présentation détaillée de cet algorithme dans [Goldfarb & Todd 89]. Bien que plus rapide que le simplexe sur les problèmes de Klee et Minty, il reste bien plus lent sur les problèmes réels.

Karmakar a développé un deuxième algorithme polynomial en 1984 [Karmarkar 84]. Il s'agit d'une méthode de *points intérieurs* qui se base sur des principes de géométrie projective et de programmation non linéaire. L'idée est de chercher des points, à l'intérieur du polyèdre des contraintes, permettant de se diriger rapidement vers le sommet optimal. Des dérivés de cet algorithme, en particulier la méthode dite *de barrière*, commencent à concurrencer le simplexe sur certains problèmes de grande taille. Pour plus de détails sur cet algorithme, on peut se référer à des ouvrages spécialisés tels que [Goldfarb & Todd 89].

Le simplexe reste cependant la méthode privilégiée de résolution des programmes linéaires formulés dans la pratique. Ceci est dû à ses performances et également à sa capacité à fournir des solutions de base, très importantes dans les approches de décomposition ou encore dans des procédures de ré-optimisation itératives (cf. [Lebbar 00]).

Nous présentons un aperçu de la méthode du simplexe au paragraphe suivant.

2.3.2.2 Méthode du simplexe

La méthode du simplexe est une méthode exacte et itérative dans laquelle on retrouve les concepts définis au paragraphe 2.3.1. En effet, d'un point de vue géométrique, étant donné qu'une solution d'un programme linéaire se trouve toujours sur un point extrême du polyèdre des contraintes, le simplexe consiste à se déplacer d'un point extrême à l'autre, le long des arêtes du polyèdre, jusqu'à trouver le point associé à la solution optimale.

Algébriquement, le simplexe s'interprète comme la détermination d'une suite de bases adjacentes réalisables telles que les valeurs associées de la fonction objectif soient croissantes.

Considérons le programme linéaire (P_1) et soit B une base réalisable (non dégénérée) issue de la matrice des contraintes. On décompose la fonction objectif :

$$z = c.x = c_N.x_N + c_B.x_B$$

En effectuant le changement de variable :

$$x_B = B^{-1}.b - B^{-1}.N.x_N$$

on peut exprimer z en fonction de x_N :

$$z = (c_N - c_B.B^{-1}.N)x_N + c_B.B^{-1}.b$$

Pour la solution de base associée à B, on a $z = c_B.B^{-1}.b$ (car $x_N = 0$).

On pose $\pi = (\pi_1,...,\pi_m)$, défini par :

$$\pi = c_B.B^{-1}$$

le vecteur des *multiplicateurs du simplexe* et :

$$\overline{c}_N = c_N - \pi.N$$

le vecteur des *profits marginaux* (ou *réduits*) des variables hors base. Les profits marginaux donnent le gain obtenu en augmentant de 1 la variable associée.

S'il existe une variable hors base x_r dont le profit marginal $\overline{c}^r_N$ est strictement positif, alors il suffit d'augmenter sa valeur dans la solution de base pour augmenter la valeur de z. La base B courante n'est donc pas optimale ; on augmente alors la variable hors base x_r, actuellement à 0, jusqu'à annuler une variable de base. À la suite de cette opération de pivotage, la variable hors base augmentée, dite *entrante* (dans la base), remplace la variable de base annulée, dite *sortante* : le problème est exprimé dans une nouvelle base B', adjacente à B. C'est le principe d'itération de la méthode du simplexe. Géométriquement, cette opération revient à passer sur un sommet adjacent du polyèdre.

Si on a, au contraire, $\overline{c}_N \leq 0$, alors toute augmentation de variable diminuerait la valeur de z; la valeur optimale (le maximum) de z est donc obtenue pour la solution de base $(B^{-1}.b|0)$. Notons que le vecteur π est alors la solution optimale du problème dual (D_1) de (P_1) ($u^* = \pi =$ $(\pi_1,...,\pi_m)$, $(\pi_1,...,\pi_m)$ sont les *valeurs duales* de (P_1)).

Sous l'hypothèse de non-dégénérescence des bases, la convergence est garantie en un nombre fini d'itérations, du fait du nombre fini de sommets d'un polyèdre.

Comme nous l'avons déjà dit, la complexité du simplexe est théoriquement exponentielle. En pratique, des études statistiques ont cependant montré qu'il converge généralement en m à $3m$ itérations (m étant le nombre de contraintes du problème). C'est ce qui en fait une méthode très efficace et très utilisée pour la résolution de problèmes réels.

Le simplexe permet également d'indiquer l'existence d'une infinité de solutions optimales ainsi que le cas où il n'y a pas de solution optimale à valeur finie. Pour plus de détails sur la méthode du simplexe, notamment sur la prise en compte de ces cas spéciaux et sur l'étude de sa complexité, on peut se référer à des ouvrages traitant de la programmation linéaire tels que [Minoux 83a, Teghem 96, Goldfarb & Todd 89].

2.3.3 Programmation linéaire en nombres entiers

De nombreux problèmes d'optimisation, issus de domaines d'applications très divers, peuvent être modélisés comme des programmes linéaires dont les variables sont contraintes à être entières.

Les programmes linéaires ainsi obtenus sont appelés *programmes linéaires en nombres entier*s (PLNE). Ils sont souvent difficiles à résoudre, du fait notamment que l'espace de recherche n'est plus convexe mais *discret*. Toutefois, la programmation linéaire en nombres entiers est déjà un outil très utile de modélisation des problèmes et de nombreuses approches de résolution ont été développées.

La formulation générale d'un PLNE est la suivante :

$$(P_2) \begin{cases} Max\ z = & c.x \\ \text{s.c.} & A.x \leq b \\ & x \in \mathbb{N}^n \end{cases}$$

En général, on suppose que A et b sont composés uniquement de valeurs entières.

On montre qu'un programme linéaire en nombres entiers peut toujours s'exprimer sous la forme d'un programme linéaire en variables booléennes (PL en $0-1$). Dans le cas particulier, fréquemment rencontré, où une partie des variables seulement sont astreintes à être entières, on parle de programme linéaire en variables mixtes (PL mixte).

Les méthodes classiques de programmation linéaire ne peuvent être utilisées systématiquement pour les PLNE puisqu'elles cherchent un sommet optimal du polyèdre des contraintes $\{x|A.x \leq b, x \geq 0\}$ qui, en général, n'a pas de coordonnées entières (cf. figure 2.2).

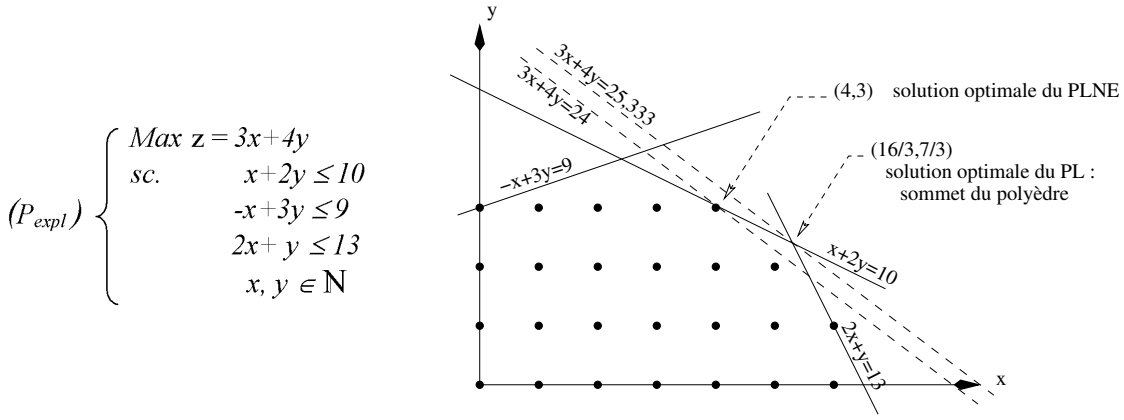


FIG. 2.2 – *Exemple de PLNE : 3 sommets du polyèdre des contraintes n'ont pas leurs coordonnées entières*

Pour certains problèmes particuliers, comme les problèmes de flot et plus généralement les problèmes dont la matrice des contraintes est totalement unimodulaire, la solution optimale de la relaxation linéaire du problème est entière. On peut donc utiliser des algorithmes classiques de programmation linéaire pour les résoudre. En dehors de ces cas particuliers, la solution optimale de relaxation linéaire d'un PLNE est quand même intéressante à calculer car elle fournit une borne supérieure (pour un problème de maximisation) de la valeur optimale du critère en variables entières.

Différentes approches ont été étudiées pour la recherche de solutions optimales entières des PLNE. Elles font appel essentiellement à des algorithmes de recherche arborescente par *séparation et évaluation* (*branch and bound*) et à des *méthodes de coupes*.

Un moyen naturel pour résoudre un PLNE de très petite taille consiste à calculer l'ensemble des solutions entières du problème et à en retenir la meilleure. Or, au delà de quelques variables, cette énumération *explicite* devient impossible du fait de la combinatoire du problème. L'idée de la recherche arborescente par séparation et évaluation est d'effectuer une énumération *implicite* des solutions du PLNE.

Cette méthode consiste à *séparer* le problème en plusieurs sous-problèmes, puis à *évaluer* chaque sous-problème en calculant une borne supérieure (en maximisation) de sa valeur optimale et en la comparant à une valeur de référence (correspondant à une solution connue du problème). On itère ensuite le processus uniquement sur les sous-problèmes dont l'évaluation est supérieure à la valeur de référence (ils sont seuls susceptibles de l'améliorer). L'ensemble des solutions est ainsi représenté par une arborescence dans laquelle un grand nombre de noeuds sont éliminés. L'évaluation des sous-problèmes se fait souvent en résolvant (par le simplexe) leur relaxation linéaire. La figure 2.3 donne un exemple de recherche arborescente par

séparation et évaluation, pour le PLNE présenté à la figure 2.2.

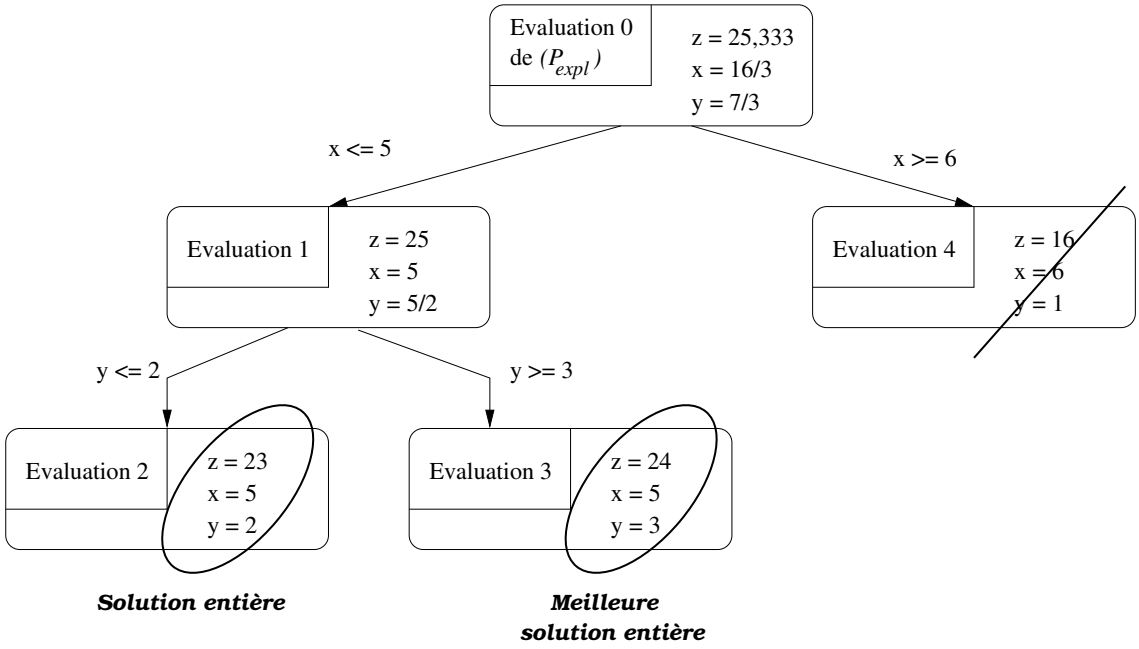


FIG. 2.3 – *Recherche par séparation et évaluation appliquée à* (P_{expl})

Les méthodes de coupes ont pour but de trouver l'enveloppe convexe des solutions entières, c'est-à-dire le plus petit polyèdre contenant toutes les solutions entières du PLNE. En effet, si on arrive à déterminer cette enveloppe convexe, la résolution de la relaxation linéaire du PLNE réduit à cet ensemble donne une solution optimale entière. En pratique, ces méthodes consistent à itérer le processus suivant jusqu'à obtenir une solution entière : résoudre la relaxation linéaire du problème puis ajouter des contraintes (les *coupes*) qui réduisent le polyèdre des solutions sans éliminer de solution entière (voir exemple de la figure 2.4).

La difficulté de ces méthodes réside dans la génération de coupes efficaces. Les plus connues sont les coupes de Gomory ; on prouve la convergence finie de la méthode avec ces coupes [Gomory 58], mais en pratique, ce ne sont pas les plus utilisées. La convergence est en effet très lente et on leur préfère en général des coupes basées sur une étude poussée de la structure du problème.

De plus, les méthodes de coupe, utilisées seules, affichent des performances modestes. En revanche, elles se révèlent efficaces associées à des méthodes de recherche arborescente (*Branch and cut*).

Pour une présentation plus détaillée de la programmation linéaire en nombres entiers, on peut se référer à des ouvrages de synthèse tel que [Minoux 83b] ou [Guéret *et al.* 00], ou bien à des ouvrages entièrement consacrés à ce domaine comme [Nemhauser & Wolsey 89], ou [Wolsey 98]. Ces livres abordent en particulier des aspects de modélisation ; les différentes techniques existantes font de la programmation linéaire en nombres entiers un outil puissant de modélisation, ce qui participe pour beaucoup à son succès.

Concernant plus particulièrement les méthodes de coupes, dans [Marchand *et al.* 99], les auteurs exposent un tour d'horizon très détaillé des différentes techniques existantes.

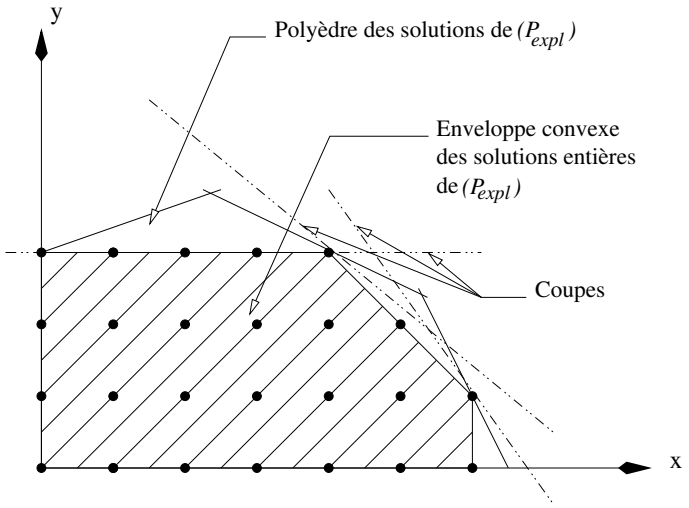


FIG. 2.4 – *Exemple de coupes appliquées au polyèdre des solutions de* (P_{expl})

Enfin, dans [Johnson *et al.* 00], les auteurs présentent les principaux progrès apportés, ces dix dernières années, en matière de programmation linéaire en nombre entiers. Ils citent, entre autres, les techniques de *preprocessing*, permettant de réduire l'espace de recherche des solutions avant la résolution d'un problème, ainsi que les procédures de *branch and price*, que nous présentons dans la suite de ce chapitre.

2.4 Génération de colonnes pour les problèmes de grande dimension

De nombreux problèmes d'optimisation combinatoire issus d'applications réelles peuvent être modélisés comme des PLNE, mais leur taille, ainsi que leur structure empêchent fréquemment une résolution par les méthodes précédemment présentées de programmation linéaire.

En effet, la relaxation linéaire de ces modèles fournit souvent une borne de mauvaise qualité qui, ajoutée à la combinatoire du problème, rend la recherche arborescente par séparation et évaluation d'une solution optimale entière, très difficile.

Une formulation alternative de ces problèmes, obtenue par une décomposition du modèle initial par la méthode de *Dantzig-Wolfe*, peut fournir de meilleures bornes. Le modèle ainsi obtenu est alors de trop grande taille pour être totalement explicité et donc résolu par un algorithme tel que le simplexe, mais on peut se ramener à une suite de sous-problèmes plus faciles à résoudre que le problème global et dont les liens de dépendance sont traités à un niveau supérieur de coordination.

Cette méthode de résolution des programmes linéaires de grande taille, appelée *génération de colonnes*, fait l'objet des paragraphes suivants.

2.4.1 Principe de résolution par génération de colonnes

La génération de colonnes, ou *programmation linéaire généralisée*, permet de résoudre certains programmes linéaires contenant un nombre de variables tel qu'il interdit l'application

de l'algorithme du simplexe au problème dans sa globalité. Le principe de cette technique consiste à ne manipuler qu'un petit nombre de variables à la fois et à identifier les variables entrant en base au cours de la résolution sans les énumérer explicitement.

Considérons (P_3), le PL sous forme standard suivant :

$$(P_3)\begin{cases} Max\ z = & c.x \\ \text{s.c.} & A.x = b \\ & x \geq 0 \end{cases}$$

On suppose que (P_3) comporte un nombre de contraintes "raisonnable" et un nombre de variables très largement supérieur ($n \gg m$), tel que la matrice A ne peut pas être explicitée. On suppose cependant que A est connue *implicitement*, c'est-à-dire que ses colonnes correspondent aux éléments d'un ensemble que l'on sait caractériser.

On suppose également que l'on dispose d'un algorithme efficace (appelé *algorithme générateur*) pour exhiber une colonne de A maximisant une fonction linéaire de la forme $z(A_j) = c_j - \pi.A_j$, où $\pi = (\pi_1,...,\pi_m)$ est un vecteur-ligne quelconque.

Soit Ω un sous-ensemble des variables de (P_3) et A' la sous-matrice de A correspondante (on suppose que A' est de rang m). On note (Pr_3) la restriction de (P_3) à Ω.

La résolution par le simplexe (cf. §2.3.2.2) de (Pr_3) fournit une solution de base optimale $(B^{*-1}.b|0)$ associée à une base B^* issue de A', de rang m.

Cette solution est admissible mais pas nécessairement optimale pour (P_3) . Pour l'améliorer, en suivant la méthode du simplexe, on cherche donc à faire entrer en base une variable de (P_3) de profit marginal strictement positif (en maximisation). On sait qu'une telle variable n'existe pas dans Ω (sinon $(B^{*-1}.b|0)$ n'est pas optimale pour (Pr_3)) ; il faut donc exhiber une des variables non explicitées dans (P_3) et *générer* la colonne de A correspondante.

Or, on sait que le profit marginal d'une variable x_j (hors base) est défini par :

$$\overline{c}_j = c_j - \pi.A_j$$

où π est le vecteur des multiplicateurs du simplexe associés à B^* ($\pi = c_B^*.B^{*-1}$), c'est-à-dire le vecteur des valeurs duales de (Pr_3).

On peut donc déterminer, en utilisant l'*algorithme générateur*, la colonne A_r de A telle que :

$$\overline{c}_r = Max_j(z = c_j - \pi.A_j)$$

Tant que $\overline{c}_r$ est strictement positif, on ajoute la variable x_r à Ω et la colonne A_r à A' et on itère le processus en résolvant le nouveau programme linéaire (Pr_3). Lorsque l'algorithme générateur détermine $\overline{c}_r \leq 0$, l'algorithme s'arrête ; en effet, toutes les variables hors base ont alors un coût réduit négatif ou nul, la solution courante est donc maximale pour (P_3).

Comme le simplexe, la méthode de résolution par génération de colonnes converge (chaque colonne de A est générée au plus une fois) et est exacte.

Bien entendu, l'hypothèse faite sur l'existence d'un algorithme générateur efficace est une hypothèse forte et peut donner le sentiment que la difficulté du problème a seulement été masquée. Cependant, dans de nombreuses applications, les colonnes de la matrice A ont une structure naturelle particulière que l'on peut exploiter avantageusement dans un algorithme générateur.

L'exemple du problème de découpe industrielle illustre particulièrement bien l'efficacité de la génération de colonnes. Il s'agit d'une des premières applications de ces techniques, que l'on doit aux travaux de Gilmore et Gomory entre 1961 et 1965 [Gilmore & Gomory 61, Gilmore & Gomory 63, Gilmore & Gomory 65]. On considère un stock (supposé infini) de rouleaux de papier de largeur L standard ; l'objectif est de satisfaire une demande d'au moins b_i pièces de largeur l_i (pour $i = 1$ à m) tout en minimisant le nombre de rouleaux à découper. On associe une variable x_j à chaque patron de découpe possible d'un rouleau (on a ainsi un nombre potentiellement très grand de variables). On obtient une formulation du problème par un programme linéaire (en nombres entiers) où la variable x_j représente le nombre de rouleaux à découper selon le patron j et dont les coefficients a_{ij} de la matrice A des contraintes représentent la quantité de pièces de largeur l_i obtenues avec le patron j. On applique alors la technique de génération de colonnes à la relaxation linéaire de ce problème et l'algorithme générateur, qui consiste à exhiber des patrons de découpe, résout dans ce cas un problème de sac-à-dos. On trouve une description détaillée de ce problème dans un grand nombre d'ouvrages traitant de la génération de colonnes (par exemple [Lasdon 70, Goldfarb & Todd 89]).

De nombreux programmes linéaires peuvent être reformulés de façon à présenter une structure adaptée à la résolution par génération de colonnes. On utilise classiquement la méthode de décomposition de Dantzig-Wolfe que nous présentons dans la suite.

2.4.2 Convergence de la méthode

La rapidité de la méthode de génération de colonnes dépend du nombre d'itérations nécessaires jusqu'à la preuve de l'optimalité, ainsi que du temps de calcul consacré à chaque étape du processus de résolution.

Le rôle de l'algorithme générateur est, à chaque itération, de fournir une colonne améliorante ou de prouver qu'il n'en existe aucune. La rapidité de la méthode de génération de colonnes est donc largement tributaire de l'efficacité de l'algorithme générateur. N'importe quelle colonne de profit marginal positif (en maximisation) peut contribuer à améliorer la solution courante, il n'est donc pas toujours utile de résoudre exactement chaque sous-problème. L'algorithme générateur peut ainsi n'en résoudre qu'une restriction, ou bien faire appel à des méthodes approchées, afin de réduire les temps de calcul. Notons toutefois que pour sortir du processus itératif, on doit utiliser au moins une fois un algorithme exact pour prouver qu'il n'existe plus de colonne améliorante.

Une autre pratique courante d'accélération de la méthode de génération de colonnes consiste à générer plusieurs colonnes améliorantes à chaque itération [Lasdon 70] ; ceci peut permettre de réduire le nombre d'itérations nécessaires pour générer les colonnes de la base optimale du problème maître.

Des problèmes de convergence sont malgré tout fréquemment observés dans l'application de la méthode de génération de colonnes, liés au fait que les valeurs duales convergent vers leur optimum en oscillant de façon irrégulière. Différents travaux de recherche portent sur les techniques d'accélération de la convergence, consistant à contrôler au mieux ces valeurs duales. Parmi ces travaux, les méthodes de stabilisation de la procédure de résolution proposées par [Merle *et al.* 99] puis [Ben Amor 02] sont particulièrement efficaces ; elles se basent sur l'introduction de perturbations dans le problème maître, couplées avec des bornes des variables duales. Les résultats reportés dans ces travaux montrent d'importantes réductions du nombre d'itérations et des temps de calcul sur différents problèmes, ainsi que la possibilité de résoudre

des problèmes de plus grande taille.

Dans [Desaulniers *et al.* 99], les auteurs présentent un ensemble de stratégies d'accélération, intervenant dans les différentes phases du processus de résolution et permettant de traiter des problèmes réels de très grande taille, dans des délais raisonnables.

2.4.3 Décomposition de Dantzig-Wolfe (1960)

Dans les programmes linéaires qui modélisent des applications réelles comportant un grand nombre de variables et de contraintes, les matrices de contraintes sont généralement "très creuses" et les éléments nuls sont distribués de telle façon qu'ils forment de grandes sous-matrices.

Ceci est dû au fait que les activités associées aux variables ne sont pas toutes "directement" liées entre elles, mais que chacune ne participe qu'à quelques-unes des contraintes du problème. Ainsi on met en évidence des sous-systèmes fortement liés en interne et fortement découplés les uns des autres. Ceci correspond à une segmentation logique du problème en sous-problèmes, liés entre eux par un ensemble restreint de contraintes (*contraintes couplantes*) et donnant une structure *bloc-angulaire* aux matrices de contraintes, comme illustré figure 2.5. Dans cet exemple, la matrice des contraintes est composée de p blocs indépendants et d'un bloc A_0 de contraintes couplantes ; pour tout $k = 1$ à K, x^k est un sous-ensemble du vecteur variable x ($x = (x^1,x^2,...,x^K)$).

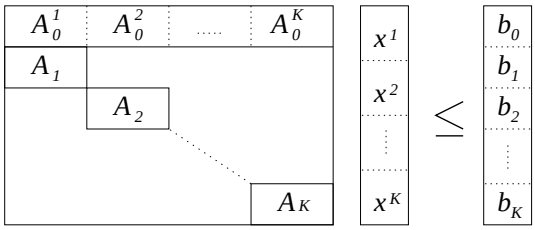


FIG. 2.5 – *Formulation des contraintes d'un PL avec une matrice bloc-angulaire*

La décomposition de Dantzig-Wolfe associe un polyèdre à chaque bloc A_k ($k = 1$ à K) et exprime les variables du problème comme combinaison linéaire des sommets de ces polyèdres, permettant ainsi une formulation équivalente du PL de départ, dans laquelle seules les contraintes couplantes sont explicitées.

Considérons un programme linéaire, (P_4), associé à la matrice de contraintes de la figure 2.5. Soit X^k le polyèdre (borné) défini par $X^k = \{x^k \mid A_k.x^k \leq b^k,\ x^k \geq 0\}$ pour tout $k = 1$ à K, (P_4) s'écrit alors (*formulation compacte* de (P_4)) :

$$(P_4) \begin{cases} Max\ z = & \sum_{k=1}^{K} c^k.x^k & \\ \text{s.c.} & \sum_{k=1}^{K} A_0^k.x^k \leq b_0 & \\ & x^k \in X^k & \forall k = 1..K \end{cases}$$

Pour tout $k = 1$ à K, soit Ω^k l'ensemble (fini) des indices des sommets x_s^k du polyèdre X^k. Tout vecteur x^k de X^k peut s'exprimer comme combinaison linéaire de ces sommets ; ie.

il existe un ensemble de scalaires λ_s^k tel que :

$$x^k = \sum_{s \in \Omega^k} \lambda_s^k . x_s^k$$

avec $\sum_{s \in \Omega^k} \lambda_s^k = 1$ et $\lambda_s^k \in [0;1]$ pour tout $s \in \Omega^k$.

On peut donc écrire le problème (P_4) sous la forme :

$$(PM_4) \begin{cases} Max\ z = & \sum_{k=1}^{K} \sum_{s \in \Omega^k} \lambda_s^k . c^k . x_s^k & \\ \text{s.c.} & \sum_{k=1}^{K} \sum_{s \in \Omega^k} \lambda_s^k . A_0^k . x_s^k \leq b_0 & \\ & \sum_{s \in \Omega^k} \lambda_s^k = 1 & \forall k = 1..K \\ & \lambda_s^k \in [0;1] & \forall k = 1..K \end{cases}$$

Dans cette nouvelle formulation, totalement équivalente à (P_4), les seules variables de décision sont les λ_s^k. En effet, même s'ils ne sont pas explicités, on sait caractériser les sommets des polyèdres X^k, ils sont donc implicitement connus. Comme illustré à la figure 2.6, la matrice des contraintes de (PM_4) contient un nombre restreint de lignes mais beaucoup plus de colonnes que la matrice initiale et ces colonnes sont implicitement connues.

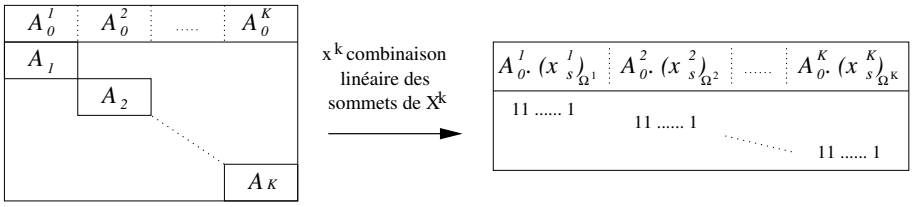


FIG. 2.6 – *Transformation de la matrice bloc-angulaire des contraintes*

Ainsi, le programme linéaire (PM_4) a la forme requise pour être résolu par génération de colonnes.

On appelle *problème maître* le programme linéaire (PM_4). À chaque itération du processus de résolution par génération de colonnes, on résout une restriction du problème maître à un sous-ensemble des variables ((PMR_4), *problème maître restreint*). On cherche ensuite à faire entrer en base de nouvelles variables, de profit marginal positif. Ceci revient, pour l'algorithme générateur, à chercher les sommets x_s^k des K polyèdres X^k qui maximisent la valeur

$$c^k . x^k - \pi . \begin{pmatrix} A_0^k \\ 1 \end{pmatrix} . x^k = x^k (c^k - \pi . \begin{pmatrix} A_0^k \\ 1 \end{pmatrix})$$

où π est le vecteur des valeurs duales de (PMR_4), c'est-à-dire à résoudre les K *sous-problèmes* :

$$(SP_k) \begin{cases} Max\ z = & \sum_{s \in \Omega^k} x_s^k (c_s^k - \pi . \binom{A_0^k}{1}) \\ \text{s.c.} & A_k . x^k \leq b_k \\ & x^k \geq 0 \end{cases}$$

Le processus itératif s'arrête lorsqu'il n'existe plus de solution strictement positive à ces problèmes. La solution optimale du problème maître restreint courant (PMR_4) est alors la solution optimale du problème maître (PM_4) et elle fournit également la solution de (P_4).

Ainsi, l'approche de résolution d'un programme linéaire par génération de colonnes peut être vue comme un schéma itératif de coopération entre deux types de problèmes plus faciles à résoudre que le problème global :

- le *problème maître*, qui est une formulation équivalente du programme linéaire initial, avec moins de contraintes et plus de variables, et dont on ne résout à chaque itération qu'une *restriction* à un sous-ensemble de variables ;
- le *problème esclave*, constitué d'un ensemble de *sous-problèmes* indépendants, résolus à chaque itération par un *algorithme générateur* et qui consiste à générer, si elle existe, une colonne qui, insérée à la matrice du problème maître restreint courant, permet d'améliorer la solution courante.

La figure 2.7 illustre ce schéma de coopération.

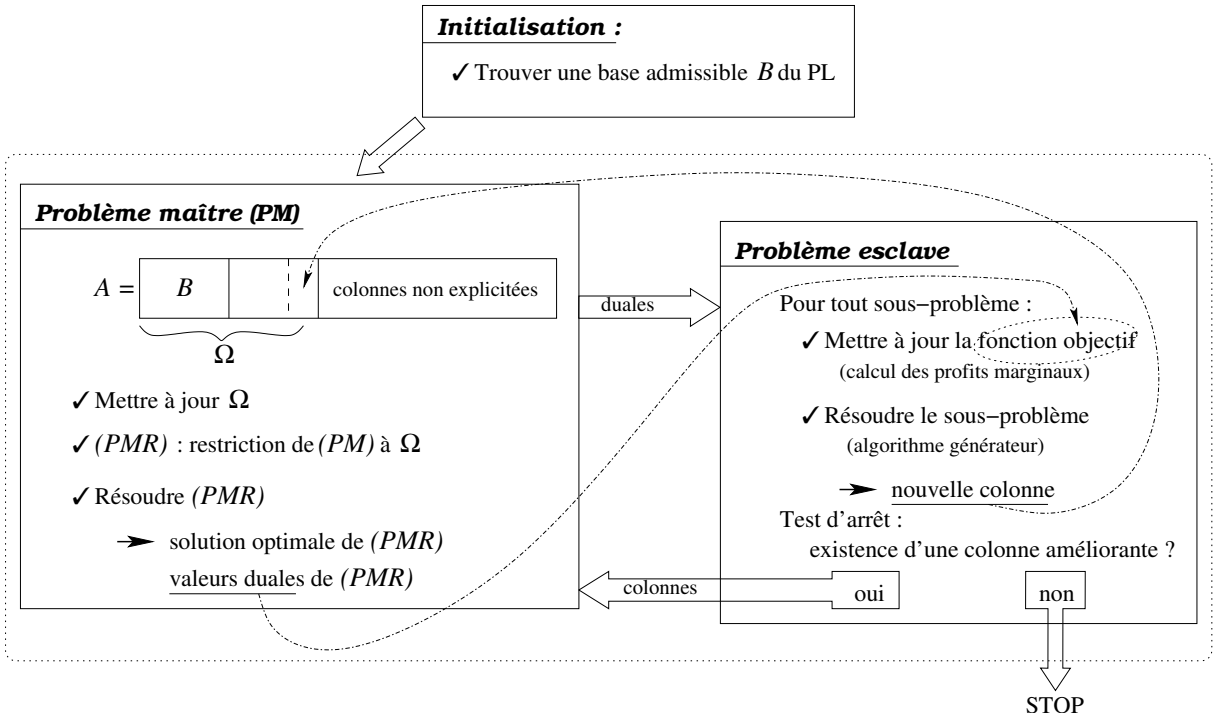


FIG. 2.7 – *Schéma de résolution d'un PL par génération de colonnes*

Une approche duale de la décomposition de Dantzig-Wolfe, appelée *décomposition de Benders*, consiste à partitionner les variables d'un PL de façon à obtenir des sous-problèmes indépendants, liés entre eux uniquement par un ensemble de variables couplantes [Benders 62].

On peut appliquer la génération de colonnes au dual d'un problème décomposé de la sorte, ce qui revient à développer une méthode de *génération de contraintes* dans le problème primal. Ce type de décomposition trouve ses applications particulièrement dans des problèmes en variables mixtes, ou des problèmes partiellement non linéaires (voir [Minoux 83b] §4.4 et [Lasdon 70] §7.3 pour une présentation détaillée de la méthode par décomposition de Benders et pour sa comparaison avec la méthode de Dantzig-Wolfe).

Notons enfin que la génération de colonnes peut être également vue comme une version duale de la relaxation Lagrangienne. Nous ne présentons pas cette méthode ici, mais on peut

en trouver une description détaillée ainsi que des éléments de comparaison avec la génération de colonnes dans différents ouvrages, en particulier dans [Minoux 83b].

2.4.4 Résolution de PLNE par génération de colonnes

2.4.4.1 Décomposition des PLNE

Nous avons vu que la méthode de Dantzig-Wolfe permet de décomposer des programmes linéaires et de les résoudre par génération de colonnes ; en pratique, cette méthode est très utilisée pour la résolution de très grands programmes linéaires en nombres entiers ou en variables mixtes.

En effet, la décomposition de Dantzig-Wolfe est adaptable aux PLNE. Considérons le problème (P'_4) défini comme le programme linéaire (P_4) auquel on ajoute la contrainte d'intégrité des variables. On a alors la formulation compacte de (P'_4) :

$$(P'_4)\begin{cases} Max\ z = & \sum_{k=1}^{K} c^k.x^k & \\ \text{s.c.} & \sum_{k=1}^{K} A_0^k.x^k \leq b_0 & \\ & x^k \in X^k & \forall k = 1..K \end{cases}$$

avec $P^k = \{x^k \mid A_k.x^k \leq b^k,\ x^k \geq 0\}$ et $X^k = P^k \bigcap \mathbb{N}^n$, pour tout $k = 1$ à K. Supposons P^k borné, on sait alors qu'il existe un ensemble fini de points (entiers) $\{x_q^k\}_{q \in Q^k} \subseteq X^k$, tel que tout $x^k \in X^k$ est combinaison linéaire des x_q^k [Vanderbeck 00].

On peut ainsi formuler le problème maître suivant :

$$(PM'_4)\begin{cases} Max\ z = & \sum_{k=1}^{K} \sum_{q \in Q^k} \lambda_q^k.c^k.x_q^k & \\ \text{s.c.} & \sum_{k=1}^{K} \sum_{q \in Q^k} \lambda_q^k.A_0^k.x_q^k \leq b_0 & \\ & \sum_{q \in Q^k} \lambda_q^k = 1 & \forall k = 1..K \\ & \lambda_q^k \in \{0,1\} & \forall k = 1..K \end{cases}$$

Ce programme linéaire contient des variables discrètes mais sa relaxation linéaire a la forme requise pour être résolue par génération de colonnes.

De nombreux cas d'application de la génération de colonnes à des problèmes combinatoires sont reportés dans la littérature. [Lübbecke & Desrosiers 02] dresse un tableau synthétique des différents types de problèmes qui ont été résolus avec succès par ces méthodes ; on relève en particulier des problèmes de découpe, avec [Gilmore & Gomory 61, Gilmore & Gomory 63] et plus récemment [Vance 98, Vanderbeck 00]), des problèmes d'affectation et d'ordonnancement de tâches [Savelsbergh 97, Van Den Akker *et al.* 00], des problèmes de tournée avec contraintes supplémentaires – fenêtres de temps et contraintes de ressources – [Desrosiers *et al.* 84, Desrochers *et al.* 92, Ribeiro & Soumis 94] et des problèmes de construction de plannings dans le domaine des transports aériens [Desaulniers *et al.* 97, Barnhart *et al.* 98a].

2.4.4.2 Algorithme générateur

Les sous-problèmes issus de la décomposition de Dantzig-Wolfe d'un PLNE sont également formulés comme des PLNE.

Pour les applications relevées dans la littérature, les colonnes du problème maître peuvent être généralement interprétées comme des chemins dans un graphe ou des flots dans un réseau. Les algorithmes générateurs font souvent appel à la théorie des graphes et la programmation dynamique et, plus rarement, à d'autres méthodes comme la programmation linéaire ou la programmation non linéaire (cf. [Lasdon 70]).

Le sous-problème se ramène fréquemment à un problème de plus court chemin dans un graphe avec contraintes supplémentaires. Les algorithmes générateurs développés pour ce type de problème se basent essentiellement sur un algorithme de programmation dynamique, pseudo-polynomial, présenté dans [Desrochers & Soumis 88, Desrochers *et al.* 92] pour la résolution par génération de colonnes d'un problème de tournée de bus scolaires (avec fenêtres temporelles et contraintes de ressources). Le principe de cet algorithme est d'associer à chaque chemin partiel du graphe une étiquette fournissant une évaluation multi-critère (sur le temps et la consommation des différentes ressources) et d'éliminer des chemins partiels grâce à des règles de dominance. C'est une extension de l'algorithme classique de Bellman-Ford pour la recherche de plus courts chemins dans un graphe [Prins 94].

Des méthodes approchées peuvent également être utilisées pour générer les colonnes du problème maître ; elles se basent généralement sur les mêmes principes mais se contentent de rechercher un chemin réalisable dans le graphe, satisfaisant l'ensemble des contraintes du sous-problème. Cependant, à une itération donnée, le fait de ne pas trouver de colonne améliorante, par un algorithme heuristique, ne prouve pas l'optimalité de la solution courante du problème maître. Pour terminer le processus itératif, on doit donc utiliser un algorithme générateur exact.

2.4.4.3 Recherche de solutions entières : "branch and price"

La génération de colonnes appliquée à un problème maître en nombres entiers (PM'_4) fournit une solution optimale de sa relaxation linéaire qui, si elle n'est pas entière, peut être ensuite utilisée comme borne (supérieure en maximisation) de la solution entière de (PM'_4) . Cette borne est souvent rapportée comme étant de bonne qualité (en particulier, par rapport à celle que l'on peut obtenir par la relaxation linéaire de la formulation initiale de (P'_4)). Ceci est dû en particulier au fait que dans la relaxation linéaire de (PM'_4), contrairement à celle de (P'_4), on réduit implicitement les ensembles X^k à leur enveloppe convexe $conv(X^k)$ pour la recherche des solutions.

Une première approche pour la recherche d'une solution entière consiste à obtenir par génération de colonnes la solution du problème maître (PM'_4), puis à lancer une recherche arborescente classique de type branch and bound sur le problème maître restreint à l'ensemble Ω des colonnes explicitées lors du processus de génération de colonnes.

C'est une approche heuristique car, même si l'ensemble de colonnes Ω permet de trouver la solution optimale continue de (PM'_4) à la racine de l'aborescence, on n'est pas assuré d'obtenir la solution optimale (ni même une solution réalisable) aux autres nœuds de l'arbre. En effet, des colonnes, hors Ω, sont susceptibles d'améliorer la solution (ou de rendre le problème faisable) au cours de la recherche arborescente.

Ainsi, une deuxième approche, appelée *branch and price*, adapte les principes du branch and bound au contexte de la génération de colonnes en permettant d'expliciter de nouvelles colonnes à chaque nœud de l'arbre, c'est-à-dire en utilisant une procédure d'évaluation des

nœuds par la méthode de génération de colonnes.

Les techniques de séparation doivent alors permettre de supprimer des solutions fractionnaires, tout en n'introduisant que des contraintes qui puissent être prises en compte aisément par l'algorithme générateur.

[Barnhart *et al.* 98b, Vanderbeck & Wolsey 96, Vanderbeck 00] détaillent les différents aspects théoriques du branch and price et discutent en particulier des types de séparation envisageables. La première stratégie est d'effectuer les séparations, comme classiquement dans un branch and bound, sur les variables du problème maître, c'est-à-dire sur les "choix" des colonnes dans la solution. Or, s'il est aisé d'ajouter dans chaque branche de l'arbre la contrainte appropriée au problème maître (par exemple $\lambda_q^k = 0$ dans une branche et $\lambda_q^k = 1$ dans l'autre), on doit modifier la structure du sous-problème associé pour qu'il intègre ce type de contrainte. Généralement, ceci dégrade considérablement l'efficacité de l'algorithme générateur. Une autre stratégie permet d'éviter cet écueil : elle consiste à séparer le problème sur les variables de la formulation initiale (P_4'). Ceci revient à effectuer la séparation sur les variables des sous-problèmes. Ainsi, à chaque nœud, on ajoute simplement des contraintes de borne sur les variables d'un sous-problème, ce qui ne modifie pas sa structure. La même procédure d'évaluation peut donc être appliquée à chaque nœud. Ainsi, on adopte généralement cette deuxième stratégie de séparation des variables.

La recheche arborescente par branch and price permet donc théoriquement de trouver la solution optimale d'un PLNE. Elle peut être également utilisée dans le cadre d'une approche heuristique pour obtenir des solutions de bonne qualité.

2.5 Conclusion

Dans ce deuxième chapitre, nous présentons les méthodes que nous privilégions pour aborder les problèmes de planification de missions spatiales étudiés par la suite.

Les techniques de Propagation de Contraintes ont pour but de simplifier la résolution d'un problème en tirant un maximum d'informations de ses contraintes, de façon à réduire le domaine des valeurs admissibles des variables et ainsi éviter de nombreuses tentatives de résolution vouées à l'échec. Suivant le type de contrainte considérée (contrainte temporelle, contrainte de partage de ressource), différentes règles de déductions peuvent être utilisées. La mise en œuvre de ces règles fait l'objet de nombreux développements algorithmiques.

La Propagation de Contraintes peut être utilisée en prétraitement d'un problème ou au cours d'un processus de résolution, de façon à restreindre l'espace de recherche des solutions. Elle peut également être vue, dans un contexte d'aide à la décision, comme un moyen de guider un utilisateur dans ses choix, d'une part en éliminant certaines valeurs de variables inconsistantes, d'autre part en répercutant les conséquences de chaque prise de décision sur le reste des décisions à prendre.

La Programmation Linéaire a pour objet l'étude théorique de problèmes d'optimisation de fonctions linéaires sous contraintes linéaires, ainsi que la conception d'algorithmes de résolution de ces problèmes. En particulier, l'algorithme du simplexe est généralement utilisé pour résoudre un problème d'optimisation linéaire.

La Programmation Linéaire propose un cadre formel permettant également de modéliser un grand nombre de problèmes d'optimisation en variables entières. Des méthodes utilisant la

relaxation linéaire de ces problèmes pour calculer des bornes, intégrées dans des processus de recherche arborescente adaptés aux problèmes en variables entières, permettent de résoudre ces problèmes (PLNE).

Pour des problèmes d'optimisation combinatoire de très grande taille, ou pour lesquels la relaxation linéaire ne fournit pas de bornes de qualité suffisante, une variante de la Programmation Linéaire, appelée Génération de Colonnes, basée sur une décomposition du modèle linéaire, permet de développer des méthodes de résolution efficaces.

Dans la partie suivante, nous nous appuyons sur le formalisme de la Programmation Linéaire pour proposer une modélisation de problèmes de planification émergeant dans trois projets spatiaux développés au CNES, que nous avons étudiés : un problème d'allocation de liens de communication dans le projet de constellation Skybridge, un problème de planification d'expériences et de communications dans le projet Netlander d'exploration de la planète Mars et un problème de planification de prises de vue pour un satellite Pléiades d'observation de la Terre. Par la suite, nous mettons en œuvre les méthodes vues dans ce chapitre pour résoudre ces problèmes.

Deuxième partie

Étude de trois projets spatiaux : analyse et modélisation des problèmes de planification

Chapitre 3

Skybridge : une constellation basse altitude pour la communication

3.1 Présentation du projet Skybridge

3.1.1 Constellations basse altitude de télécommunication

Afin de répondre à des besoins toujours croissants en matière de télécommunications, de nouveaux systèmes basés sur l'exploitation de constellations de satellites ont été proposés comme alternative ou complément des systèmes de télécommunication déjà existants. Ces nouveaux systèmes peuvent en particulier être utilisés pour un accès mondial haut-débit à des services multimédias interactifs.

Il existe plusieurs catégories de constellations de satellites correspondant aux différentes altitudes utilisées. Si les premiers systèmes satellitaires exploités étaient majoritairement géostationnaires (*i.e.,* évoluant sur une orbite équatoriale à une altitude de 36 000 kilomètres), les études se portent désormais vers les constellations à moyenne altitude (MEO : medium earth orbit, entre 5 000 et 15 000 kilomètres) et surtout à basse altitude (LEO : low earth orbit, entre 500 et 2 000 kilomètres).

En effet, outre le fait que l'orbite géostationnaire tend à devenir saturée, les constellations en orbite basse présentent plusieurs avantages techniques et financiers.

Tout d'abord, la taille, le poids, ainsi que les coûts de fabrication et de mise à poste des satellites MEO et LEO sont réduits par rapport à ceux d'un satellite géostationnaire. De plus, la propagation des signaux est jusqu'à 50 fois plus rapide avec des satellites en basse altitude qu'avec un satellite géostationnaire. Ceci permet donc de proposer de nouveaux types de services utilisant des transmissions quasi-instantanées des données. Enfin, contrairement aux satellites géostationnaires qui couvrent d'autant plus difficilement les régions qu'elles sont éloignées de la zone équatoriale, les constellations MEO et LEO assurent une couverture globale et de qualité homogène de toute la surface de la planète. Ce point est particulièrement important si l'on considère que le marché des télécommunications est principalement constitué de pays situés dans des latitudes autour de 45° dans les hémisphères nord et sud.

L'exploitation de constellations MEO ou LEO pour les télécommunications présente cependant quelques inconvénients. D'une part, du fait de leur faible altitude, ces constellations doivent comporter un très grand nombre de satellites pour assurer une couverture mondiale

à chaque instant. D'autre part, un satellite non-géostationnaire n'apparaît plus fixe vu d'un point au sol mais **défilant**, c'est-à-dire qu'un satellite n'est visible pour un utilisateur que sur certains intervalles temporels. Ceci induit une dynamique très spécifique dans le réseau de communication associé si l'on veut assurer des communications continues aux utilisateurs.

3.1.2 Architecture de la constellation Skybridge

Le système Skybridge est un projet de réseau d'accès mondial à des services de télécommunication interactifs à haute vitesse (accès Internet, visioconférences, télé-travail, ...). Il utilise une constellation LEO de 80 satellites, ainsi que 200 stations sol (appelées également **gateways**) permettant chacune la connexion d'utilisateurs et avec la possibilité de s'interconnecter entre elles lorsqu'elles sont en visibilité d'un même satellite (voir figure 3.1).

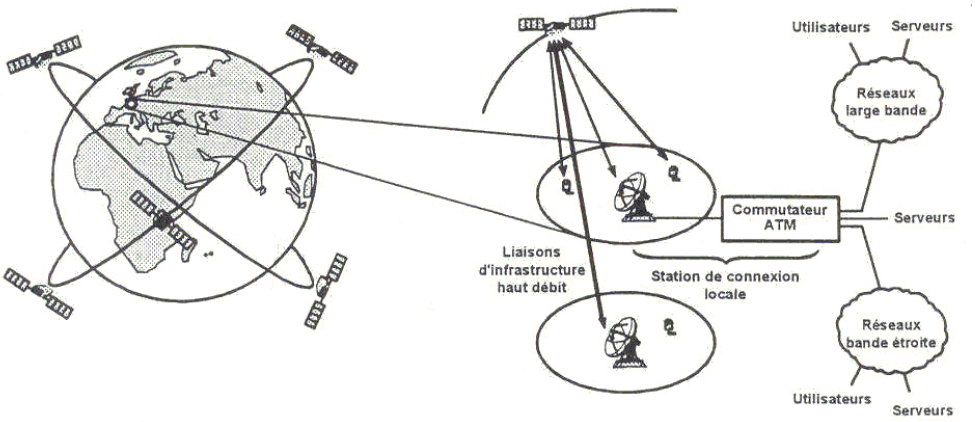


FIG. 3.1 – *Fonctionnement général du système Skybridge*

La constellation est de type Walker, c'est-à-dire qu'elle possède certaines propriétés géométriques telles que :

- tous les satellites à même altitude (1 457 kilomètres) et équirépartis sur leur orbite,
- des orbites circulaires, de même inclinaison par rapport au plan équatorial et équiréparties autour de la Terre.

Dans la constellation Skybridge les satellites sont répartis sur 20 plans orbitaux inclinés à 53° comptant chacun 4 satellites équidistants. La géométrie de cette constellation permet une couverture permanente dans la bande de latitudes ±68°. Dans les zones tempérées, elle permet une couverture par 2 à 6 satellites simultanément. Les satellites sont visibles d'un point de la Terre sur des intervalles temporels (ou fenêtres de visibilité) de 15 à 20 minutes.

Les satellites utilisés sont à **faisceaux fixes** ce qui signifie que leurs antennes sont mobiles et qu'ils sont ainsi capables, tout en se déplaçant sur leur orbite, de pointer une antenne sur une zone précise au sol. Ce type de matériel favorise le chevauchement entre les zones couvertes par les différents satellites et permet ainsi de s'adapter à la demande en pointant un maximum d'antennes de la constellation sur les zones où se concentre la demande.

Le rôle des satellites est de relier chaque utilisateur avec la station terrestre de connexion la plus proche par des antennes pointant vers les stations de connexion.

À Terre, les 200 stations sol sont réparties de façon à pouvoir assurer l'interconnexion de tous les utilisateurs avec les réseaux terrestres de communication (voir figure 3.2).

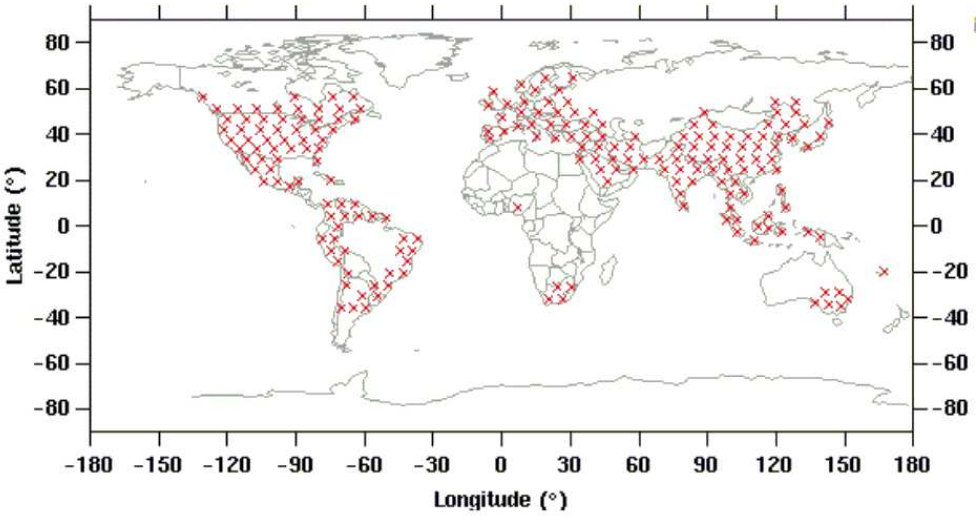


FIG. 3.2 – *Répartition des stations sol sur la Terre*

La surface terrestre couverte par le système Skybridge est divisée en **cellules**, qui sont des zones circulaires d'environ 350 kilomètres de rayon centrées sur une station sol. Tous les utilisateurs situés dans une même cellule se connectent au réseau via une même station sol. Le rôle d'une station sol est d'assurer l'interconnexion des utilisateurs de la cellule avec les réseaux terrestres de communication.

Une description détaillée des caractéristiques techniques du système Skybridge et de ses capacités de service est présentée dans [Sourisse & Sorre 97] [1].

3.2 Problématique

Nous avons vu que, du fait de leur faible altitude, les satellites d'une constellation comme Skybridge ne sont visibles pour un utilisateur ou pour une station sol, que sur certains intervalles temporels.

Lorsque l'on veut assurer la continuité d'une communication établie entre un utilisateur et sa station sol via un satellite, on doit donc basculer la communication du satellite qui s'apprête à sortir de la zone de visibilité à un satellite qui entre en visibilité. Ce basculement est appelé **handover** ; il s'agit d'une commutation de liaison transparente pour l'utilisateur. On doit effectuer un handover chaque fois qu'une liaison déjà établie ne peut plus être maintenue.

Le déroulement d'un handover dans le système Skybridge se décompose en quatre étapes illustrées dans la figure 3.3 :

– Étape 1 : l'utilisateur est en liaison avec le satellite 1 et attend un signal provenant de la station sol autorisant la liaison avec le satellite 2.

1. Voir également le site http://www.skybridgesatellite.com.

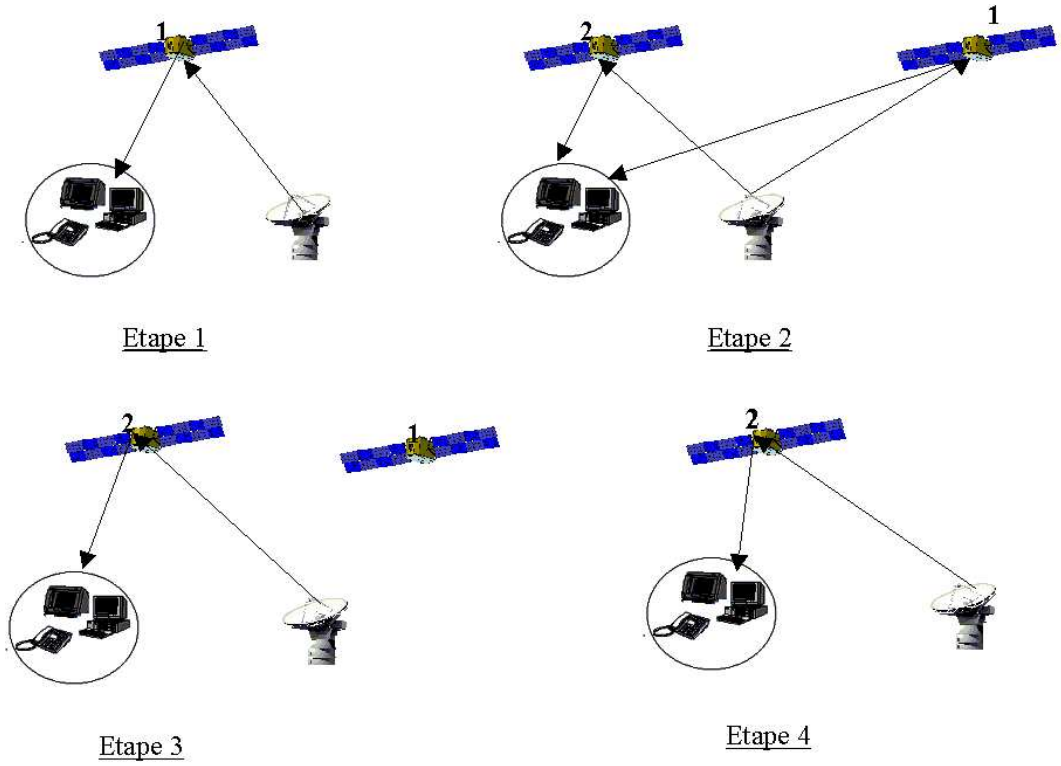


FIG. 3.3 – *Déroulement d'un handover*

- Étape 2 : l'utilisateur communique avec le satellite 1 et le satellite 2 pendant au moins une durée ΔH (**période de double visibilité**).
- Étape 3 : la liaison avec le satellite 1 est coupée dès que le signal provenant de la station n'est plus reçu par ce satellite.
- Étape 4 : l'utilisateur est en liaison avec le satellite 2.

De plus, pour des raisons de performance du système, on estime que la durée d'une liaison, hors réalisation des handovers, doit être supérieure à une durée minimale ΔT (de l'ordre de 120 secondes pour le système Skybridge). On peut ainsi schématiser le déroulement d'une communication entre un utilisateur et sa station sol via la constellation par la figure 3.4.

Pour assurer une couverture permanente de la Terre, chaque utilisateur doit pouvoir être connecté à chaque instant par au moins un satellite et par au moins deux satellites lors d'un handover. Pendant le handover, l'utilisateur utilise deux fois plus de ressources que nécessaire pour une simple liaison. De plus, les liens de communication établis pendant un handover sont de moins bonne qualité.

Dans le cadre d'études menées par le CNES sur l'établissement de plans prévisionnels d'utilisation des ressources d'une constellation comme Skybridge, on cherche donc à planifier l'allocation des liens de communication des stations sol et des utilisateurs avec les satellites de façon à assurer une couverture globale et continue de la Terre et en cherchant à minimiser le nombre de handovers réalisés.

La constellation étudiée est cyclique (cycles de durée 11 jours) et les calculs d'éphémérides des satellites mettent en évidence l'existence de sous-cycles au terme desquels on retrouve une

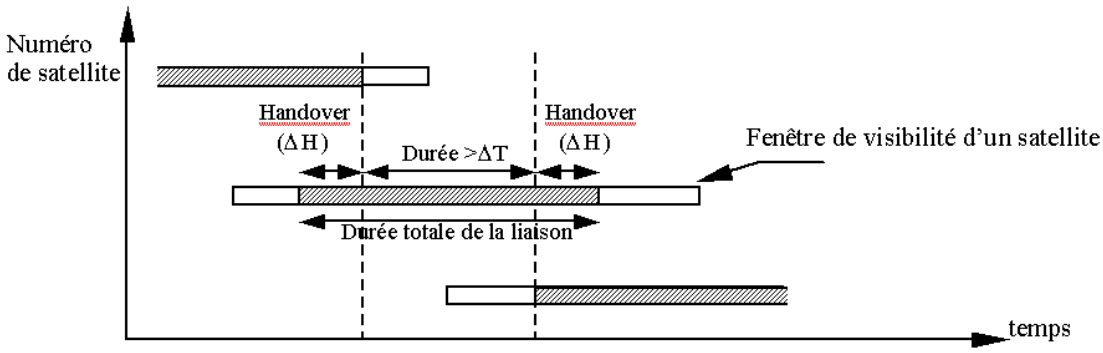


FIG. 3.4 – *Décomposition d'une communication*

même configuration de la constellation à une permutation près des satellites. Ces sous-cycles ont une durée d'environ 10 heures ; on considère donc le problème d'allocation des liens de communication sur un horizon de 10 heures.

Autant que possible, les handovers sont réalisés simultanément pour tous les utilisateurs d'une même cellule (**handovers de faisceau**); c'est-à-dire que tout le trafic d'une cellule via un satellite est commuté pendant une unique période de double visibilité (d'une durée au moins égale à $\Delta H = 60$ secondes). Ceci nous amène à considérer un problème simplifié en représentant l'ensemble des utilisateurs d'une cellule par un unique utilisateur situé au centre de la cellule, ce qui revient à ne s'intéresser qu'aux liaisons entre stations sol et satellites.

Nous supposons également que la demande est uniforme et qu'elle se traduit par le besoin d'établir un lien de communication utilisant une unique antenne d'un unique satellite à tout moment et dans toute cellule. Un satellite comporte un nombre fixe d'antennes ; le nombre de liens de communication que peut établir simultanément un satellite avec les stations sol est ainsi limité à ce nombre d'antennes.

Notre problème consiste donc, étant connues les fenêtres de visibilité sur un horizon de 10 heures des différents satellites pour les stations sol, à trouver des séquences d'allocation de liens stations/satellites qui assurent, pour chaque station sol, un lien continu avec la constellation et qui minimisent le nombre total de handovers réalisés.

3.3 Analyse du problème

La complexité du problème d'allocation des liens stations/satellites est due à différents facteurs. Tout d'abord, la dynamique de la constellation est telle que les conflits de ressources satellites, sont très nombreux puisqu'à chaque instant une station sol est en visibilité de plusieurs satellites et un satellite est visible pour plusieurs stations. De plus, les nombreuses contraintes temporelles auxquelles sont soumises les communications, ainsi que l'exécution des handovers, induisent que tout événement, tel que le choix d'effectuer un handover avec un satellite particulier, a des conséquences durant un intervalle de temps considérable et sur quasiment tout le système. Ainsi, un mauvais choix d'allocation d'un satellite à une station peut entraîner à plus ou moins long terme une discontinuité dans la communication, ce qui rend la solution non réalisable du fait de la contrainte forte de qualité de service à fournir. Trouver

une solution réalisable est donc déjà un problème difficile. Par ailleurs, le grand nombre de satellites et de stations sol ajouté à celui des combinaisons de choix de décisions possibles rendent le problème fortement combinatoire.

Depuis 1997, différentes méthodologies ont été étudiées et mises en œuvre au CNES pour traiter ce problème.

Le premier souci pour la résolution d'un tel problème a été de montrer l'existence de solutions réalisables avant même de chercher à optimiser certains critères tels que le nombre de handovers.

La première méthode a été de développer des heuristiques simples permettant de générer des solutions en considérant les stations sol les unes après les autres et en fixant des règles de priorité simples entre les satellites pour régler les conflits de ressources [Mancel 00]. Ces algorithmes ont permis d'obtenir des premières solutions pour un nombre restreint de stations sol. Ces stratégies aboutissent dans de nombreux cas à des situations de blocage, où un mauvais choix à un instant donné entraîne une rupture d'un lien de communication qui aurait pu être évitée.

Une approche par Réseaux de Petri a été étudiée en 1999 [Dafir 99]. Nous avons vu (cf. chapitre 1) que les Réseaux de Petri sont bien adaptés pour exprimer les conflits entre ressources, les contraintes de relation d'ordre, mais posent problème pour la prise en compte des contraintes temporelles. Ainsi, dans le modèle mis en place, le temps est discrétisé et chaque pas de temps correspond à un changement dans les fenêtres de visibilité (début ou fin d'une fenêtre) ; le modèle ne prend pas en compte la contrainte liée à la période de double visibilité pour la réalisation des handovers. Pour des problèmes réduits à un petit nombre de stations, cette méthode permet de trouver, s'il en existe, une solution réalisable mais elle ne permet pas de chercher des solutions optimisées. Une approche basée uniquement sur les Réseaux de Petri ne s'avère donc pas satisfaisante puisque les contraintes de précédence exprimées par la structure du réseau de Petri ne sont pas prépondérantes et que les autres contraintes, en particulier celles liées aux fenêtres de visibilité, doivent être stockées sous la forme brute d'attributs associés aux jetons. Or le nombre de ces contraintes augmente exponentiellement avec la taille du problème.

Suite à ces travaux, nous avons cherché à associer différentes méthodes pour modéliser et résoudre le problème dans sa globalité. En particulier, nous avons pensé à introduire des techniques de propagation de contraintes dans le modèle par Réseaux de Petri afin de réduire les conflits de ressources à régler.

Dans cette optique, nous avons proposé un modèle mathématique pour formaliser les variables et les contraintes, en particulier les contraintes temporelles, du problème d'allocation des liens stations/satellites.

3.4 Modélisation du problème

Nous proposons une formalisation du problème d'allocation de liens sous forme d'un programme linéaire en variables mixtes.

Nous introduisons d'abord les indices et paramètres suivants :

– T : horizon temporel de planification,

- i : numéro de position dans une séquence d'allocations pour une station, $i = 1, \ldots, nsup$ où $nsup$ est une borne supérieure du nombre d'allocations de liens pour une station sur l'horizon T ; on définit également une borne inférieure $ninf$ du nombre d'allocations de liens pour une station sur l'horizon T,
- j : numéro de satellite, $j = 1,2,\ldots,s$ où s est le nombre de satellites de la constellation,
- k : numéro de station sol $k = 1,2,\ldots,g$ où g est le nombre de stations sol considérées,
- m : numéro de fenêtre de visibilité entre une station sol et un satellite sur l'horizon T, $m = 1,2,\ldots,w$ où w est une borne supérieure du nombre de fenêtres de visibilité qu'il peut y avoir entre une station sol et un satellite sur l'horizon T,
- $Sv_{j,m,k}$: date de début de la $m^{\text{ème}}$ fenêtre de visibilité entre la station k et le satellite j,
- $Fv_{j,m,k}$: date de fin de la $m^{\text{ème}}$ fenêtre de visibilité entre la station k et le satellite j.

Nous posons alors les variables :

- $X_{i,j,m,k}$: variable binaire égale à 1 si la $m^{\text{ème}}$ fenêtre de visibilité du satellite j pour la station k est affectée en $i^{\text{ème}}$ position à k, à 0 sinon.
- $S_{i,k}$: variable réelle égale à la date de début de la $i^{\text{ème}}$ allocation de la station k.
- y : borne supérieure du nombre total d'affectations d'un satellite sur l'horizon T.
- z : borne supérieure du nombre d'allocations d'un satellite à un même rang, sur l'horizon T.

Nous reviendrons par la suite sur les deux dernières variables.

Les contraintes du problème s'expriment par les équations suivantes :

$$\sum_m \sum_j X_{i,j,m,k}.Sv_{j,m,k} \leq S_{i,k} \qquad i{=}1..nsup,k{=}1..g \qquad (1)$$

$$\sum_m \sum_j X_{i,j,m,k}.(Fv_{j,m,k} - (2\Delta H + \Delta T)) \geq S_{i,k} \qquad i{=}1..nsup,k{=}1..g \qquad (2)$$

$$\sum_m \sum_j X_{i,j,m,k}.(Fv_{j,m,k} - \Delta H) \geq S_{i+1,k} \qquad i{=}1..nsup{-}1,k{=}1..g \qquad (3)$$

$$X_{i+1,j,m,k}.(\Delta H + \Delta T) \leq S_{i+1,k} - S_{i,k} + (1 - X_{i+1,j,m,k}).\ M \qquad i{=}1..nsup{-}1,j{=}1..s,m{=}1..w,k{=}1..g \qquad (4)$$

$$\sum_m \sum_j X_{i,j,m,k} = 1 \qquad i{=}1..ninf{-}1,k{=}1..g \qquad (5)$$

$$S_{1,k} = 0 \qquad k{=}1..g \qquad (6)$$

$$\sum_m \sum_j X_{i,j,m,k} \geq \sum_m \sum_j X_{i+1,j,m,k} \qquad i{=}ninf..nsup,k{=}1..g \qquad (7)$$

$$\sum_i \sum_{m|Fv_{j,m,k} \geq T} \sum_j X_{i,j,m,k} = 1 \qquad k{=}1..g \qquad (8)$$

$$\sum_i \sum_m \sum_k X_{i,j,m,k} \leq y \qquad j{=}1..s \qquad (9)$$

$$\sum_m \sum_k X_{i,j,m,k} \leq z \qquad j{=}1..s,i{=}1..nsup \qquad (10)$$

Les contraintes (1) et (2) assurent que chaque allocation de satellite est réalisée dans une fenêtre de visibilité ; chaque $S_{i,k}$ doit être choisie dans la fenêtre correspondante : $[Sv_{j,m,k}; Fv_{j,m,k} - (2\Delta H + \Delta T)]$. $S_{i,k}$ ne peut pas être supérieure à $Fv_{j,m,k} - (2\Delta H + \Delta T)$ puisque $(2\Delta H + \Delta T)$ est la durée minimale de chaque lien d'une station avec un satellite (une durée ΔH pour le handover qui permet d'allouer le satellite, plus une durée minimale de

communication ΔT, plus une nouvelle durée ΔH pour le handover qui permet de relâcher le satellite).

La contrainte de double visibilité pour la réalisation des handovers est exprimée en (3). Pour chaque station, elle force chaque allocation à débuter au moins ΔH avant la fin de la précédente.

La contrainte de durée minimale ΔT d'une communication est établie dans (4) en imposant une durée minimale $\Delta H + \Delta T$ entre deux dates de début d'allocations successives. On utilise dans cette formule un paramètre M qui est un réel positif suffisamment grand pour rendre l'inéquation triviale si $X_{i+1,j,m,k} = 0$.

Les équations (5) et (6) permettent l'initialisation des solutions : chaque station doit avoir au moins $ninf$ allocations de liens et la première commence à la date 0.

Afin de conserver, pour chaque séquence d'allocations, des indices de position cohérents, la contrainte (7) exprime le fait que, pour chaque station k, s'il existe une $i^{\text{ème}}$ allocation (*i.e.,* il existe j,m tels que $X_{i,j,m,k} = 1$), alors il en existe une $(i-1)^{\text{ème}}$ (*i.e.,* il existe j',m' tels que $X_{i-1,j',m',k} = 1$).

La contrainte (8) garantit qu'un lien est alloué à chaque station à la date T ; la continuité de liaison de chaque station avec la constellation est ainsi assurée sur tout l'horizon considéré.

Les contraintes (9) et (10) nous permettent de prendre en compte dans une certaine mesure le caractère limité des ressoures satellites. On a vu précédemment que le nombre de liens que peut établir simultanément un satellite est limité par son nombre d'antennes ; mais ce nombre n'est pas bien défini et on ne focalise pas notre étude sur ce point. De plus, on sait que le partage des ressources est difficile à modéliser linéairement sans une discrétisation du temps. Ainsi, dans un premier temps, on ne prend pas en compte les contraintes de ressources liées aux satellites. Les contraintes (9) et (10) ne sont alors pas présentes dans le modèle et on introduit le critère d'optimisation de la façon suivante.

Les solutions recherchées doivent satisfaire les précédentes contraintes et minimiser le nombre de handovers réalisés sur l'horizon T, ce qui revient à minimiser le nombre d'allocation de liens sur cet horizon. La fonction objectif à minimiser s'écrit alors sous la forme :

$$Min\ (\textstyle\sum_{i,j,m,k} X_{i,j,m,k}) \tag{11}$$

Ensuite, afin de tendre à minimiser également le nombre d'antennes d'un même satellite utilisées simultanément, on introduit les contraintes (9) et (10) qui définissent les variables y et z comme étant, respectivement, le nombre maximal d'antennes utilisées par satellite sur tout l'horizon T et le nombre maximal d'antennes utilisées par satellite et par numéro de fenêtre de visibilité avec les différentes stations sol.

L'expression (11) est ensuite augmentée des variables y et z et la fonction objectif devient :

$$Min\ (\textstyle\sum_{i,j,m,k} X_{i,j,m,k}\ +\ y\ +\ z) \tag{12}$$

Nous obtenons ainsi un programme linéaire en variables mixtes qui modélise toutes les contraintes temporelles de notre problème, dans lequel le temps n'est pas discrétisé.

3.5 Travaux connexes

Deux travaux traitant de problématiques très proches de la nôtre ont été publiés.

Un problème de dimensionnement d'une constellation de type Skybridge est étudié dans la thèse de [Lebbar 00]. L'auteur propose un modèle linéaire en nombres entiers basé sur une discrétisation temporelle et une résolution par des techniques de génération de colonnes et de contraintes. Cette approche permet d'obtenir des bornes inférieures de bonne qualité et, associée à une décomposition temporelle, permet de générer des solutions entières sur des horizons temporels limités.

Plus récemment, un travail de thèse portant sur l'allocation de ressources dans la constellation Skybridge a été publié [Landes 02]. Le problème étudié prend en compte de nombreuses contraintes techniques du problème réel. L'auteur modélise le problème d'allocation de liens de communication comme un problème de recherche de chemins dans un graphe d'intervalles. Des algorithmes tabou et de recherche par voisinage variable permettent d'obtenir des solutions sur un horizon temporel correspondant à un sous-cycle (environ 10 heures).

D'autres travaux concernant l'étude de systèmes satellitaires de télécommunication ont été publiés, mais ils portent essentiellement sur des systèmes géostationnaires (comme **Eutelsat**, [Prins 91]), ou des constellations ayant la capacité d'établir des liens inter-satellites (telles que **Iridium** ou **Teledesic** [Pratt *et al.* 99, Chang *et al.* 95, Ferreira *et al.* 00, Ferreira *et al.* 01b]). Dans ces deux cas, les problèmes ainsi que les contraintes du système sont assez différents de celui qui nous concerne. En particulier, la contrainte de chevauchement liée à la réalisation des handovers est très spécifique aux problèmes posés par les constellations basse altitude et elle est peu traitée, en temps que contrainte forte, dans la littérature.

3.6 Bilan de l'étude sur Skybridge

Le problème d'allocation des liens de communication dans la constellation Skybridge tire sa particularité et sa grande complexité des handovers qu'il faut réaliser régulièrement pour assurer un lien continu entre deux utilisateurs.

Considérant des hypothèses simplificatrices concernant essentiellement la demande de service (agrégation des utilisateurs, demande constante et unitaire pour chaque utilisateur), nous avons modélisé ce problème sous forme d'un programme linéaire en variables mixtes dans lequel le temps n'est pas discrétisé et qui prend en compte toutes les contraintes liées au processus de réalisation des handovers.

Le projet Skybridge ayant été abandonné au cours de l'automne 2000 et suite aux grandes difficultés rencontrées par la plupart des projets de ce type (allant jusqu'au "dé-orbitage" de la constellation Iridium), notre étude concernant le problème d'allocation de liens de communication dans les constellations basse altitude s'est rapidement arrêtée. Nous évoquons ci-après les résultats de validation obtenus ainsi que les pistes de travail envisagées, mais ces travaux n'ont pas pu être poursuivis et nous n'y reviendrons plus dans la suite de ce mémoire.

Nous avons validé et testé notre modèle en utilisant le logiciel de programmation linéaire Xpress-MP. Les résultats obtenus nous permettent de trouver une solution optimale pour des problèmes réduits à quelques utilisateurs et sur un horizon de calcul d'une heure environ [Mancel 00]. Cependant, l'utilisation de la programmation linéaire est rapidement limitée par la taille du problème à considérer et, s'il semble possible d'augmenter l'horizon de calcul en

effectuant une décomposition temporelle – comme proposé dans [Lebbar 00] –, il est clair que la programmation linéaire seule ne permet pas de résoudre le problème pour des zones géographiques plus étendues.

Partant du constat qu'aucune méthode étudiée jusqu'alors au CNES n'était totalement satisfaisante, mais que chacune présentait des aspects intéressants, l'idée était de combiner différentes techniques pour aborder ce problème. En particulier, une coopération des réseaux de Petri et des récents résultats obtenus concernant leur analyse par la logique linéaire [Pradin-Chézalviel *et al.* 99], avec des outils de programmation linéaire était envisagée [Mancel *et al.* 01b].

Toutefois, comme nous l'avons déjà mentionné, la structure du réseau de Petri n'exprime qu'une faible partie des contraintes. En fait, seule la contrainte d'affectation, modélisée par l'équation (8), est prise en compte.

S'appuyer sur cette seule contrainte pour essayer de limiter l'espace de recherche des solutions nous semble peu efficace comme nous l'avions déjà dit dans le paragraphe 1.3.2.4, chapitre 1. Si le projet Skybridge avait été pousuivi, nous nous serions plutôt orienté vers des approches par décomposition et|ou génération de colonnes comme nous l'avons fait dans le cadre des projets Netlander et Pléiades que nous présentons dans les chapitres suivants.

Chapitre 4

Netlander : projet d'exploration de la planète MARS

4.1 Présentation de la mission

L'étude de la planète Mars permet de suivre l'essentiel des processus ayant affecté, sur des échelles de temps géologiques ou saisonnières, les planètes internes du système solaire et en particulier la Terre. En effet, Mars a connu toutes les étapes de l'évolution planétaire, depuis le bombardement intense du premier milliard d'années du système solaire jusqu'à la mort géologique, en passant par une phase de forte activité et elle présente aujourd'hui encore des types de surface acquis tout au long de son histoire. L'étude de Mars intéresse ainsi une large communauté scientifique incluant géologues, géophysiciens, climatologues, ... mais aussi hydrologues et biologistes car c'est également la seule planète avec la Terre qui ait pu présenter à un moment de son histoire les conditions propres à l'émergence et au développement de la vie.

Les premières missions d'exploration de Mars remontent au milieu des années 60 avec en particulier les premières photographies de la planète réalisées par les sondes américaines **Mariner**. Ensuite, les missions **Viking 1** et **2**, qui ont cartographié Mars en 1976 et y ont déposé 2 atterrisseurs pour réaliser les premières mesures *in situ*, ont apporté l'essentiel des connaissances sur Mars jusqu'au milieu des années 90.

Suite à des revers financiers et plusieurs échecs cuisants au cours des années 90, l'exploration de Mars s'oriente vers des missions plus légères, moins coûteuses et à participation internationale. On retient en particulier le succès de la mission **Pathfinder** qui en 1997 fait atterrir sur Mars, de façon innovante, un petit robot mobile (rover), protégé par des coussins d'air au lieu d'être freiné par des rétrofusées.

Actuellement, trois sondes gravitent autour de Mars : **Mars Global Surveyor**, **Mars Odyssey**, qui avait permis de supposer la présence d'eau dans le sous-sol martien et, très récemment, la sonde européenne **Mars Express** qui a prouvé la véracité de cette hypothèse, par observation directe de glace d'eau sur la calotte polaire australe de Mars.

Les prochaines étapes de l'exploration martienne vont consister en deux principaux points :

– d'une part, la réalisation d'expériences et mesures *in situ*, avec, par exemple, les rovers américains **Spirit** et **Opportunity**, lancés dans la fenêtre de tir de juin–juillet 2003 (la

mission de l'atterrisseur britannique **Beagle 2**, lancé avec la sonde Mars Express dans cette même fenêtre de tir, mais probablement détruit lors de son atterrissage sur Mars, s'inscrivait également dans cet axe d'étude) ;
– d'autre part, le retour sur Terre d'échantillons martiens en vue d'analyses fines de leur structure et de leur composition avec le programme **Mars Sample Return** prévu pour 2014.

Dans ce cadre, le CNES a entrepris la mise en œuvre d'un programme d'exploration de Mars, appelé **PREMIER** (Programme de Retour d'Échantillons Martiens et Installation d'Expériences en Réseau) dont les deux principaux volets sont :

– une participation au programme **Mars Sample Return** (mené par la NASA) ; ce volet inclut le développement et la gestion des opérations de l'orbiteur martien de la mission ;
– le développement du projet **Netlander** qui consiste à déployer à la surface de Mars quatre atterrisseurs (ou sondes) fonctionnant en réseau et destinés à étudier la géophysique et la météorologie martienne. Les États-Unis ainsi que plusieurs partenaires européens participent à ce projet, en particulier la Finlande, l'Allemagne, la Belgique et la Suisse.

Le CNES assure la maîtrise d'œuvre du projet Netlander et a la responsabilité du développement du système d'entrée atmosphérique, de descente et d'atterrissage des sondes. La charge utile scientifique est développée par plusieurs instituts européens et américains.

Selon le scénario nominal, le projet Netlander doit se dérouler de la façon suivante. Un orbiteur martien (prototype de l'orbiteur de la mission Mars Sample Return) et quatre sondes, appelées *netlanders*, doivent être lancés en Août 2007 par une fusée Ariane 5 pour rejoindre la planète Mars. Après une phase de croisière d'une dizaine de mois, l'orbiteur se met en place autour de Mars et les sondes sont éjectées de l'orbiteur pour effectuer un atterrissage comparable à celui de la mission Pathfinder, avec une répartition de trois sondes dans l'hémisphère nord et de la quatrième dans l'hémisphère sud. Les stations, une fois immobilisées sur Mars déploient leurs panneaux solaires et leur antenne de télécommunication, puis activent leurs différents instruments de mesures. Ces étapes successives sont illustrées dans la figure 4.1.

La phase dite de *routine* de la mission débute : les sondes réalisent différentes expériences – indépendamment les unes des autres ou bien en réseau – et elles envoient régulièrement sur Terre les résultats accumulés ; toute communication entre les sondes et la Terre utilise un orbiteur martien comme relais (cf. figure 4.2), qui peut être l'orbiteur prototype de Mars Sample Return ou éventuellement un des autres orbiteurs martiens envoyés lors de précédentes missions.

La durée de fonctionnement des sondes est estimée à une année martienne, soit environ deux années terrestres. Elle dépendra essentiellement de la vitesse de dégradation des panneaux solaires qui permettent l'alimentation électrique des sondes.

Différents types d'expériences sont prévus pour permettre l'étude géophysique et météorologique de Mars. Les expériences sont déclenchées soit à une date connue soit sur événement.

4.2 Problématique

La taille de la mémoire de masse (MdM) disponible au niveau de chaque sonde, l'énergie électrique – produite par les panneaux solaires et consommée par les instruments et par

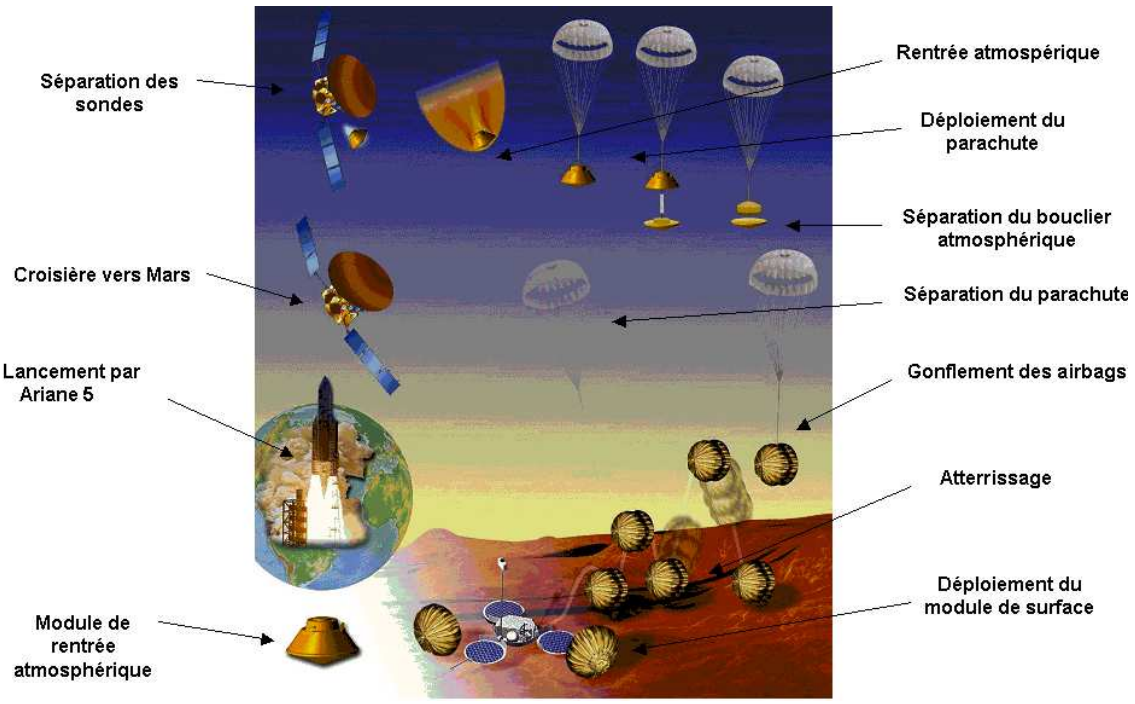


FIG. 4.1 – *Déroulement de la mise à poste de la mission Netlander*

l'antenne de communication –, ainsi que les opportunités et le débit de communication entre chaque sonde et l'orbiteur puis entre l'orbiteur et la Terre, ne permettent pas d'envisager une utilisation continue et simultanée de l'ensemble des instruments qui équipent les sondes. Des choix sont nécessaires et il faut organiser dans le temps les diverses opérations des sondes et de l'orbiteur.

Le schéma retenu consiste à effectuer cette planification de façon régulière au sol (au centre terrestre des opérations – NMOC pour Netlander Mission Operation Center), en concertation avec les scientifiques impliqués dans la mission, puis à télécharger les plans de travail ainsi obtenus à l'orbiteur puis aux sondes. Ce choix est justifié par différents points. Tout d'abord, les valeurs nominales, concernant les consommations en mémoire et en énergie des instruments et de l'antenne des sondes, les débits de communication sondes/orbiteur et orbiteur/Terre, ainsi que la production énergétique des panneaux solaires des sondes, sont *a priori* connues. Les fenêtres de visibilité entre chaque sonde et l'orbiteur et entre l'orbiteur et la Terre sont également prévisibles une fois l'orbite déterminée. De plus, des informations sur l'évolution de la mémoire de masse et de l'énergie électrique disponibles peuvent être obtenues au sol au cours de la mission. Enfin, les scientifiques veulent rester impliqués dans la boucle de décision tout au long de la mission et si l'on peut envisager une automatisation de l'organisation des expériences à réaliser à court terme (quelques jours), il n'en va pas de même sur les moyen et long termes.

Un **plan de travail** définit d'une part, la mise en œuvre des instruments, d'autre part, le type de données attendues en résultat. La mise en œuvre des instruments sur chaque sonde est planifiée pour une période donnée (de l'ordre de la semaine). L'allocation des zones de mémoire de masse des sondes aux résultats des différentes expériences est programmée dans le plan de travail ainsi que les visibilités orbiteur/sonde qui seront utilisées pour rapatrier les

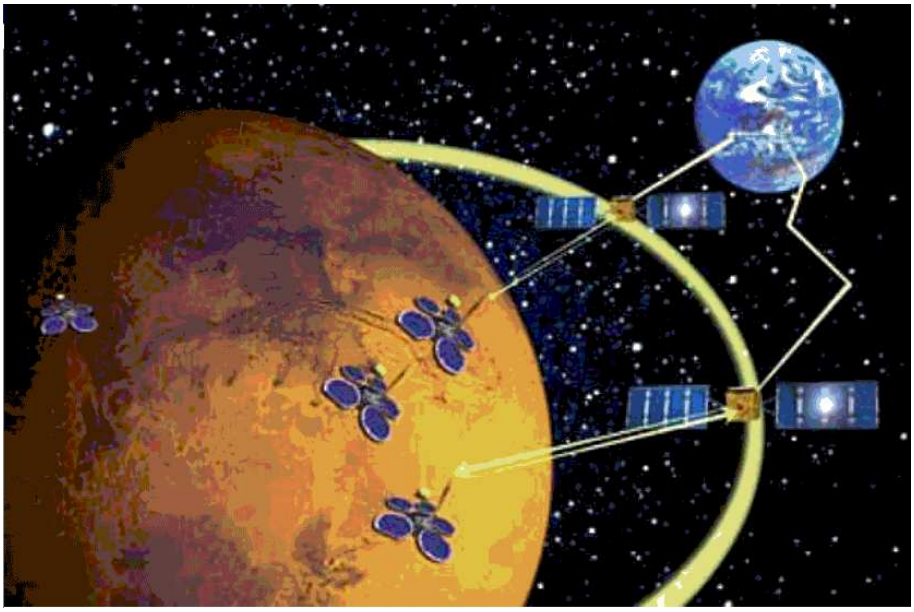


FIG. 4.2 – *Mode de communication entre les sondes et la Terre*

données.

Il existe un décalage temporel d'au moins deux jours entre l'envoi d'un plan de travail et le retour à Terre des premiers résultats correspondants. Ce décalage est dû en particulier aux durées des expériences, aux durées de transfert des informations et à la nécessité de respecter les fenêtres de visibilité pour effectuer les communications.

On appelle **boucle de programmation** l'enchaînement d'actions qui commence par l'élaboration d'un plan de travail et qui se termine par le retour au sol des données résultant de ce plan. Les différentes actions qui composent une boucle de programmation interviennent donc aux différents niveaux : sol, orbiteur et sondes. La figure 4.3 schématise les principales étapes d'une boucle de programmation.

Le problème posé est celui de l'optimisation de ces boucles de programmation. Un premier travail a consisté à caractériser précisément les composantes d'une boucle de programmation ainsi que les critères d'optimisation souhaités et à identifier et formaliser les problèmes de décision qui se posent.

Il en ressort qu'"optimiser une boucle de programmation" consiste à effectuer les choix qui permettent à la fois de minimiser la durée de cette boucle (afin d'obtenir au plus vite les résultats d'expériences) et de maximiser la valeur, en termes de quantité, de qualité et de pertinence scientifique, des résultats d'expériences obtenus.

Les décisions intervenant au cours d'une boucle de programmation concernent, d'une part, les communications sondes/orbiteur et orbiteur/Terre (sélection des fenêtres de visibilité à utiliser et ordonnancement des créneaux de communication) et, d'autre part, les expériences à réaliser (sélection d'un ensemble d'expériences cohérent et ordonnancement de cet ensemble).

Il apparaît ensuite que les fenêtres de visibilité orbiteur/Terre ainsi que les débits disponibles pour les communications orbiteur/Terre ne sont pas limitants. On considère ainsi qu'on dispose d'un créneau de communication orbiteur/Terre journalier suffisant pour envoyer un

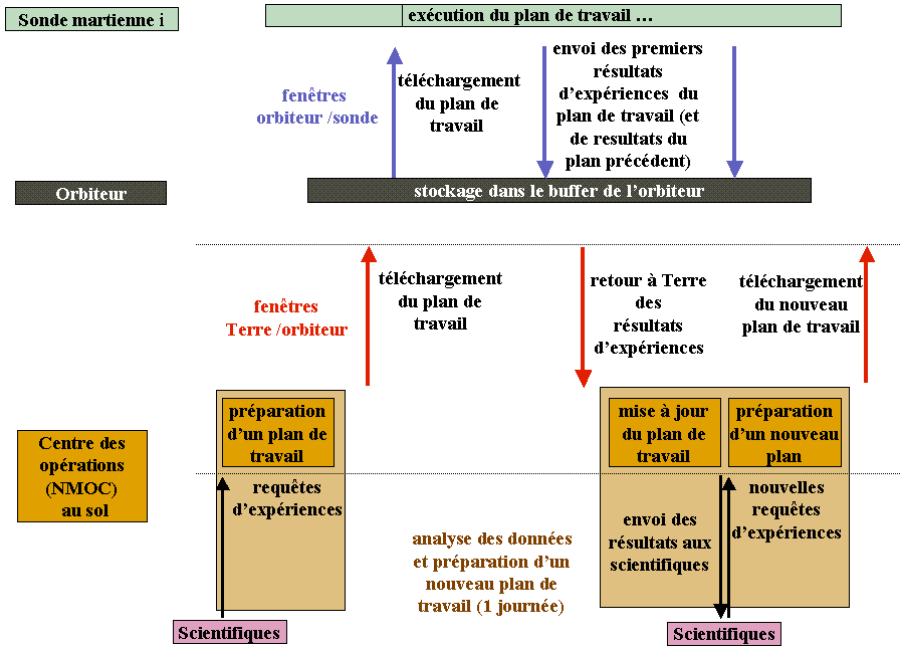


FIG. 4.3 – *Boucle de programmation*

plan de travail à l'orbiteur et télécharger au sol toutes les données stockées dans le buffer de l'orbiteur.

Ainsi, le problème de décision qui nous intéresse est double : il s'agit en effet de planifier à la fois les communications orbiteur/sondes et les tâches qui constituent les plans de travail envoyés aux sondes lors de ces communications. Les solutions recherchées doivent vérifier des contraintes fortes, liées essentiellement au partage de la mémoire de masse, de l'énergie électrique et du temps.

4.3 Analyse du problème

4.3.1 Revue de la littérature

Les travaux publiés sur le projet Netlander sont peu nombreux et concernent essentiellement les aspects technologiques de la mission. Il n'existe aucune publication traitant du problème de planification des communications et des expériences de la mission Netlander ; en effet, c'est dans le cadre de notre étude sur le projet Netlander que ce problème a été défini.

D'autres missions d'exploration récentes ou en cours d'étude nécessitent la planification, au cours de la mission, de certaines tâches. Les problèmes posés sont d'autant plus difficiles à résoudre que la diversité des expériences envisagées est grande et que l'environnement du déroulement de la mission est incertain ou mal connu.

Ainsi, plusieurs travaux menés à ce sujet, en particulier par la NASA, font appel à des heuristiques simples pour générer des plans initiaux, ou même considèrent que ces plans sont fournis par des experts et se focalisent sur l'exécution du plan en prévoyant une réaction aux aléas. Le but est alors de permettre à l'engin spatial de modifier son plan de façon à s'adapter

aux changements qui interviennent dans l'environnement de sa mission et ainsi d'éviter de se trouver dans une situation de blocage qui pourrait mettre fin à l'ensemble de la mission [Estlin & Gaines 02, Jónsson *et al.* 00].

Cependant, dans le cadre d'une mission où les expériences à réaliser sont diverses et complexes, trouver un plan réalisable, même de qualité moyenne, est souvent déjà un problème difficile. Dans [Hall & Magazine 94] les auteurs abordent le problème général de l'optimisation d'une mission spatiale en proposant plusieurs heuristiques de planification des expériences ainsi qu'un calcul de borne supérieure permettant d'évaluer la qualité des heuristiques.

Notons enfin qu'un problème de planification issu de la mission Mars Express, proche de celui posé par Netlander a récemment été traité dans [Cesta *et al.* 02a]. Dans cette mission, l'orbiteur martien est équipé d'instruments de mesure qui exécutent différentes expériences. Le problème consiste à planifier, en fonction d'un plan d'expériences supposé connu, les transferts de données de l'orbiteur vers la Terre, dans des délais permettant d'éviter l'écrasement de données sur le buffer de l'orbiteur. Les auteurs définissent le problème MEX-MDP pour Mars EXpress Memory Dumping Problem et présentent un outil logiciel d'aide à la décision. Différentes fonctions d'interaction sont proposées à l'utilisateur, avec, en particulier, la possibilité d'améliorer une solution initiale (générée automatiquement par un algorithme glouton) "à la main" ou en paramétrant un algorithme de recherche locale.

4.3.2 Proposition d'une décomposition

Les décisions concernant les communications orbiteur/sondes et les expériences à réaliser sont de nature différente.

En effet, il s'avère tout d'abord qu'elles ne sont pas soumises aux mêmes contraintes. Une communication orbiteur/sonde nécessite que les antennes des deux engins soient en visibilité et elle requiert une assez faible puissance électrique. L'exécution d'une expérience sur une sonde consomme également de l'énergie électrique avec une puissance variable selon le type d'expérience ; elle requiert de plus un emplacement alloué sur la MdM pour stocker les résultats. Enfin, suivant leur type (expériences réalisées en réseau, déclenchées sur événement ou sur date, etc.), les expériences sont soumises à différentes contraintes temporelles.

Par ailleurs, les planifications des communications et des expériences interviennent à des étapes différentes de la vie du projet. Les communications orbiteur/sondes peuvent être planifiées avant le début de la mission avec des réajustements au cours de son déroulement (sur des horizons d'environ 1 mois) en fonction de l'évolution de l'énergie électrique disponible sur les sondes et sur l'orbiteur, tandis que la planification des expériences se fera au cours de la mission, sur des horizons d'une semaine environ, avec des réajustements éventuels quotidiens permettant ainsi d'être réactif à l'évolution de l'environnement martien.

Enfin, on peut également considérer que les communications orbiteur/sondes et les expériences n'ont pas le même niveau de priorité dans le déroulement de la mission. En effet, si les expériences réalisées au cours de la mission définiront l'intérêt scientifique que l'on pourra en retirer, les communications sont quant à elles absolument nécessaires au fonctionnement même de la mission. En effet, les communications orbiteur/sondes assurent le seul lien entre la Terre et les sondes martiennes ; elles servent à envoyer les plans de travail aux différentes sondes et à rapatrier au sol les résultats des expériences ainsi que différentes données de servitude. L'exécution d'une expérience n'est envisageable que dans la mesure où l'on peut envoyer

l'ordre correspondant aux sondes et en récupérer les résultats. Ainsi, les communications orbiteur/sondes apparaissent prioritaires par rapport aux expériences.

Il nous semble alors naturel de décomposer le problème global de planification des communications et des expériences de la mission Netlander en deux sous-problèmes : nous considérons d'une part, le problème de planification des communications orbiteur/sondes et d'autre part, celui de la planification des expériences. Le tableau 4.1 synthétise les principales caractéristiques des deux sous-problèmes.

		planification des communications orbiteur/sondes	planification des expériences
	critère d'optimisation	minimisation de la durée d'une boucle de programmation	maximisation de la "valeur" d'une boucle de programmation
	horizon de planification	1 mois	1 semaine
ressources requises	visibilités orbiteur/sondes	oui	non
	buffer (orbiteur)	oui	non
	puis. électrique (sondes)	oui	oui
	MdM (sondes)	non	oui
connaissances requises	visibilités orbiteur/sondes	déterminées en début de projet	–
	buffer (orbiteur)	évaluation en début de projet + mises-à-jour en cours de mission	–
	puis. électrique (sondes)	évaluation en début de projet + mises-à-jour en cours de mission	évaluation fine en cours de mission
	MdM (sondes)	–	évaluation fine en cours de mission

TAB. 4.1 – *Caractéristiques des deux sous-problèmes*

Notons que ces deux problèmes ne sont pas totalement indépendants ; nous avons vu en particulier qu'ils sont liés par le partage de la ressource énergétique des sondes martiennes, utilisée à la fois pour l'exécution des expériences et pour les communications. Cependant, la priorité étant donnée à la réalisation de communications orbiteur/sondes plutôt que d'expériences, nous allons considérer d'abord le sous-problème de planification des communications, puis celui de planification des expériences en supposant alors connue l'énergie électrique consommée par les communications.

4.3.3 Sous-problème de planification des communications orbiteur/sondes

4.3.3.1 Analyse du problème

On dispose d'une visibilité Terre/orbiteur de huit heures par jour. On suppose défini un créneau quotidien de communication Terre/orbiteur suffisant pour *monter* un plan de travail aux sondes et pour *descendre* au sol toutes les données stockées dans le buffer de l'orbiteur. Les fenêtres de visibilité entre l'orbiteur et les sondes sont, quelle que soit l'orbite envisagée, beaucoup plus fréquentes et beaucoup plus courtes ; on considère qu'on dispose, par sonde et entre deux créneaux de commmunication Terre/orbiteur, d'au moins une fenêtre de visibilité avec l'orbiteur permettant de monter le plan de travail jusqu'à la sonde.

Le retour des données au sol est ainsi quotidien et il n'est donc pas rythmé directement par les dates de communication orbiteur/sonde. En revanche, le choix de ces créneaux de communication influe sur la quantité de données qui pourra être descendue au sol chaque jour. Ainsi, dans le but de minimiser la durée de retour au sol des données, on doit planifier les communications orbiteur/sondes de façon à maximiser le volume de données échangées.

Différentes contraintes portent sur l'exécution de ces communications : d'une part, des contraintes temporelles :

- chaque communication ne peut s'effectuer que dans une fenêtre de visibilité,
- chaque communication doit durer un temps minimum (de l'ordre de 6 minutes),
- un temps de reconfiguration doit être observé après chaque communication (estimé à 2 minutes),

d'autre part, des contraintes de partage de ressources :

- l'énergie électrique disponible au niveau des sondes et de l'orbiteur est limitée et se renouvelle par les panneaux solaires au cours du temps ; une communication ne peut être réalisée que dans la mesure où la puissance requise est disponible,
- la capacité du buffer de l'orbiteur est également limitée et une communication n'est utile qu'à la condition que les données qui lui sont envoyées par la sonde puissent être stockées (notons que la MdM des sondes est aussi limitée mais le volume de données qui leur sont envoyées n'est pas critique car il correspond uniquement aux plans de travail),
- enfin, l'orbiteur peut communiquer avec une seule sonde à la fois. Cette contrainte se pose lorsque l'orbiteur survole les trois sondes situées dans le même hémisphère ; en effet, du fait de leur proximité, leurs fenêtres de visibilité se chevauchent.

Une première analyse du sous-problème de planification des communications orbiteur/sondes, faite en collaboration avec les experts du projet Netlander, nous permet de considérer que l'énergie électrique disponible pour les communications, au niveau de l'orbiteur et des sondes, ainsi que la capacité du buffer de l'orbiteur, sont des facteurs très peu limitants devant celui des fenêtres de visibilité orbiteur/sondes. De plus, la puissance électrique requise pour une communication ainsi que le débit d'échange des données sont des valeurs supposées constantes. L'énergie consommée par une communication ainsi que le volume de données échangées sont donc linéairement dépendants de la durée de cette communication. On peut donc simplifier les contraintes de partage de la ressource électrique et du buffer de l'orbiteur en les traduisant en termes de durée maximale de communication sur certains intervalles temporels. D'autre part, maximiser le volume de données échangées entre deux communications quotidiennes entre la Terre et l'orbiteur est équivalent à maximiser la durée des communications orbiteur/sondes.

4.3.3.2 Modélisation

Le problème de planification des communications orbiteur/sondes peut se formuler de la façon suivante : connaissant les fenêtres de visibilité entre les sondes et l'orbiteur, il s'agit de déterminer, sur un horizon T donné, les créneaux de communication de chaque sonde, de façon à maximiser la durée totale de communication.

Nous présentons un modèle de ce problème, basé sur une discrétisation temporelle, sous forme du programme linéaire en nombres entiers détaillé dans la suite.

Nous introduisons les indices et paramètres suivants :

- T : horizon temporel de planification,
- t : pas de temps de l'horizon T,
- $\mathcal{J} = (j_1,j_2,...,j_J)$: liste des pas de temps correspondants à la fin d'une communication Terre/orbiteur (l'intervalle entre deux éléments consécutifs de la liste correspond à une journée),
- s : numéro de sonde, $s = 1$ à N ($N = 4$, le nombre de sondes),
- I_s : union des intervalles d'indisponibilité sur T de l'orbiteur pour la sonde s (il s'agit de l'ensemble complémentaire sur T des fenêtres de visibilité entre la sonde s et l'orbiteur),
- $dmin$: durée minimale d'une communication,
- dr : durée de reconfiguration de l'orbiteur entre deux communications successives,
- $Dmax1_{orb}$: durée maximale de communication de l'orbiteur entre deux communications Terre/orbiteur (traduit la capacité du buffer de l'orbiteur),
- $\mathcal{K}_s = (k_1,k_2,...,k_{K_s})$: liste des pas de temps définissant les intervalles sur lesquels sont évalués l'énergie électrique disponible pour la sonde s,
- $Dmax_s(k_i,k_{i+1})$: durée maximale de communication de la sonde s sur l'intervalle $[k_i;k_{i+1}]$, $i = 1,...,K_s$ (traduit l'énergie électrique disponible sur l'intervalle $[k_i;k_{i+1}]$ sur s),
- $\mathcal{L} = (l_1,l_2,...,l_L)$: liste des pas de temps définissant les intervalles sur lesquels sont évalués l'énergie électrique disponible pour l'orbiteur,
- $Dmax2_{orb}(l_i,l_{i+1})$: durée maximale de communication de l'orbiteur sur l'intervalle $[l_i;l_{i+1}]$, $i = 1,...,L$ (traduit l'énergie électrique disponible sur l'intervalle $[l_i;l_{i+1}]$ sur l'orbiteur).

Nous posons la variable binaire $X_{s,t}$, égale à 1 si, au pas de temps t, la sonde s communique avec l'orbiteur, égale à 0 sinon.

Les contraintes du problème s'expriment par les inéquations :

$$X_{s,t} = 0 \qquad s{=}1..N,\ t{\in}I_s \qquad (13)$$

$$\sum_s X_{s,t} \leq 1 \qquad t{=}1..T \qquad (14)$$

$$\sum_{t'=t+1}^{t+dmin} X_{s,t'} \geq dmin.(X_{s,t+1} - X_{s,t}) \qquad t{=}1..(T{-}dmin),\ s{=}1..N \qquad (15)$$

$$X_{s,t} \leq X_{s,t-1} \qquad t{=}(T{-}dmin{+}1)..T,\ s{=}1..N \qquad (16)$$

$$\sum_{t'=t+1}^{t+dr-1} X_{s,t'} \leq dr.(1 - X_{s,t-1} + X_{s,t}) \qquad t{=}2..(T{-}dr{+}1),\ s{=}1..N \qquad (17)$$

$$\sum_{t=j_i}^{j_{i+1}} \sum_s X_{s,t} \leq Dmax1_{orb} \qquad i{=}1..J{-}1 \qquad (18)$$

$$\sum_{t=l_i}^{l_{i+1}} \sum_s X_{s,t} \leq Dmax2_{orb}(l_i,l_{i+1}) \qquad i{=}1..L{-}1 \qquad (19)$$

$$\sum_{t=k_i}^{k_{i+1}} X_{s,t} \leq Dmax_s(k_i,k_{i+1}) \qquad i{=}1..K_{s-1},\ s{=}1..N \qquad (20)$$

Les contraintes (13) et (14) expriment le fait que l'orbiteur est une ressource disjonctive, non continûment disponible : la contrainte (13) impose que la sonde s ne communique pas avec l'orbiteur sur son ensemble I_s ($X_{s,t} = 0$ sur cet ensemble). L'expression (14) permet de vérifier qu'à chaque pas de temps l'orbiteur communique avec au plus une sonde.

Les contraintes (15) et (16) imposent une durée minimale $dmin$ aux communications. En particulier, la contrainte (15) exprime le fait que si, à un pas de temps t_0, une sonde s débute une communication avec le satellite ($X_{s,t_0-1} = 0$ et $X_{s,t_0} = 1$), alors $X_{s,t}$ vaut 1 pour t de t_0 à $t_0 + dmin - 1$ (la communication de s qui débute en t_0 dure au moins $dmin$ pas de temps). La contrainte (16) traite les derniers pas de temps de l'horizon T : à partir du pas de temps $T - dmin + 1$, aucune communication de peut débuter.

La contrainte (17) force chaque sonde à respecter un temps de reconfiguration dr après une communication ; pour toute sonde s, cette contrainte est active pour chaque pas de temps t_0 qui marque la fin d'une communication (c'est-à-dire pour t_0 tel que $X_{s,t_0-1} = 1$ et $X_{s,t_0} = 0$). On a alors $X_{s,t} = 0$ pour tout t de t_0 à $t_0 + dr - 1$.

La contrainte (18) permet d'exprimer la capacité limitée du buffer de l'orbiteur. On sait que le buffer de l'orbiteur est "vidé" à chaque communication avec la Terre (c'est-à-dire à chaque pas de temps j_i, $i = 1..J$) ; donc, dans chaque intervalle $[j_i,j_{i+1}]$, la capacité du buffer à recevoir des données des sondes est la même et elle correspond à une durée maximale de communication pour l'orbiteur ($Dmax1_{orb}$).

Enfin, les expressions (19) et (20) permettent de prendre en compte les contraintes liées à l'énergie électrique. Sur chacun des intervalles définis par les listes $\mathcal{L}$ et $\mathcal{K}_s$, on impose une durée maximale de communication qui correspond à la consommation électrique maximale sur cet intervalle. La contrainte (19) porte sur la ressource électrique de l'orbiteur et la contrainte (20) définit ces durées maximales pour chaque sonde.

Le critère d'optimisation considéré étant de maximiser la durée totale des communications orbiteur/sonde, on obtient la fonction de maximisation en sommant toutes les variables :

$$Max\ (\sum_{s=1}^{N} \sum_{t=1}^{T} X_{s,t}) \qquad (21)$$

4.3.4 Sous-problème de planification des expériences

4.3.4.1 Analyse du problème

Les quatres sondes martiennes sont identiques et comportent chacune les instruments scientifiques suivants :

- une station météorologique pour mesurer la pression, la température, l'humidité atmosphérique, etc.,
- un capteur de champ électrique pour la mesure de la conductivité électrique de l'atmosphère et de ses champs électriques,
- un radar pour la détection de poches de glace et d'eau dans le sous-sol martien,
- un magnétomètre pour l'étude des champs magnétiques, de l'ionosphère et également pour la détection d'eau dans le sous-sol,
- une caméra panoramique stéréo multispectrale pouvant réaliser différents types de prises de vues,
- deux seismomètres de différentes largeur de bande pour l'étude de l'activité sismique de Mars et la localisation de poches d'eau,
- un système radio permettant une étude de la géodésie de Mars (il est également utilisé pour les communications avec l'orbiteur),
- un ensemble de capteurs thermiques pour étudier la température du sol martien,
- et un microphone.

On peut ainsi réaliser une multitude d'expériences dont le mode de fonctionnement varie selon différents critères.

Une grande partie des expériences doivent être réalisées en réseau, c'est-à-dire sur les 4 sondes simultanément. Celles qui peuvent être effectuées indépendamment les unes des autres sont dites *multi-sites*. Le radar, le microphone et la caméra fonctionnent en multi-site, les expériences utilisant les autres instruments doivent être réalisées en réseau pour présenter un intérêt scientifique.

Certaines expériences ont une durée fixe (de quelques secondes à une heure), alors que d'autres expériences, qui assurent généralement des fonctions de veille (par exemple les expériences de météorologie ou de sismologie), devraient idéalement fonctionner de manière continue.

Enfin, la consommation électrique et la place mémoire requise sur la MdM diffèrent selon le type d'instrument utilisé et le mode de fonctionnement choisi. Par exemple, la caméra, selon le type de prise de vue que l'on souhaite obtenir, peut consommer entre 72 et 1300 mWh et nécessiter entre 1,3 et 40 Mbit de place mémoire.

On suppose connue la disponibilité au cours du temps des ressources des sondes (mémoire de masse et puissance électrique). Étant donné l'ensemble des demandes d'expériences émanant des différents centres scientifiques impliqués dans la mission, on souhaite planifier sur de courts horizons (la semaine typiquement) les expériences qui vont être réalisées par les sondes martiennes. Les plans de travail (sur chaque horizon, on détermine un plan de travail par sonde) ainsi générés doivent maximiser la "valeur" scientifique de la mission.

Plusieurs difficultés se posent pour définir formellement ce problème.

Tout d'abord, une expérience peut faire appel à l'utilisation de différents instruments (de façon simultanée ou en suivant un certain séquencement). Ainsi, le nombre total d'expériences

réalisables est *a priori* très grand ; en fait, il est même difficile à évaluer car on ne sait actuellement pas décrire explicitement la totalité des expériences envisagées par l'ensemble des centres scientifiques impliqués dans la mission.

Afin de pouvoir définir implicitement l'ensemble des expériences possibles, nous introduisons les deux notions suivantes :

- un **mode opératoire** est une procédure élémentaire d'utilisation d'un instrument permettant de réaliser un type de mesure spécifique. Un mode opératoire est caractérisé par l'instrument qu'il utilise, ses durées minimale et maximale d'exécution $\underline{Du}$ et $\overline{Du}$ (dans le cas des modes opératoires à durée fixe, $\underline{Du} = \overline{Du} = Du$), sa puissance électrique requise P (dans le cas des modes opératoires à durée fixe, l'énergie consommée est constante, égale à $P * Du$), son débit de données δ (dans le cas des modes opératoires à durée fixe, la place mémoire nécessaire est constante, égale à $\delta.Du$).

  Les modes opératoires sont de plus liés entre eux par des contraintes d'exclusion. D'une part, deux modes opératoires utilisant un même instrument ne peuvent être réalisés simultanément, d'autre part, l'utilisation de certains instruments, comme le radar par exemple, est extrêmement sensible et interdit l'utilisation en parallèle d'un instrument qui pourraît gêner la mesure (en faisant vibrer la sonde par exemple). Ainsi, on associe à chaque mode opératoire la liste des modes opératoires qui ne lui sont pas compatibles.
- une **expérience élémentaire** est la requête de mise en œuvre d'un mode opératoire sur une sonde. Une expérience élémentaire est ainsi caractérisée par le mode opératoire concerné mop, le numéro de sonde s à laquelle elle est affectée, une date de début au plus tôt est, une date de fin au plus tard lft, les durées minimale et maximale d'exécution $\underline{du}$ et $\overline{du}$ avec éventuellement $\underline{du} = \overline{du} = du$ (et dans le cas des modes opératoires à durée fixe, on a $\underline{du} = \overline{du} = du = Du$).

Une **expérience** requise par un centre scientifique est un groupement d'expériences élémentaires liées par un ensemble de contraintes. Ces contraintes, définies par le centre scientifique qui émet la requête, sont difficiles à recenser actuellement, mais on peut dresser une liste, probablement non exhaustive, des différents types de contraintes que l'on pourra rencontrer :

- des contraintes de simultanéité d'exécution de certaines expériences élémentaires sur une même sonde ou sur des sondes différentes (dates de début ou de fin d'exécution identiques, durées d'exécution identiques),
- des contraintes de séquencement des expériences élémentaires,
- des contraintes de délai minimal ou maximal entre l'exécution de deux expériences élémentaires.

Nous proposons donc d'exprimer les requêtes émanant des centres scientifiques non pas en terme d'"expériences" mais en termes d'"expériences élémentaires", telles que nous venons de les définir, et de contraintes liant ces expériences élémentaires entre elles.

Une seconde difficulté pour la formalisation du problème vient ensuite du fait que les contraintes liant les expériences élémentaires peuvent être plus ou moins "dures". En effet, certaines expériences ne peuvent avoir de valeur scientifique que dans la mesure où toutes les expériences élémentaires qui les composent sont réalisées, alors que d'autres peuvent présenter encore un intérêt à n'être réalisées que partiellement ou en ne respectant pas toutes leurs contraintes. Dans un but de simplification du problème, nous faisons l'hypothèse que toutes les contraintes du problème sont dures.

Enfin, les plans de travail doivent tendre à maximiser la valeur scientifique de la mission, c'est-à-dire satisfaire au mieux l'ensemble des scientifiques engagés dans la mission. Or, ils sont nombreux, chacun spécialiste d'un domaine d'étude de Mars ; leurs intérêts et leurs priorités sont ainsi souvent divergents. Il s'avère donc difficile d'associer une "valeur" à une expérience, une expérience élémentaire ou un mode opératoire et difficile également d'évaluer un plan de travail.

Nous ne proposons donc pas de critère d'optimisation pour ce problème. Il nous semble que l'élaboration et l'évaluation d'un plan de travail ne peuvent être faites sans une concertation des différents scientifiques avec le centre de mission, en tenant compte de l'évolution de l'environnement de la mission.

4.3.4.2 Modélisation

Nous proposons d'aborder la planification des expériences comme un problème d'aide à la décision. En effet, ce problème est combinatoire et fortement contraint (avec une partie importante des contraintes encore mal connues ou définies dans les requêtes des scientifiques) et générer un plan de travail réalisable n'est pas trivial.

En particulier, on souhaite pouvoir générer des plans de travail vérifiant les contraintes liées au partage de la mémoire de masse et de la puissance électrique des sondes. Par la suite, nous limitons notre étude aux contraintes liées à ces deux ressources.

Soit n le nombre de requêtes d'expériences élémentaires émises par l'ensemble des scientifiques pour un horizon de planification T, discrétisé en τ pas de temps.

Nous introduisons la variable binaire $E_{i,s,t}$ égale à 1 si l'expérience élémentaire i est exécutée sur la sonde s au pas de temps t, égale à 0 sinon ($i = 1..n$, $s = 1..4$ et $t = 1..\tau$).

Un plan de travail réalisable sur l'horizon T correspond a une instanciation des variables $E_{i,s,t}$ vérifiant l'ensemble des contraintes du problème.

La contrainte de partage de la puissance électrique disponible s'exprime pour chaque sonde et chaque pas de temps par l'expression :

$$\sum_{i=1}^{n} E_{i,s,t}.P_i \leq \mathcal{P}_s(t) \qquad s{=}1..4,\ t{=}1..\tau \tag{22}$$

où P_i est la puissance requise par l'expérience élémentaire i et $\mathcal{P}_s(t)$ est la puissance électrique disponible sur la sonde s au pas de temps t.

La contrainte de partage de la mémoire de masse s'exprime pour chaque sonde et chaque pas de temps par l'expression :

$$\sum_{t'=1}^{t} \sum_{i=1}^{n} E_{i,s,t'}.D_i \leq \mathcal{V}_s(t) \qquad s{=}1..4,\ t{=}1..\tau \tag{23}$$

où D_i est le débit de données produit par l'expérience élémentaire i et $\mathcal{V}_s(t)$ est la somme de la place mémoire disponible sur la sonde s au pas de temps t et du volume de données qui a été effacé de la mémoire de la sonde s depuis le premier pas de temps.

Notons que l'évaluation de $\mathcal{P}_s(t)$ et $\mathcal{V}_s(t)$ n'est pas triviale car ces valeurs dépendent des différentes expériences réalisées aux pas de temps qui précèdent t. De plus, les variables $E_{i,s,t}$ peuvent être soumises à d'autres contraintes que (22) et (23), non formulées ici, liées par exemple à l'existence de dates de début au plus tôt ou de fin au plus tard des expériences, de durées minimales ou maximales d'exécution.

4.4 Conclusion

Dans ce chapitre, nous présentons le projet Netlander, d'exploration de la planète Mars. Le système étudié se compose principalement de quatre sondes posées à la surface de Mars et d'un orbiteur qui relaie les communications entre la Terre et ces sondes.

À partir du problème d'"optimisation des boucles de progammation" de la mission, tel que posé par le CNES, nous définissons le problème de planification des communications et des expériences pour la mission Netlander. La formalisation de ce problème est complexe car de nombreux paramètres sont à prendre en compte, dont certains sont actuellement mal connus. En particulier, les critères d'optimisation ne sont pas tous clairement définis et la différenciation entre contraintes et préférences n'est pas toujours établie.

Pour aborder ce problème, nous proposons de le décomposer en deux sous-problèmes : le problème de planification des communications sondes/orbiteur d'une part et le problème de planification des expériences réalisées par les sondes d'autre part. Nous montrons que, compte-tenu de l'état actuel des connaissances sur le système, on peut traiter ces deux problèmes indépendamment l'un de l'autre.

Le premier sous-problème est un problème d'optimisation combinatoire bien défini, pour lequel nous proposons un modèle linéaire en nombres entiers basé sur une discrétisation de l'horizon temporel.

Les incertitudes ou méconnaissances du système Netlander concernent plutôt le second sous-problème. Il se pose donc plus en termes d'aide à la décision que d'optimisation. Pour ce problème, nous formulons les contraintes liées au partage des deux principales ressources : l'énergie (batteries électriques et panneaux solaires des sondes) et la place mémoire (mémoire de masse des sondes).

Après les problèmes de planification issus des projets Skybridge et Netlander, le chapitre suivant s'intéresse à la gestion des satellites Pléiades, destinés à l'observation de la Terre.

Dans la partie suivante, nous proposons des méthodes de résolution pour ces différents problèmes. Nous reviendrons en particulier sur les problèmes de planification des communications sondes/orbiteur et des expériences dans le projet Netlander, au chapitre 6. Nous résolvons le problème de planification des communications sondes/orbiteur par Programmation Linéaire en Nombres Entiers et nous développons une méthode d'aide à la décision basée sur un raisonnement énergétique pour le problème de planification des expériences.

Chapitre 5

Pléiades : Satellites d'observation de la Terre

5.1 Présentation du projet

Au milieu du XIXè siècle, le ballon a permis au photographe Félix Nadar, de réaliser au-dessus de Paris les toutes premières photographies aériennes. Au début du siècle dernier, l'avion a montré rapidement tous ses avantages en tant que plate-forme d'observation de la Terre à usages civils ou militaires. De nos jours, des avions spécialement équipés effectuent des missions pour la cartographie, l'étude des forêts, l'urbanisme, l'espionnage, la surveillance de la pollution, la recherche des ressources naturelles, etc. La photographie aérienne par avion est fiable et d'une grande précision mais elle ne peut être réalisée que dans le cadre de missions ponctuelles, limitées dans le temps et l'espace, et particulièrement coûteuses.

Lancés à partir des années 60 avec la constellation américaine **Tiros**, les satellites d'observation de la Terre offrent une alternative avantageuse à l'utilisation des avions. Ils permettent en effet d'observer, par divers moyens (instruments optiques, détecteurs thermiques, capteurs radar) et de façon répétitive, des portions de territoires immenses. De plus, les images produites sont très détaillées et d'un coût de réalisation 3 à 4 fois inférieur à celui d'une prise de vue réalisée à partir d'avion.

À la suite des satellites Tiros, utilisés essentiellement pour la prévision météorologique, de nombreux systèmes satellitaires, dédiés à différentes missions et équipés d'instruments d'observation de plus en plus performants ont été élaborés et exploités. Parmi les plus connus, on peut citer :

- la série de satellites **Landsat** développés par la NASA dans le cadre du programme américain de gestion des ressources naturelles et dont les lancements se sont étalés de 1972 pour Landsat 1 à 1999 pour Landsat 7 ;
- la constellation européenne **MétéoSat** dédiée à l'information météorologique, composée de satellites géostationnaires dont les premiers lancements remontent à 1977 ;
- les satellites français **SPOT** qui constituent le système satellitaire d'observation de la Terre le plus utilisé au monde pour l'imagerie à usage civil.

Le projet SPOT a débuté en 1978 au CNES, en coopération avec la Belgique et la Suède. Cinq satellites ont été lancés entre 1986 et 2002 et actuellement quatre d'entre-eux sont encore

utilisés (SPOT 1, 2, 4 et 5). Ils sont équipés d'instruments optiques permettant une haute résolution spatiale (de 10 m pour SPOT 1 à 2,5 m pour SPOT 5) et capables de restituer le relief à quelques mètres près grâce à un jeu de miroirs autorisant un *dépointage* (ou angle de visée) latéral par rapport à la trace au sol du satellite (figure 5.1).

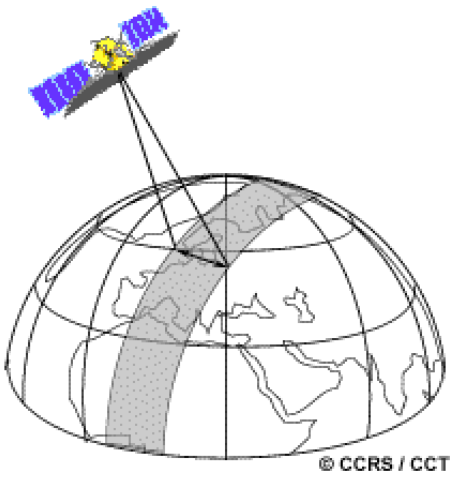


FIG. 5.1 – *Dépointage latéral d'un satellite*

Les satellites SPOT sont situés sur une orbite circulaire basse altitude (832 km), héliosynchrone, quasi-polaire (inclinée à 98,7) et phasée. Le terme héliosynchrone indique que le plan orbital du satellite forme un angle constant avec le plan orbital de la Terre autour du soleil ; ceci signifie que le satellite survole un même point du globe toujours à la même heure, ce qui assure un éclairement constant et permet la comparaison des différents clichés d'une même zone. Du fait de son orbite particulière, un satellite SPOT est éclairé lors de sa "descente" du Nord au Sud de la Terre et est plongé dans l'ombre de celle-ci lorsqu'il "remonte" du Sud vers le Nord. Ainsi, il ne peut réaliser des prises de vues que sur des "demi-révolutions", qui correspondent aux phases de descente Nord-Sud autour de la Terre (cf. figure 5.2).

La figure 5.3 donne un exemple des traces au sol "éclairées" obtenues au cours d'une journée pour un satellite SPOT.

Depuis 1999, le CNES travaille à l'élaboration d'un nouveau programme d'observation de la Terre, **Pléiades**, qui devrait succéder à la série SPOT d'ici à 2007.

Pléiades se veut une constellation de satellites d'observation de la Terre capable de couvrir l'ensemble des besoins européens (civils, militaires, pour l'industrie, le service public, etc.).

C'est un système évolutif qui devrait être constitué à terme de six satellites (deux équipés de capteurs optiques et quatre équipés de capteurs radars), utilisant le même type d'orbite que les satellites SPOT, mais plus petits, plus performants et moins coûteux que leurs prédécesseurs.

Un avantage particulier de ces satellites "nouvelle génération" est leur *agilité*, c'est-à-dire qu'ils peuvent, tout en se déplaçant sur leur orbite, se mouvoir en attitude selon les trois axes de roulis, tangage et lacet (cf. figure 5.4). Ils ont ainsi une capacité de dépointage de leur instrument de mesure – optique ou radar – dans toutes les directions (et non pas uniquement en roulis comme pour les satellites SPOT).

Ceci leur permet de réaliser des clichés latéralement, "en avant" et "en arrière" par rapport

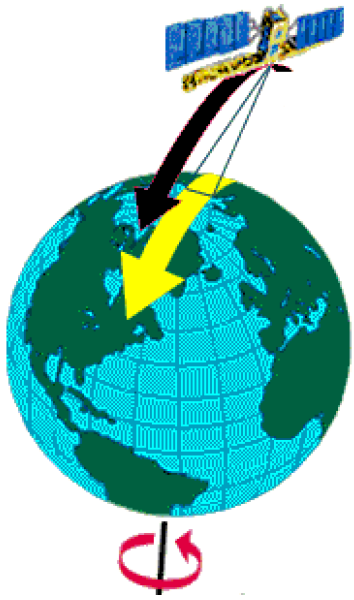


FIG. 5.2 – *Phase de descente Nord-Sud d'un satellite en orbite quasi-polaire*

à leur trace au sol (cf. figures 5.5 et 5.6).

Dans la suite, nous nous focalisons sur les satellites Pléiades munis d'instruments optiques.

5.2 Problématique

Les satellites agiles d'observation sont commandés depuis le centre de mission au sol, de façon à répondre à des requêtes d'images de certaines zones terrestres, émanant de façon continue des différents utilisateurs du système.

Une requête est caractérisée par :

- une zone géographique qui peut être une *cible* (un cercle de dimension limitée) ou un *polygone* (une large zone de forme polygonale) ;
- un intervalle de validité, exprimé par l'utilisateur (en jours dans la plupart des cas) ;
- des contraintes angulaires d'acquisition (angles minimaux et maximaux en roulis et tangage) ;
- un type qui peut être mono ou stéréo ; dans le cas d'une demande stéréo, il faut photographier deux fois la zone en respectant des contraintes angulaires entre les deux photographies ;
- un poids ou une priorité qui exprime son importance aux yeux de l'utilisateur et/ou du centre de mission.

Le cœur des instruments optiques d'un satellite est un alignement de photo-diodes qui permet d'acquérir à chaque instant l'image d'un segment sur la surface terrestre. C'est la combinaison du mouvement du satellite agile sur son orbite et sur lui-même qui permet d'acquérir l'image d'une bande comme la conjonction d'images de segments contigus. Ces bandes sont rectangulaires, et considérées de largeur fixe (*fauchée au sol* du satellite). Leur longueur, ainsi que la direction d'acquisition restent libres.

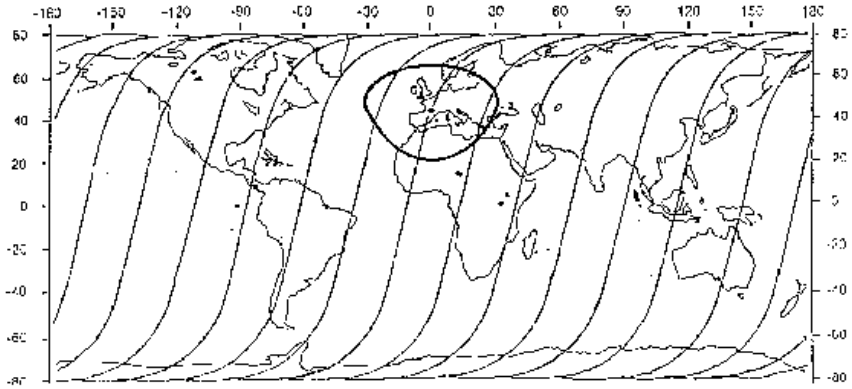


FIG. 5.3 – *Traces au sol d'un satellite SPOT sur une journée*

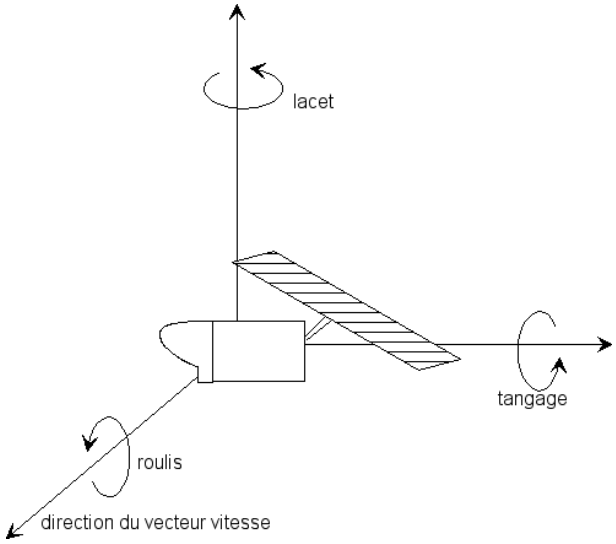


FIG. 5.4 – *Axes de roulis, tangage et lacet d'un satellite*

La fauchée au sol de l'instrument est suffisante pour acquérir une cible en une seule "passe", c'est-à-dire avec une seule bande. Elle n'est en revanche pas suffisante pour acquérir en un seule passe les polygones. Ceux-ci sont donc prédécoupés en bandes parallèles, de même largeur et de longueur variable. Les bandes résultant du découpage d'un polygone peuvent être acquises à partir de révolutions différentes du satellite.

Du fait de l'agilité des satellites, en particulier leur capacité de dépointage en tangage, une bande peut être de plus acquise dans un sens ou dans l'autre. On appelle **prise de vue** l'acquisition photographique d'une bande par le satellite ; à une même bande correspondent donc deux prises de vue possibles (figure 5.7). Avant d'effectuer une prise de vue, le satellite doit se positionner de façon à orienter correctement son instrument optique. Les mouvements nécessaires dépendent largement de la position dans laquelle il se trouvait auparavant (c'est-à-dire, de la prise de vue précédente) et leur durée n'est pas négligeable.

Dans le cas de requêtes stéréoscopiques, chaque bande est dédoublée et les bandes *jumelles*

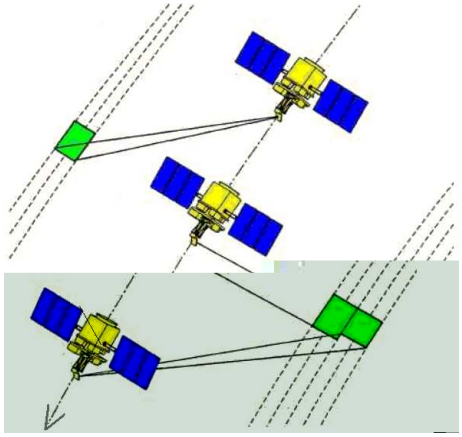


FIG. 5.5 – *Agilité d'un satellite Pléiades (3 positions d'un même satellite)*

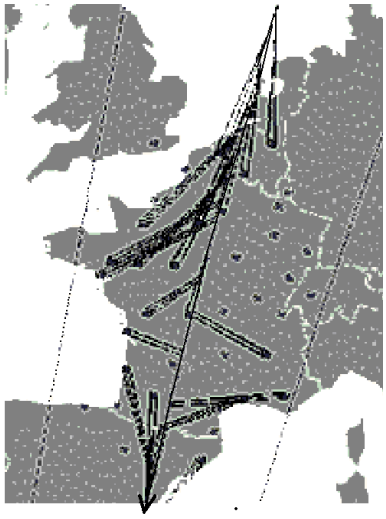


FIG. 5.6 – *Exemple d'acquisitions d'images le long d'une trace au sol*

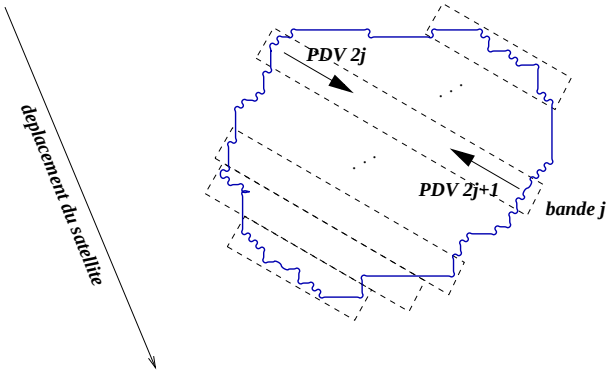


FIG. 5.7 – *Découpage en bandes d'une requête*

doivent être acquises dans le même sens (et depuis la même orbite) ; donc à chaque couple de bandes jumelles correspondent seulement deux – et non pas quatre – couples de prises de vue possibles.

On sait que le nombre de requêtes est très élevé ; on cherche donc à planifier les images à réaliser par les différents satellites au cours du temps, de façon à satisfaire au mieux l'ensemble des utilisateurs.

Il se pose donc au préalable un problème d'affectation des requêtes aux différents satellites et un problème de détermination de l'horizon de planification.

Dans cette étude, nous considérons que le problème d'affectation entre les satellites est déjà résolu et nous choisissons comme horizon de travail une demi-révolution – éclairée – d'un satellite autour de la Terre (effectuée en 50 minutes environ).

Pour une demi-révolution, nous supposons connu l'ensemble des requêtes d'images des différents utilisateurs. De plus, le prédécoupage des polygones des requêtes est défini à l'avance ;

on a ainsi un ensemble de bandes candidates, chacune caractérisée par :

- la requête dont elle est issue ;
- deux intervalles de réalisation (un pour chaque prise de vue) qui dépendent des contraintes angulaires et de l'intervalle de validité de la requête associée, ainsi que de la fenêtre de visibilité du satellite avec la bande ;
- une durée d'acquisition ; la vitesse d'acquisition de l'instrument étant constante, on considère que la durée d'acquisition d'une bande est proportionnelle à sa longueur.

Étant donné un ensemble de requêtes d'images – prédécoupées en bandes – dont on peut déduire un ensemble de prises de vue candidates, il s'agit donc de sélectionner et ordonnancer les prises de vues que le satellite devra réaliser sur une demi-révolution, de façon à satisfaire au mieux l'ensemble des clients.

Pour évaluer la satisfaction des clients, on suppose que l'on sait associer un gain à chaque requête. Deux méthodes de calcul sont alors envisagées.

Une première évaluation consiste à associer un gain à chaque bande candidate, proportionnellement à la fraction de requête qu'elle représente et à sommer les gains des bandes acquises. Avec ce mode de calcul linéaire, le fait de photographier une requête entièrement n'est pas privilégié par rapport au fait de photographier, uniquement partiellement, plusieurs requêtes. Or, du point du vue du client, il n'est pas évident que la réalisation de 50% d'une requête, soit à 50% satisfaisante, il est donc souhaitable de privilégier l'acquisition totale des requêtes.

Une seconde méthode d'évaluation consiste ainsi à pondérer non linéairement la fraction acquise de chaque requête, à calculer les gains correspondants et à sommer ces gains. De cette façon, on considère par exemple qu'une requête acquise à 40% apporte 10% du gain total associé à la requête, qu'une requête acquise à 70% apporte 40% du gain, etc. Nous reviendrons par la suite sur ces aspects de définition du critère d'optimisation du problème (paragraphe 5.3.2).

Le problème de sélection et ordonnancment des prises de vue d'un satellite Pleiades apparaît d'emblée difficile, du fait de l'agilité des satellites et du grand nombre de requêtes qui en font un problème fortement combinatoire.

5.3 Analyse du problème

5.3.1 Revue de la littérature

Une première étude de ce problème a été réalisée par l'ONERA du point du vue de sa modélisation et de sa résolution [Lemaître *et al.* 02]. Sur la base d'un même modèle utilisant des variables de type "successeur" (une variable booléenne est associée à chaque succession de deux prises de vue), quatre algorithmes sont testés et comparés pour la recherche de solutions en un temps de calcul limité à 2 minutes :

- un algorithme glouton et un algorithme de Programmation Dynamique qui ne prennent pas en compte les contraintes stéréoscopiques et qui considèrent le critère d'optimisation linéaire,
- un algorithme de Programmation Par Contraintes et un algorithme Tabou de Recherche Locale qui prennent en compte toutes les contraintes ainsi que le critère non linéaire.

Compte tenu du modèle choisi et du temps de calcul fixé, l'algorithme de recherche locale fournit les résultats les plus satisfaisants.

Par la suite, le problème Pléiades a été soumis comme sujet du challenge ROADEF 2003[1] ; il s'agit là de trouver, en un temps limité à 3 minutes, une solution respectant l'ensemble des contraintes et maximisant un critère de qualité non linéaire. À cette occasion, différentes méthodes approchées ont été proposées, utilisant une recherche Tabou [Cordeau & Laporte 03, Jaumard *et al.* 03, Habet & Vasquez 03], le recuit simulé [Kuipers 03, Vasquez *et al.* 03, Wild *et al.* 03], ou un algorithme génétique [Ribeiro 03] ou encore la programmation par contraintes dans un contexte de recherche arborescente partielle de type LDS (*Limited Discrepancy Search* – [Harvey & Ginsberg 95]) [Caseau 03, de Givry & Jeannin 03]. L'algorithme de recuit simulé présenté par [Kuipers 03] fournit les meilleurs résultats ; il utilise un paramétrage *ad hoc* des différentes équations de la méthode et définit le voisinage d'une solution (séquence admissible de prises de vues) comme l'ensemble des séquences admissibles que l'on peut obtenir en retirant les prises de vues issues d'au plus 4 requêtes et en ajoutant un nombre aléatoire de prises de vues issues d'au plus 4 requêtes.

D'autres travaux ont été menés sur des problèmes proches de gestion de missions de satellites d'observation de la Terre. En particulier, [Gabrel 99] puis [Vasquez & Hao 03] s'intéressent au calcul de bornes supérieures pour le problème de planification de prises de vues du satellite SPOT5 (voir chapitre 5 et [Bensana *et al.* 99] pour une présentation plus détaillée de la problématique SPOT5). [Gabrel 99] propose un modèle linéaire du problème qui permet d'obtenir les premières bornes supérieures par résolution de la relaxation linéaire, puis améliore sensiblement ces bornes en décomposant le modèle et en utilisant une procédure de génération de colonnes pour sa résolution. Plus récemment, [Vasquez & Hao 03] propose une approche dite "partition/backtrack-dynamique" qui consiste à effectuer une partition des variables du problème grâce à une recherche Tabou, puis à résoudre chaque sous-problème ainsi obtenu de façon exacte par un algorithme de backtracking itératif inspiré du principe des poupées russes. Cette méthode améliore les bornes connues et permet de traiter des instances de plus grande taille.

Par ailleurs, [Wolfe & Sorensen 00] propose trois algorithmes différents pour l'ordonnancement de tâches dans le domaine de l'observation de la Terre par satellite : un algorithme glouton, un algorithme utilisant une procédure de "look-ahead" puis un algorithme génétique. D'une façon générale, les algorithmes gloutons sont largement utilisés pour fournir des solutions ; ils sont ensuite couplés à des techniques heuristiques d'amélioration, très dépendantes du problème spécifique considéré (cf. [Pemberton 00], [Frank *et al.* 01], [Bresina *et al.* 97]).

5.3.2 Modélisation

Nous introduisons les indices et paramètres suivants :

- T : horizon de planification (correspondant à une demi-révolution du satellite autour de la Terre, soit 50 minutes environ),
- N_R : nombre total de requêtes,
- r : numéro de requête, $r = 1$ à N_R,
- Nb_r : nombre de bandes issues du prédécoupage de la requête r,

1. ROADEF : Société Française de Recherche Opérationnelle et d'Aide à la Décision ; détails du challenge sur `http://www.prism.uvsq.fr/~vdc/ROADEF/CHALLENGES/2003/challenge2003_en.html`

- N_B : nombre total de bandes candidates,
- j : numéro de bande, $j = 0$ à $N_B - 1$,
- k : numéro de prise de vue ; à toute bande j correspondent les deux prises de vues $2j$ et $2j + 1$; on a donc $k = 0$ à $2N_B - 1$ (figure 5.7),
- N_{sup} : nombre maximal de prises de vue réalisables sur l'horizon T (ce paramètre n'est pas connu *a priori*, mais par défaut on peut fixer $N_{sup} = N_B$),
- i : position dans une séquence solution de prises de vue à exécuter, $i = 1$ à N_{sup},
- $Tmin0(j)$, $Tmax0(j)$: dates de début au plus tôt et au plus tard de la prise de vue $2j$,
- $Tmin1(j)$, $Tmax1(j)$: dates de début au plus tôt et au plus tard des prises de vue $2j+1$,
- $Du(j)$: durée d'acquisition pour la bande j ; les durées des prises de vue $2j$ et $2j + 1$ sont identiques,
- $Dt(k,k')$: durée de transition entre la fin de la prise de vue k et le début de la prise de vue k' ; cette durée est induite par le changement d'attitude du satellite nécessaire entre deux prises de vue (notons que pour une bande j et une prise de vue k, $Dt(2j,k) \neq Dt(2j + 1,k)$). L'évaluation de cette durée n'est pas triviale et fait l'objet d'études spécifiques ; l'hypothèse retenue considère que $Dt(k,k')$ ne dépend pas de la date de fin exacte de la prise de vue k [Lemaître *et al.* 02],
- Tw : vecteur des "bandes jumelles" ; pour $j = 1$ à $N_B - 1$, $Tw(j) = -1$ si la bande j n'est pas issue d'une requête stéréoscopique, $Tw(j) = j'$ où j' est la bande jumelle de j sinon,
- Str : matrice indiquant, pour chaque requête et chaque bande issue de cette requête, le numéro de bande associé,
- $R(j)$: requête dont est issue la bande j,
- $S(r)$: surface de la requête r,
- $Su(j)$: surface utile de la bande j, *i.e.,* surface de l'intersection de la bande j avec la requête associée,
- $G(r)$: gain associé à l'acquisition totale de la requête r.

[Lemaître *et al.* 02] présente un modèle mathématique du problème Pléiades qui utilise trois types de variables : variables réelles de date de début de prise de vue, variables binaires de sélection d'une prise de vue et variables binaires de succession entre deux prises de vue. Nous proposons ci-après un modèle différent, faisant appel aux deux variables suivantes :

- $se(k,i)$, variable binaire égale à 1 si la prise de vue k est sélectionnée en position i, égale à 0 sinon ;
- $t(i)$, variable réelle égale à la date de début de la prise de vue en position i dans la séquence construite.

Les contraintes du problème s'expriment par les expressions suivantes :

$$\sum_i (se(2j,i) + se(2j+1,i)) \leq 1 \qquad j=0..N_B-1 \qquad (24)$$

$$\sum_j (se(2j,i) + se(2j+1,i)) \leq 1 \qquad i=1..N_{sup} \qquad (25)$$

$$\sum_j (se(2j,i+1) + se(2j+1,i+1)) \leq \sum_j (se(2j,i) + se(2j+1,i)) \qquad i=1..N_{sup}-1 \qquad (26)$$

$$\sum_j (Tmin0(j).se(2j,i) + Tmin1(j).se(2j+1,i)) < t(i) \qquad i=1..N_{sup} \qquad (27)$$

$$t(i) < \sum_j (Tmax0(j).se(2j,i) + Tmax1(j).se(2j+1,i)) \qquad i=1..N_{sup} \qquad (28)$$

$$\begin{aligned} t(i+1) > {} & t(i) + Du(j) + \sum_l Dt(2j,2l).se(2l,i+1) \\ & + \sum_l Dt(2j,2l+1).se(2l+1,i+1) \\ & + T(se(2j,i) - 1) \end{aligned} \qquad i=1..N_{sup},\ j=0..N_B-1 \qquad (29)$$

$$\begin{aligned} t(i+1) > {} & t(i) + Du(j) + \sum_l Dt(2j+1,2l).se(2l,i+1) \\ & + \sum_l Dt(2j+1,2l+1).se(2l+1,i+1) \\ & + T(se(2j+1,i) - 1) \end{aligned} \qquad i=1..N_{sup},\ j=0..N_B-1 \qquad (30)$$

$$\sum_i se(2j,i) = \sum_i se(2Tw(j),i) \qquad j=0..N_B-1 \mid Tw(j)>j \qquad (31)$$

$$\sum_i se(2j+1,i) = \sum_i se(2Tw(j)+1,i) \qquad j=0..N_B-1 \mid Tw(j)>j \qquad (32)$$

Les contraintes (24) et (25) limitent chaque bande à être photographiée au plus une fois et chaque position dans la séquence à être occupée par au plus une prise de vue. La contrainte (26) interdit les "trous" dans la séquence : s'il n'y a aucune prise de vue sélectionnée en position i dans la séquence, alors il n'y a plus aucune prise de vue sélectionnée pour tout $i' = i+1$ à N_{sup}.

Les expressions (27) à (30) expriment les contraintes temporelles du problème. Les contraintes (27) et (28) concernent les fenêtres de réalisation des différentes prises de vue : la date de début au plus tôt (respectivement au plus tard) de la prise de vue placée en position i dans la séquence est $Tmin0(j)$ ou $Tmin1(j)$ (respectivement $Tmax0(j)$ ou $Tmax1(j)$) selon que la prise de vue est de la forme $2j$ ou $2j+1$ (*i.e.,* selon le sens d'acquisition de la bande). Une durée de transition, entre la fin d'une prise et le début de la suivante, est assurée grâce aux contraintes (29) et (30). (29) traite le cas où une prise de vue de forme $2j$ est en position i et (30) traite le cas où une prise de vue de forme $2j+1$ est en position i. Ces deux contraintes font appel à l'utilisation d'une constante de type "grand M" (ici T) les rendant inactives dans le cas où la prise de vue $2j$ (respectivement, $2j+1$) n'est pas sélectionnée en position i dans la séquence.

Enfin, les expressions (31) et (32) expriment les contraintes stéréoscopiques du problème : pour tout couple de bandes jumelles, j et $Tw(j)$, soit ces deux bandes sont acquises, dans le même sens, soit aucune des deux ne l'est. La contrainte (31) traite le cas où les bandes jumelles sont acquises par des prises de vue de forme $2j$ et la contrainte (32) traite celui où les prises de vue sont de forme $2j+1$. Ces contraintes sont définies pour tout $j = 1..N_B$ tel que $Tw(j) > j$, ce qui permet d'ignorer les bandes non stéréoscopiques ($Tw(j) = -1 < j$) et de ne pas avoir de contrainte redontante (les couples (j, $Tw(j)$) et ($Tw(j)$, j) sont traités par la même contrainte).

La satisfaction de l'ensemble des utilisateurs, que l'on souhaite maximiser, se mesure sur la base des gains $G(r)$, chacun associé à l'acquisition d'une requête et que l'on pondère d'un coefficient qui tient compte de la fraction acquise $fr(r)$ de cette requête.

Soit Su_{l_r} la surface acquise par la bande $Str(r,l)$ (pour $r = 1$ à N_R et $l = 1$ à Nb_r), on a l'égalité :

$$Su_{l_r} = \sum_{i=1}^{N_{sup}} Su(Str(r,l)) \times [se(2(Str(r,l),i) + se(2(Str(r,l)+1,i)] \qquad r=1..N_R,\ l=1..Nb_r \tag{33}$$

La fraction acquise d'une requête r, $f(r)$,est donc définie par :

$$fr(r) = \frac{1}{S(r)} \times \sum_{l=1}^{Nb_r} Su_{l_r} \qquad r=1..N_R \tag{34}$$

Afin d'exprimer le fait que l'on souhaite favoriser l'acquisition totale des requêtes (le but étant de ne pas multiplier le nombre de requêtes partiellement satisfaites), ce coefficient évolue de manière non linéaire et suivant une pente qui croît avec l'augmentation de la fraction acquise comme illustré sur la figure 5.8. Ceci nous conduit à la fonction de maximisation :

$$Max \ \sum_r G(r) \times g(fr(r)) \tag{35}$$

où $g(.)$ est la fonction représentant l'évolution du facteur de pondération.

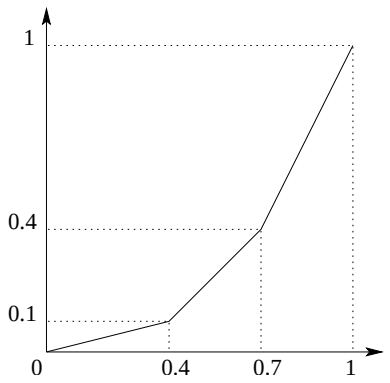


FIG. 5.8 – *Fonction g de pondération du gain*

La fonction $g(.)$ rend le critère (35) non linéaire. Dans un but de simplification du problème, nous linéarisons cette fonction en posant $g(fr(r)) = fr(r), \forall\, r$. On a alors l'égalité :

$$\sum_r G(r) \times fr(r) = \sum_j gain(j).\sum_i (se(2j,i) + se(2j+1,i))$$

où $gain(j) = G(R(j)) * Su(j)$ est le gain associé à l'acquisition de la bande j. Ceci nous permet d'obtenir le critère de maximisation suivant, linéaire et supérieur au critère (35) :

$$Max \ \sum_j gain(j).\sum_i (se(2j,i) + se(2j+1,i)) \tag{36}$$

On nomme P le modèle linéaire ainsi obtenu pour le problème Pléiades, constitué des contraintes (24) à (32) et du critère de maximisation (36).

5.4 Étude de la complexité du problème Pléiades

On peut pressentir que le problème Pléiades appartient à la classe des problèmes NP-difficiles[2]. Pour le démontrer, nous proposons une réduction du problème de décision associé au problème de Voyageur de Commerce Sélectif (STSP pour *Selective Traveling Salesman Problem*) vers le problème de décision associé au problème Pléiades.

Soit $J = \{1..N\}$ un ensemble de clients. On associe un gain $g_j \geq 0$ à chaque client $j \in J$ et un temps de parcours $d_{jl} \geq 0$ entre chaque couple de clients $(j,l) \in J^2$. Le STSP consiste à déterminer une tournée de gain maximal, partant d'un dépôt o, visitant au plus une fois chaque client et dont la durée ne dépasse pas une limite imposée T. [Golden *et al.* 87] puis [Laporte & Martello 90] montrent que ce problème est NP-difficile.

Notons STSP_{dec} et Pléiades_{dec} les problèmes de décision associés respectivement au STSP et au problème Pléiades. Pour montrer que le problème Pléiades est NP-difficile, montrons que Pléiades_{dec} est NP-complet.

À la différence du problème d'optimisation Pléiades, Pléiades_{dec} cherche à sélectionner et ordonnancer un ensemble de prises de vues permettant d'obtenir un gain supérieur à une valeur fixée, g, plutôt qu'un gain optimal. Soit une solution admissible du problème Pléiades, il est possible de vérifier en un temps polynomial en la taille des données si cette solution est admissible pour Pléiades_{dec}, c'est-à-dire si elle vérifie l'ensemble des contraintes du problème et si le gain obtenu est supérieur à g. Le problème Pléiades_{dec} appartient donc à la classe NP.

En reprenant les notations précédentes, STSP_{dec} consiste à déterminer s'il existe une tournée partant d'un dépôt o, visitant au plus une fois chaque client de J, d'une durée totale inférieure ou égale à T et collectant un gain supérieur à une valeur g.

Sans perte de généralité pour le STSP, on peut introduire les éléments suivants, issus du modèle P proposé précédemment pour le problème Pléiades :

- à chaque client $j \in J$, on associe :
  - deux prises de vues $2j$ et $2j+1$,
  - les intervalles de réalisation $[Tmin0(j), Tmax0(j)] = [0,T]$ et $[Tmin1(j), Tmax1(j)] = [0,0]$,
  - une durée $Du(j) = 0$,
  - l'indicateur de bande jumelle $Tw = -1$ (pas de bande jumelle),
  - un gain $gain(j) = g_j$;
- à tout couple de clients $(j,l) \in J^2$, on associe les durées de transition $Dt(2j,2l) = d_{jl}$ et $Dt(2j,2l+1) = Dt(2j+1,2l) = Dt(2j+1,2l+1) = 0$.

On peut alors modéliser le STSP_{dec} comme une instance du problème Pléiades_{dec}, où seules les prises de vues de forme $2j$ peuvent être réalisées (les intervalles de réalisation des prises de vues de forme $2j+1$ sont vides) et dans laquelle il n'y a pas de requête stéréoscopique ($Tw = -1$ pour tout $j \in J$). Ainsi, répondre au problème Pléiades_{dec} permet de répondre au problème STSP_{dec}. On peut donc réaliser une réduction du problème STSP_{dec} vers le

2. Un problème d'optimisation est NP-difficile si le problème de décision associé (problème pour lequel la réponse est *oui* ou *non*) est NP-complet. Un problème de décision, D, est NP-complet s'il appartient à la classe NP (si on peut vérifier en un temps polynomial qu'une instance donnée permet de répondre affirmativement au problème) et s'il existe un problème D', NP-complet, que l'on peut réduire polynomialement en D (D est une généralisation de D'). Pour une présentation plus rigoureuse et formelle de la théorie de la complexité nous renvoyons à la lecture d'ouvrages spécialisés comme [Garey & Johnson 79].

problème Pléiades$_{dec}$; notons que cette réduction s'effectue en temps polynomial. Comme STSP$_{dec}$ est NP-complet, Pléiades$_{dec}$ est également NP-complet. Nous pouvons donc conclure que le problème Pléiades est un problème NP-difficile.

5.5 Conclusion

Dans ce chapitre, nous présentons la mission d'observation de la Terre des satellites Pléiades. Il s'agit de satellites plus "agiles" que ceux actuellement en place (comme les satellites SPOT) et qui induisent de ce fait un nouveau problème complexe de planification.

Contrairement aux problèmes relatifs au projet Netlander, vus au chapitre précédent, le problème Pléiades a déjà fait l'objet de travaux de recherche concernant sa modélisation et sa résolution. Nous proposons un modèle original, sous forme d'un programme linéaire en variables mixtes, et nous montrons que ce problème d'optimisation combinatoire est NP-difficile.

Face à la grande combinatoire du problème (plusieurs centaintes de requêtes) et compte tenu des temps de résolution très courts imposés en pratique, la plupart des approches de résolution proposées font appel à des heuristiques. Toutefois, dans le but de générer des bornes supérieures, voire de fournir certaines solutions optimales, nous poursuivons, au chapitre 7, l'étude de ce problème dans la voie des méthodes exactes. Nous proposons une décomposition du modèle, basée sur l'observation des similitudes que présente le problème Pléiades avec le STSP (problème de voyageur de commerce sélectif) et nous développons une méthode de résolution par Génération de Colonnes.

Ce chapitre conclut la deuxième partie du mémoire. Dans cette partie, nous avons présenté trois projets de missions spatiales développés au CNES, et proposé des modèles pour les problèmes de planification qui en découlent : un problème d'allocation de liens de communication dans la constellation Skybridge, un problème de planification de communications et de planification d'expériences pour le projet Netlander, et un problème de sélection et ordonnancement de prises de vue d'un satellite Pléiades. On peut constater que ces problèmes se posent à différents stades de la vie d'un projet et que le travail de modélisation qui en résulte est de difficulté variable. Dans le cas de Skybridge et surtout Pléiades, le système est déjà bien défini et on dispose de documents faisant état de la description du problème ou de travaux relatifs à la modélisation et la résolution de ces problèmes. Le projet Netlander est quant à lui dans une phase amont, où de nombreuses décisions restent à prendre concernant les caractéristiques physiques et fonctionnelles du système. Dans ce cas, une partie considérable de l'étude a consisté précisément à caractériser la problématique et à la formaliser [Mancel & Saleman 01, Mancel *et al.* 01a], avant même d'en donner un modèle mathématique.

Dans la troisième et dernière partie, nous poursuivons l'étude sur les projets Netlander (au chapitre 6) et Pléiades (au chapitre 7), par la proposition de méthodes de résolution.

Troisième partie

Résolution de problèmes de planification dans les projets Netlander et Pléiades

Chapitre 6

Planification de communications et d'expériences dans le projet Netlander

6.1 Introduction

Nous avons vu au chapitre 4 que le problème de planification des communications et des expériences dans le projet Netlander peut être décomposé en deux sous-problèmes considérés comme indépendants. Le premier sous-problème est celui de la planification des communications entre l'orbiteur de Mars et les sondes ; c'est un problème d'optimisation combinatoire que nous avons modélisé par un programme linéaire en nombres entiers. Le second sous-problème est celui de la planification des expériences réalisées par les sondes. Différents paramètres de ce problème sont encore méconnus (en particulier, on ne sait pas définir un critère de qualité associé à un plan de travail et il est difficile de différencier les contraintes et les préférences) et générer un plan de travail réalisable par les sondes est déjà complexe et fera nécessairement l'objet de négociations entre différents experts engagés dans la mission. Nous l'abordons donc comme un problème d'aide à la décision.

Dans ce chapitre, nous résolvons d'une part, le problème de planification des communications orbiteur/sondes par une méthode classique de PLNE et nous généralisons le problème au cas où plusieurs orbiteurs peuvent être utilisés pour les communications. D'autre part, pour le problème de planification des expériences, nous proposons un calcul de courbes d'évaluation de la charge des deux principales ressources, que sont l'énergie électrique et la mémoire de masse, en faisant appel à des concepts de raisonnement énergétique. Ces courbes peuvent être utilisées pour guider l'élaboration des plans de travail d'une sonde.

6.2 Planification des communications orbiteur/sondes

Le problème de la planification des communications orbiteur/sondes dans le projet Netlander consiste, étant donné l'ensemble des fenêtres de visibilité entre les sondes et l'orbiteur martien sur un horizon T, à déterminer les créneaux de communication de chaque sonde sur T, de façon à maximiser la durée totale de communication.

Ce problème peut être modélisé par le programme linéaire en nombres entiers proposé au chapitre 4 (pages 75 et 76).

Le CNES a calculé les éphémérides de l'orbiteur sur des périodes de trente jours (avec un pas de temps de deux minutes), pour différents scénarios susceptibles d'être mis en œuvre dans le projet Netlander. On souhaite obtenir une solution du problème sur ces 30 jours. Les données fournies nous ont incité à tenter une résolution directe par des outils de programmation linéaire. En effet, pour chaque sonde, on remarque que les périodes de visibilités avec l'orbiteur sont très minoritaires devant les périodes de "non-visibilité". Ainsi, même si la taille du problème sur un horizon de trente jours (21 600 pas de temps) est considérable (de l'ordre de 85 000 variables et 300 000 contraintes), il s'avère que le nombre de pas de temps t sur lesquels les variables d'affectation $X_{s,t}$ peuvent être non nulles est très limité (de l'ordre de 200 pour chaque sonde s). Par conséquent, une grande majorité de variables sont fixées à zéro (par les contraintes (13)) et un grand nombre des contraintes (15) et (17) sont ainsi rendues "inactives".

Dans la suite, nous présentons les différents tests effectués sur le problème initial, puis nous étudions les limites d'une résolution par programmation linéaire en généralisant ce problème de planification des communications au cas où plusieurs orbiteurs peuvent être utilisés.

6.2.1 Description des orbites testées et des outils utilisés

Les trois scénarios fournis par le CNES diffèrent uniquement par l'orbite considérée pour calculer les éphémérides de l'orbiteur martien. Dans tous les cas, on a les caractéristiques suivantes :

- quatre sondes martiennes dont les positions sont spécifiées dans le tableau 6.1 ;
- un degré d'élévation minimum de 20° pour établir les communications orbiteur/sonde ;
- un horizon de simulation de 30 jours avec une discrétisation du temps par pas de 2 minutes (21 600 pas).
- la durée minimale d'une communication, $dmin$, est de 6 minutes (3 pas de temps) ;
- la durée de reconfiguration de l'orbiteur, dr, est de 2 minutes (1 pas de temps) ;
- la durée de communication est limitée pour chaque sonde à 40 minutes (20 pas de temps) par jour (on pose $\mathcal{K} = \mathcal{J}$ et $Dmax_s(j_i,j_{i+1}) = 20$ pour $i = 1..J$) ;
- la capacité du buffer, ainsi que les ressources électriques de l'orbiteur n'étant pas définies, on relâche les contraintes de durée maximale de communication de l'orbiteur (on pose $Dmax1_{orb}(t,t') = \infty$ et $Dmax2_{orb}(t,t') = \infty$ pour tout intervalle $[t,t']$) ;

sonde i	longitude (degrés)	latitude (degrés)	altitude (mètres)
1	71	-22	0
2	-132	35	0
3	-60	12	0
4	-75	25	0

TAB. 6.1 – *Position des sondes martiennes*

Les différentes orbites envisagées sont décrites dans [Ridder *et al.* 01] ; ce sont des orbites héliosynchrones et basse altitude (entre quatre cents et huit cents kilomètres). Pour chaque scénario, nous disposons de deux jeux de données : le premier contient les visibilités de l'orbiteur pour les différentes sondes pendant leurs périodes d'ensoleillement et le second contient

toute les visibilités (à la fois de jour et de nuit). Rappelons que dans le mode de fonctionnement nominal, les sondes sont supposées ne pas communiquer avec l'orbiteur lorsqu'elles ne sont pas ensoleillées, nous avons toutefois testé notre modèle sur les deux types de jeux de données.

Le modèle linéaire a été implémenté et testé sur une station Sun-Blade-100 à 502MHz, possédant 604Mo de RAM. Les programmes ont été développés en C++, compilés avec CC 5.0. Nous utilisons l'outil de programmation linéaire ILOG CPLEX 7.5 (avec les bibliothèques ILOG CONCERT 1.2) pour la résolution du problème relâché ainsi que la recherche arborescente de la solution entière.

6.2.2 Résultats du modèle initial

Les résultats obtenus sur les six jeux de données sont présentés dans le tableau 6.2.

	jeu de données	temps CPU (secondes)	solution PL relâché	solution optimale entière
scénario 1	jour (sc1.1)	80	702	695
	jour et nuit (sc1.2)	86	1705	1705
scénario 2	jour (sc2.1)	73	263,75	257
	jour et nuit (sc2.2)	74	646	638
scénario 3	jour (sc3.1)	75	724	724
	jour et nuit (sc3.2)	79	1710	1710

TAB. 6.2 – *Résultats obtenus par programmation linéaire pour le problème de planification des communications orbiteur/sondes*

Pour chaque scénario $i = 1$ à 3, nous avons testé le modèle sur l'horizon de planification total (21 600 pas de temps), en considérant d'une part, uniquement les visibilités orbiteur/sondes de jour (jeu sc$i$.1) et, d'autre part, toutes les visibiltés orbiteur/sondes (jeu sc$i$.2). Pour chaque jeu de données, le tableau 6.2 donne le temps CPU d'exécution du programme, la solution optimale de la relaxation linéaire du problème (durée totale de communication en nombre de pas de temps) et la solution optimale entière du problème.

Pour les six tests effectués, on obtient la **solution optimale** du problème sur l'horizon total de **trente jours**, en moins de **90 secondes**. Dans la moitié des cas (sc1.2, sc3.1 et sc3.2), on note que la solution du problème relâché est la solution entière (les valeurs 702 et 646 obtenues pour les jeux de données sc1.1 et sc2.2 ne correspondent pas à des solutions entières). Pour les autres cas, la solution du problème relâché est très proche de la solution entière (1,4 % en moyenne).

Ces résultats sont satisfaisants puisqu'ils nous permettent de répondre de façon optimale et en un temps très limité au problème de planification des communications orbiteur/sondes

sur un horizon de 30 jours ; ceci même lorsque l'on prend en compte les visibilités de nuit en plus de celles de jour, ce qui double, en moyenne, la taille des jeux de données (jeux sc1.2, sc2.2 et sc3.2).

À partir des solutions obtenues, on peut donner une première analyse des orbites proposées dans les différents scénarios.

En effet, on observe que la durée totale de communication sur une durée de 30 jours varie, selon le scénario choisi, de 514 minutes (257 pas de temps pour sc2.1, soit environ 8 heures) à 1448 minutes (724 pas de temps pour sc3.1, soit environ 24h) dans le cas nominal de fonctionnement, où on ne considère que les visibilités de jour (sc1.1, sc2.1 et sc3.1). Ceci donne une durée de communication moyenne par jour martien (appelé *sol*, avec 1 *sol* = 24 heures 37 minutes) et par sonde variant entre 4 et 12 minutes. Or, les études menées au CNES sur le projet Netlander préconisent d'utiliser des orbites martiennes permettant d'assurer une moyenne de 40 à 60 minutes de communication orbiteur/sonde par *sol* et par sonde en début de vie et 20 minutes en fin de vie, avec au moins une communication par *sol* et par sonde [Mancel & Saleman 01, Marsal *et al.* 01]. Le premier constat que l'on peut faire est que les orbites proposées dans les trois scénarios que nous avons testés ne semblent pas compatibles avec ces exigences.

De plus, même dans le cas où l'on autorise les communications pendant les périodes de nuit des sondes (sc1.2, sc2.2 et sc3.2), bien que la durée totale de communication augmente considérablement (elle est plus que doublée), la durée moyenne de communication par *sol* et par sonde reste inférieure à 30 minutes (14,7 pas de temps pour sc3.2).

Ces durées moyennes de communication pour la solution optimale sont présentées dans le tableau 6.3 en nombre de pas de temps. Ce tableau donne également, pour chaque jeu de données, les durées minimale et maximale de communication par *sol* et par sonde.

On remarque que les orbites testées permettent d'obtenir, au mieux, 40 minutes de communication par *sol* et par sonde (durée maximale de 20 pas de temps pour sc1.2, sc 3.1 et sc 3.2) et que seules les orbites des scénarios 1 et 3 permettent d'assurer, à condition de prendre en compte les visibilités de nuit, au moins une communication par *sol* et par sonde (avec une durée minimale de 3 pas de temps).

Le scénario 2 semble particulièrement mal adapté puisqu'il a les plus petites durées moyennes de communication, qu'il ne permet qu'au plus une durée maximale de communication par *sol* et par sonde de 22 minutes et qu'il ne garantit en aucun cas d'avoir une communication par *sol* et par sonde.

Afin d'étudier la possibilité de trouver des solutions de bonne qualité en termes de temps total de communication et qui permettent d'assurer un minimum non nul de communication par *sol* et par sonde, on modifie le critère d'optimisation du modèle linéaire en prenant la fonction de maximisation :

$$Max\ (\min_{s,j}(\sum_{t=\tau_j}^{\tau_{j+1}-1} X_{s,t}))$$

où $j = 1..N_{sol}$ avec N_{sol} le nombre de *sols* dans l'horizon T et τ_j est la date de début du jour j de T.

Ce nouveau critère d'optimisation consiste donc à maximiser la durée minimale de communication par *sol* et par sonde. L'idée est de résoudre le problème en deux temps, en cherchant d'abord une solution maximisant la durée minimale de communication par *sol* et par sonde,

puis en incluant cette valeur comme borne inférieure de durée de communication par *sol* et par sonde pour rechercher une solution qui maximise le temps total de communication.

Malheureusement, les tests effectués sur les six jeux de données avec le nouveau critère d'optimisation, n'améliorent pas les durées minimales de communication par *sol* et par sonde (les solutions obtenues sont exactement égales aux valeurs de la première colonne du tableau 6.3). Ceci montre que, pour les orbites proposées dans les trois scénarios de test, il n'existe pas de solution permettant d'assurer au moins une communication par *sol* et par sonde, si on ne prend en compte que les visibilités de jour. Par ailleurs, si on autorise les communications de nuit, seuls les scénarios 1 et 3 permettent d'assurer au moins une communication par *sol* et par sonde.

jeu de données	durée min. de comm. (/*sol* et /sonde)	durée moy. de comm. (/*sol* et /sonde)	durée max. de comm. (/*sol* et /sonde)
sc 1.1	0	5,9	12
sc 1.2	3	14,6	20
sc 2.1	0	2,2	5
sc 2.2	0	5,5	11
sc 3.1	0	6,2	20
sc 3.2	3	14,7	20

TAB. 6.3 – *Synthèse des durées de communication par sol et par sonde pour la solution optimale*

6.2.3 Généralisation du problème

Le système Netlander est conçu pour fonctionner avec quatre sondes et un unique orbiteur, mais il est probable qu'il y ait un ou plusieurs autres satellites en orbite autour de Mars lors de la mission et que l'on puisse également les utiliser pour relayer les communications entre la Terre et les sondes martiennes. Il est donc intéressant d'étudier une généralisation du problème de planification des communications au cas "multi-orbiteurs".

Nous proposons une adaptation simple de notre modèle linéaire en ajoutant un indice à la variable de décision $X_{s,t}$: soit N_{orb} le nombre d'orbiteurs, on pose, pour tout $o = 1$ à N_{orb}, $X_{o,s,t}$ égale à 1 si, au pas de temps t, la sonde s communique avec l'orbiteur o et égale à 0 sinon.

De nouvelles contraintes viennent s'ajouter au modèle initial, concernant la capacité de communication des sondes. En effet, chaque sonde ne possédant qu'une antenne, elle communique avec au plus un orbiteur à la fois. Ceci se traduit par l'inégalité :

$$\sum_{o} X_{o,s,t} \leq 1 \qquad t=1..T,\ s=1..N \qquad (37)$$

Les contraintes (13) à (15) et (17) à (19) du modèle initial (voir page 76) sont déclinées pour chaque orbiteur et on obtient ainsi le modèle linéaire suivant :

$$
\begin{cases}
Max \quad \sum_{o,s,t} X_{o,s,t} & & (21_{bis}) \\
\text{s.c.} & & \\
\quad X_{o,s,t} = 0 & s{=}1..N,\ o{=}1..N_{orb},\ t{\in}I_{s,o} & (13_{bis}) \\
\quad \sum_{s} X_{o,s,t} \leq 1 & t{=}1..T,\ o{=}1..N_{orb} & (14_{bis}) \\
\quad \sum_{o} X_{o,s,t} \leq 1 & t{=}1..T,\ s{=}1..N & (37) \\
\quad \sum_{t'=t+1}^{t+dmin} X_{o,s,t'} \geq dmin.(X_{o,s,t+1} - X_{o,s,t}) & t{=}1..(T{-}dmin),\ s{=}1..N,\ o{=}1..N_{orb} & (15_{bis}) \\
\quad \sum_{o} X_{o,s,t} \leq \sum_{o} X_{o,s,t-1} & t{=}(T{-}dmin{+}1)..T,\ s{=}1..N & (16_{bis}) \\
\quad \sum_{t'=t+1}^{t+dr-1} X_{o,s,t'} \leq dr_o.(1 - X_{o,s,t-1} + X_{o,s,t}) & t{=}2..(T{-}dr{+}1),\ s{=}1..N,\ o{=}1..N_{orb} & (17_{bis}) \\
\quad \sum_{t=j_i}^{j_{i+1}} \sum_{s} X_{o,s,t} \leq Dmax1_o & i{=}1..J{-}1,\ o{=}1..N_{orb} & (18_{bis}) \\
\quad \sum_{t=l_i}^{l_{i+1}} \sum_{s} X_{o,s,t} \leq Dmax2_o(l_i,l_{i+1}) & i{=}1..L{-}1,\ o{=}1..N_{orb} & (19_{bis}) \\
\quad \sum_{t=k_i}^{k_{i+1}} \sum_{o} X_{o,s,t} \leq Dmax_s(k_i,k_{i+1}) & i{=}1..K_s{-}1,\ s{=}1..N & (20_{bis})
\end{cases}
$$

où les paramètres dr_o, $Dmax1_o$ et $Dmax2_o$, pour chaque orbiteur ($o = 1..N_{orb}$), remplacent respectivement les paramètres dr, $Dmax1_{orb}$ et $Dmax2_{orb}$ du modèle initial. Ainsi, dr_o est la durée de reconfiguration de l'orbiteur o entre deux communications successives, $Dmax1_o$ traduit la capacité du buffer de l'orbiteur o et $Dmax2_o(l_i,l_{i+1})$ traduit l'énergie électrique de l'orbiteur o, sur l'intervalle $[l_i,l_{i+1}]$ (pour $i = 1..L$). Les autres paramètres et indices utilisés dans ce modèle sont présentés page 75.

Les contraintes (13_{bis}) à (20_{bis}), ainsi que le critère de maximisation (21_{bis}) ont respectivement la même signification, dans un contexte multi-orbiteurs, que les contraintes (13) à (20) et que le critère (21) du modèle initial. Nous renvoyons donc aux détails donnés page 76 pour leur explication.

Nous avons testé la résolution par programmation linéaire de ce modèle pour des instances à deux et trois orbiteurs en utilisant les jeux de données (avec visibilités de jour uniquement) fournis par le CNES et en autorisant un temps maximum d'exécution de 4 minutes. Le tableau 6.4 indique pour chaque couple de scénarios, puis pour les trois scénarios ensemble, le temps d'exécution du programme et la solution obtenue (en nombre de pas de temps) pour différents horizons de planification : d'une part pour l'horizon maximal résolu, d'autre part, pour quatre horizons successifs (de 5 400 pas de temps chacun) qui couvrent l'horizon total de planification.

L'horizon temporel que l'on peut traiter est naturellement d'autant plus petit que le nombre d'orbiteurs considéré est élevé. Pour exemple, nous obtenons – dans un délai de trois minutes et demi – une solution optimale pour des problèmes à deux satellites sur un horizon de dix-neuf jours, et à trois satellites sur un horizon de huit jours et demi. Pour des problèmes à plus de trois orbiteurs, nous manquons de jeux de données ; notons que nous avons réalisé des tests jusqu'à cinq satellites en considérant plusieurs fois le même scénario. On résout de tels problèmes sur un horizon de cinq jours mais ces instances ne sont pas représentatives de cas réels puisqu'on multiplie artificiellement les mêmes intervalles de visibilité. Notons enfin que les différents scénarios présentés dans le tableau 6.4 ne permettent toujours pas d'assurer une durée de communication de 40 minutes par *sol* et par sonde.

	horizon maximal résolu			horizons de 7 jours 1/2							
				0 – 5400		5400 – 10800		10800 – 16200		16200 – 21600	
	horizon	CPU	solut°	CPU	solut°	CPU	solut°	CPU	solut°	CPU	solut°
sc1.1 +sc2.1	0 – 13331 (18 jours 1/2)	205	756	53	249	47	235	48	221	46	230
sc1.1 +sc3.1	0 – 13680 (19 jours)	207	524	53	222	53	220	53	202	52	204
sc2.1 +sc3.1	0 – 13680 (19 jours)	205	589	47	241	47	235	47	221	47	200
sc1.1 +sc2.1 +sc3.1	0 – 6075 (8 jours 1/2)	181	291	157	284	156	276	156	257	144	259

TAB. 6.4 – *Résultats obtenus par programmation linéaire pour le problème généralisé*

6.2.4 Conclusion

Le problème de planification des communications orbiteur/sondes dans le projet Netlander, bien qu'en variables dicrètes, est peu combinatoire et reste dans le domaine des problèmes traitables par une méthode exacte classique et par des outils commerciaux comme CPLEX.

Ceci est principalement dû à la nature des données traitées. En effet, les sondes étant suffisamment éloignées les unes des autres, les fenêtres de visibilité de l'orbiteur avec les différentes sondes se chevauchent relativement peu entre elles, ce qui limite les conflits de partage de l'orbiteur. De plus, du point de vue d'une sonde, l'orbiteur est la plupart du temps indisponible, ce qui fixe un grand nombre de variables à zéro.

Dans le cas généralisé où l'on considère plusieurs orbiteurs, on a naturellement plus de conflits à gérer, mais on voit que l'on obtient encore des résultats satisfaisants par Programmation Linéaire en Nombres Entiers. On pourrait poursuivre l'étude de ce problème, pour le résoudre sur de plus grands horizons temporels par exemple, en développant une approche par décomposition du modèle, suivant les différentes orbites (décomposition de Benders par exemple, avec une résolution des problèmes liés à chaque orbite par PLNE et gestion des contraintes couplantes (37) à un niveau supérieur de coordination).

Dans la suite de ce chapitre, nous traitons le problème de planification des expériences d'une sonde dans le projet Netlander.

6.3 Planification des expériences

Rappelons qu'une expérience élémentaire i est caractérisée par les attributs suivants :

- un mode opératoire mop_i,
- un numéro de sonde s_i,
- une fenêtre de réalisation $[est,lft]_i$,
- une durée minimale $\underline{du_i}$,
- une durée maximale $\overline{du_i}$.

On associe à chaque expérience élémentaire i deux variables st_i et ft_i réprésentant les dates de début et de fin d'exécution de i (si i est exécutée, $st_i,ft_i \in [est,lft]_i$ et $\underline{du_i} \leq ft_i - st_i \leq \overline{du_i}$). La figure 6.1 donne une représentation dans le temps d'une expérience élémentaire.

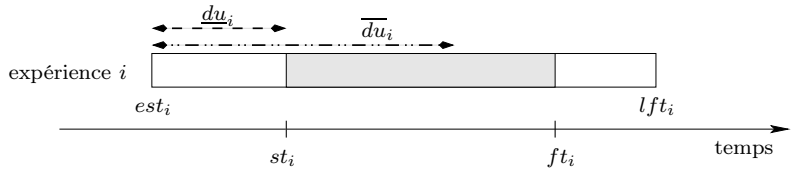


FIG. 6.1 – *Représentation d'une expérience élémentaire*

Comme on considère le problème de planification des créneaux de communication prioritaire sur celui de la planification des expériences, on suppose que l'on connaît déjà les créneaux de communication des sondes sur chaque horizon de planification.

Étant donné un horizon T, les créneaux de communication d'une sonde s sur T, ainsi qu'un ensemble d'expériences élémentaires candidates $\mathcal{E}c$ sur s, le problème de planification des expériences dans le projet Netlander consiste à élaborer un plan de travail réalisable pour s sur T. Il s'agit donc de sélectionner un sous-ensemble d'expériences élémentaires $\mathcal{E} \subseteq \mathcal{E}c$ et à les ordonnancer sur T (instancier les variables st_i et ft_i pour tout $i \in \mathcal{E}$), de façon à satisfaire des contraintes liées au partage de la mémoire de masse et de l'énergie électrique de la sonde s.

On se place dans un contexte d'aide à la décision. Afin d'aider un responsable de mission, ou un ensemble d'experts scientifiques au cours d'une négociation, nous proposons, pour chacune des deux ressources considérées, de fournir des courbes d'évaluation de la charge au cours de l'élaboration du plan. Nous faisons appel à des concepts de raisonnement énergétique, introduit au chapitre 2, pour établir et analyser ces courbes.

Un plan de travail *en cours d'élaboration* est caractérisé par un sous-ensemble d'expériences sélectionnées $\mathcal{E}$ de $\mathcal{E}c$ et par l'état des variables st_i et ft_i pour tout $i \in \mathcal{E}$. On a $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2$ où $\mathcal{E}_1$ est l'ensemble des expériences élémentaires sélectionnées i dont les dates de début et fin d'exécution st_i et ft_i sont instanciées et où $\mathcal{E}_2$ est l'ensemble des expériences élémentaires sélectionnées dont les dates de début et fin d'exécution st et ft ne sont pas encore instanciées ($\mathcal{E}_1 \cap \mathcal{E}_2 = \emptyset$). Si $\mathcal{E}_2 = \emptyset$ ($\mathcal{E} = \mathcal{E}_1$) et si les contraintes de partage de la mémoire de masse et de l'énergie électrique sont satisfaites, alors $\mathcal{E}$ fournit un plan de travail réalisable pour la sonde s, sur l'horizon T.

6.3.1 Courbe de charge de la mémoire de masse d'une sonde

6.3.1.1 La mémoire de masse d'une sonde

La mémoire de masse d'une sonde a une taille limitée mais elle est réutilisable, c'est-à-dire que les données stockées peuvent être écrasées et remplacées par de nouvelles données.

Les règles permettant d'écraser les données et donc de libérer la ressource mémoire ne sont pas clairement définies. Différentes hypothèses ont été envisagées.

On peut tout d'abord considérer que la mémoire de masse fonctionne en boucle : lorsque toute la place mémoire est occupée, les plus anciennes données sont écrasées. Ce mode de fonctionnement permet de ne jamais interdire la réalisation d'une expérience à cause de la place mémoire qu'elle nécessite mais cela entraîne le risque de perdre des résultats d'expériences avant de les avoir rapatriés au sol.

À l'inverse, on peut supposer que la mémoire n'est libérée que sur commande ; c'est-à-dire,

qu'après avoir reçu les données au sol, on envoie dans un prochain plan de travail l'ordre de les écraser aux sondes correspondantes. On assure ainsi qu'il n'y a pas de perte de données, même en cas d'échec d'une communication, mais ces précautions forcent un stockage prolongé de tous les résultats d'expériences et limitent ainsi considérablement la quantité de données différentes que l'on pourra stocker sur un horizon T et donc le nombre d'expériences qui pourront être exécutées sur cet horizon.

On peut également envisager que les données sont effacées au terme d'une période fixée, que l'on suppose suffisante pour avoir le temps, en général, de rapatrier toutes les données au sol. La pertinence de cette hypothèse dépend largement de la qualité de l'estimation de la durée nécessaire de conservation des données. De plus, même si on connaît une estimation fiable en moyenne (ce qui n'est pas le cas actuellement), la prise en compte d'une durée fixe de conservation peut entraîner à la fois l'écrasement de données non encore rapatriées et une sous-utilisation de la mémoire de masse, suivant le contexte de la mission.

Enfin, on peut considérer que chaque transfert de données réalisé lors d'une communication sonde/orbiteur est réussi, c'est-à-dire qu'à la fin de chaque communication sonde/orbiteur, la mémoire de masse de la sonde peut être libérée de toutes les données qui ont été transférées lors de cette communication. C'est cette dernière hypothèse que nous retenons dans la suite car elle offre un compromis satisfaisant entre le risque d'écraser des données non rapatriées au sol (uniquement dans le cas où une communication sonde/orbiteur échoue) et celui de limiter inutilement la réalisation d'expériences (une donnée n'est jamais forcée à être conservée alors qu'elle a déjà été reçue au sol).

Puisque l'on suppose connaître les créneaux de communication sonde/orbiteur, on peut évaluer précisément le processus de renouvellement de la mémoire de masse de chaque sonde. C'est une ressource doublement contrainte au sens de [Blazewicz *et al.* 01]. En effet, elle est à la fois renouvelable (partiellement, après chaque communication sonde/orbiteur) et consommable (entre deux communications sonde/orbiteur, on dispose d'une quantité limitée de mémoire) (voir paragraphe 1.2.4, chapitre 1).

6.3.1.2 Élaboration de la courbe de charge de la mémoire de masse

Soit T un horizon de planification, $\{[sc_j,fc_j]_{s,T} \mid j = 1..Ncom_{s,T}, s = 1..4\}$ l'ensemble des intervalles de communication sonde/orbiteur sur T destinés à la transmission des données à l'orbiteur (avec $Ncom_{s,T}$ le nombre de créneaux de communication de la sonde s sur T).

La taille totale, $MdM_{s,T}$, de la mémoire de masse est supposée constante et connue pour chaque sonde s et chaque horizon T (elle peut diminuer au cours de la mission en fonction des aléas et dégradations que subit la sonde).

Le débit de transfert de données d'une sonde à l'orbiteur est constant ; on le note δ_{com}. On peut donc évaluer la place mémoire disponible $M_{j,s,T}$ sur la sonde s, à la fin de chaque communication sonde/orbiteur $[sc_j,fc_j]_{s,T}$. Dans la suite, pour simplifier les notations, on omet les indices s et T (il n'y a pas d'ambiguïté puisque l'on s'intéresse à une unique sonde sur un unique horizon).

On peut donc exprimer la place mémoire M_j, disponible sur une sonde à la fin du créneau de communication $[sc_j,fc_j]$ par l'équation suivante :

$$M_j = \min(MdM\ ,\ M_{j-1} - data(fc_{j-1},fc_j) + \delta_{com} \times (fc_j - sc_j))$$

où $data(fc_{j-1},fc_j)$ est la place mémoire nécessaire pour enregistrer les données émises par les expériences entre les dates fc_{j-1} et fc_j. Notons que $fc_0 = 0$ et M_0 est la place mémoire disponible à la date 0 de l'horizon T ($M_0 \leq MdM$).

La figure 6.2 illustre le processus de renouvellement de la mémoire de masse d'une sonde s sur un horizon T. Les zones hachurées représentent les créneaux de communication entre la sonde et l'orbiteur. La place mémoire disponible à la fin de l'horizon T, M_T, est la valeur initiale de la mémoire de masse disponible pour l'horizon de planification suivant.

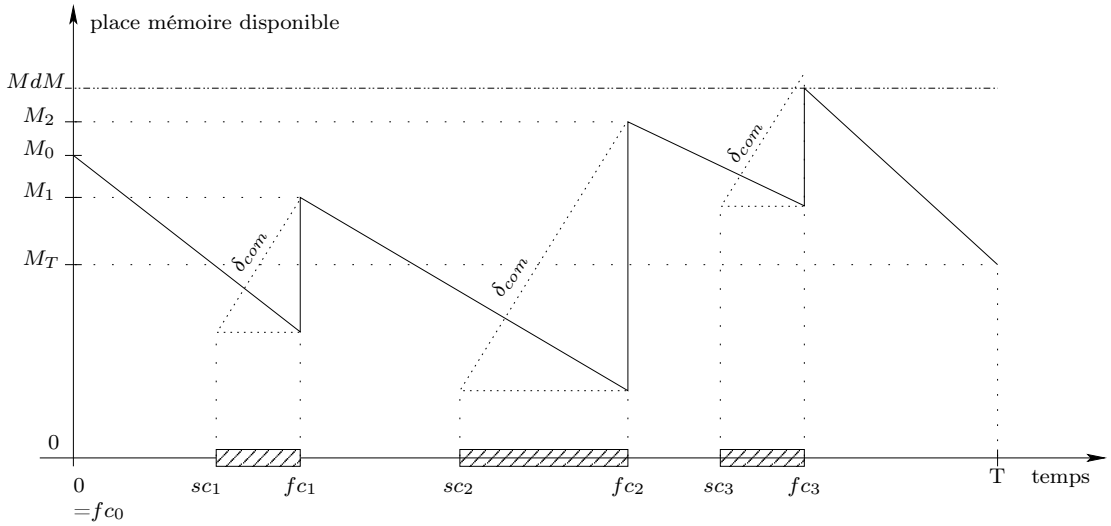


FIG. 6.2 – *Représentation schématique du renouvellement de la mémoire de masse d'une sonde sur un horizon T*

Soit un plan de travail en cours d'élaboration pour la sonde s, sur l'horizon T et $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2$ l'ensemble associé des expériences élémentaires sélectionnées.

Pour tout intervalle $[t,t'] \subseteq T$, on peut écrire $data(t,t')$ comme la somme de $data_{\mathcal{E}_1}(t,t')$ et $data_{\mathcal{E}_2}(t,t')$ où $data_{\mathcal{E}_1}(t,t')$ est la place mémoire requise pour l'exécution des expériences de $\mathcal{E}_1$ dans l'intervalle de temps $[t,t']$ et $data_{\mathcal{E}_2}(t,t')$ est la place mémoire requise pour l'exécution des expériences de $\mathcal{E}_2$ dans l'intervalle de temps $[t,t']$.

$data_{\mathcal{E}_1}(t,t')$ peut être calculé puisque pour toute expérience i de $\mathcal{E}_1$, st_i et ft_i ont une valeur fixée. On a donc :

$$data_{\mathcal{E}_1}(t,t') = \sum_{i \in \mathcal{E}_1 \mid [t,t'] \cap [st_i,ft_i] \neq \emptyset} data_i(t,t')$$

où

$$data_i(t,t') = (\min(t',ft_i) - \max(t,st_i)) \times \delta_{mop_i}$$

est la place mémoire requise pour l'exécution de i sur l'intervalle $[t,t']$ (rappelons que δ_{mop_i} est le débit de données du mode opératoire associé à l'expérience i). La figure 6.3 montre les intervalles temporels mis en jeu pour le calcul de $data_i(t,t')$ pour quatre expériences élémentaires appartenant à $\mathcal{E}_1$.

En revanche, puisque $\mathcal{E}_2$ est l'ensemble des expériences sélectionnées dont les dates de

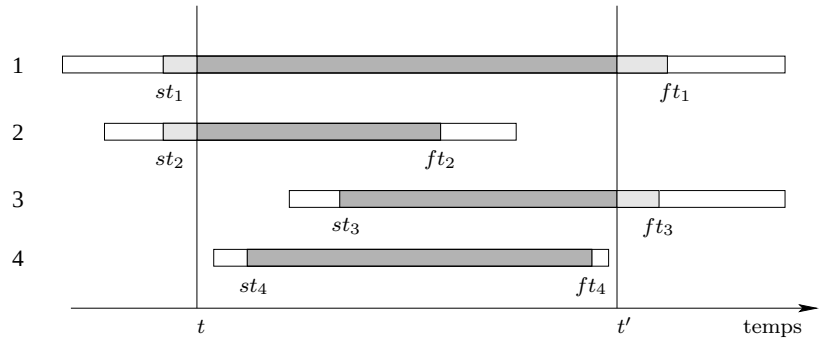


FIG. 6.3 – *Intervalles de réalisation de quatre expériences élémentaires de $\mathcal{E}_1$ sur $[t,t']$*

début et de fin d'exécution ne sont pas encore instanciées, on ne peut pas définir exactement $data_{\mathcal{E}_2}(t,t')$.

Nous proposons une borne inférieure, $\underline{data}_{\mathcal{E}_2}(t,t')$, et une borne supérieure, $\overline{data}_{\mathcal{E}_2}(t,t')$, de $data_{\mathcal{E}_2}(t,t')$. $\underline{data}_{\mathcal{E}_2}(t,t')$ est la place mémoire *obligatoire* requise pour $\mathcal{E}_2$ sur l'intervalle $[t,t']$ et $\overline{data}_{\mathcal{E}_2}(t,t')$ est la place mémoire *maximale* requise pour $\mathcal{E}_2$ sur sur l'intervalle $[t,t']$.

On a :

$$\underline{data}_{\mathcal{E}_2}(t,t') = \sum_{\substack{i\in\mathcal{E}_2 \mid est_i+\underline{du}_i>t \\ \text{et } lft_i-\underline{du}_i<t'}} \underline{data}_i(t,t')$$

et

$$\overline{data}_{\mathcal{E}_2}(t,t') = \sum_{i\in\mathcal{E}_2 \mid [t,t']\cap[st_i,ft_i]\neq\emptyset} \overline{data}_i(t,t')$$

où

$$\underline{data}_i(t,t') = \min(est_i + \underline{du}_i - t,\, t' - lft_i + \underline{du}_i,\, t' - t,\, \underline{du}_i) \times \delta_{mop_i}$$

est la place mémoire obligatoire requise pour l'exécution de i sur l'intervalle $[t,t']$ et

$$\overline{data}_i(t,t') = \min(lft_i - t\,,\, t' - est_i\,,\, t' - t\,,\, \overline{du}_i) \times \delta_{mop_i}$$

est la place mémoire maximale requise pour l'exécution de i sur l'intervalle $[t,t']$. On retrouve ici des formulations du type consommation énergétique vues en 3.2, l'énergie consommée étant de la place mémoire. Les figures 6.4 et 6.5 montrent les intervalles temporels considérés pour le calcul de $\underline{data}_i(t,t')$ et $\overline{data}_i(t,t')$ pour quatre expériences élémentaires appartenant à $\mathcal{E}_2$. Ce sont les intervalles de "consommation de mémoire de masse" obligatoire (figure 6.4) et maximale (figure 6.5) sur l'intervalle $[t,t']$.

On obtient ainsi un encadrement de la quantité de données générées par les expériences de $\mathcal{E}$ sur l'intervalle $[t,t']$:

$$data_{\mathcal{E}_1}(t,t') + \underline{data}_{\mathcal{E}_2}(t,t') \leq\ data(t,t')\ \leq data_{\mathcal{E}_1}(t,t') + \overline{data}_{\mathcal{E}_2}(t,t')$$

La place mémoire M_j disponible sur la sonde à la fin de chaque créneau de communication $[sc_j,fc_j]$ pour $j = 1$ à $Ncom$ peut donc être encadrée comme suit :

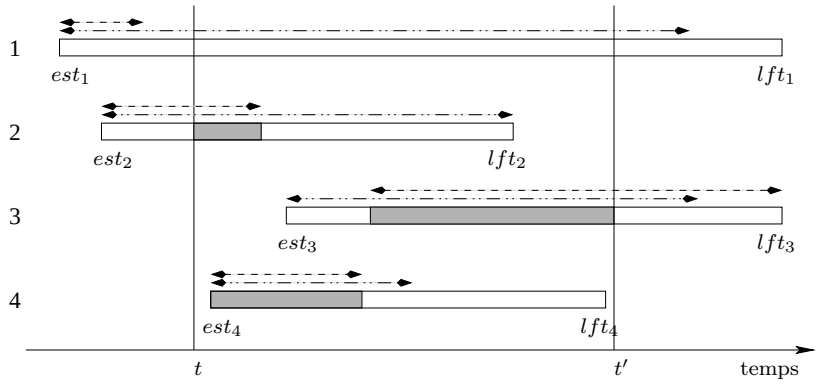


FIG. 6.4 – *Intervalles de consommation de mémoire de masse obligatoire de quatre expériences élémentaires de* $\mathcal{E}_2$ *sur* $[t,t']$

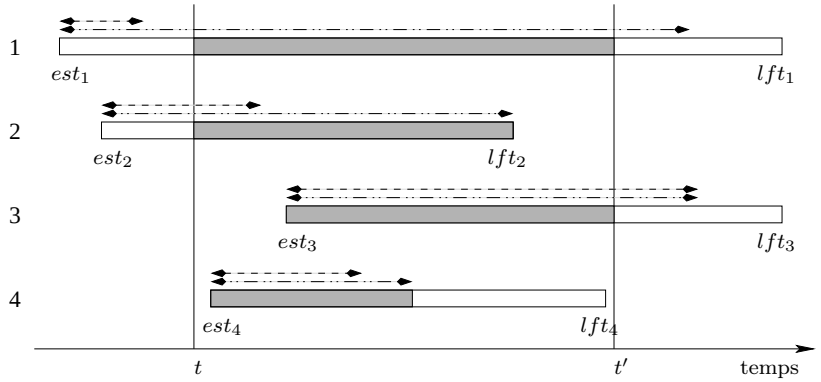


FIG. 6.5 – *Intervalles de consommation de mémoire de masse maximale de quatre expériences élémentaires de* $\mathcal{E}_2$ *sur* $[t,t']$

$$\underline{M}_j \leq M_j \leq \overline{M}_j$$

avec

$$\underline{M}_j = \min(MdM\ ,\ M_{j-1} - data_{\mathcal{E}_1}(fc_{j-1},fc_j) - \overline{data}_{\mathcal{E}_2}(fc_{j-1},fc_j) + \delta_{com} \times (fc_j - sc_j))$$

et

$$\overline{M}_j = \min(MdM\ ,\ M_{j-1} - data_{\mathcal{E}_1}(fc_{j-1},fc_j) - \underline{data}_{\mathcal{E}_2}(fc_{j-1},fc_j) + \delta_{com} \times (fc_j - sc_j))$$

On établit deux courbes, $\mathcal{C}_{sup}$ et $\mathcal{C}_{inf}$, représentant d'une part, une évolution "optimiste" de la place disponible sur la mémoire de masse au cours du temps (courbe de *charge obligatoire* de la mémoire de masse) et d'autre part, une évolution "pessimiste" de cette place mémoire disponible (courbe de *charge maximale*).

Pour construire ces courbes, on pose les règles suivantes pour chaque intervalle $[fc_{j-1},fc_j]$ $(j = 1$ à $Ncom)$:

– pour toute expérience $i \in \mathcal{E}_1$, telle que $[fc_{j-1},fc_j] \cap [st_i,ft_i] \neq \emptyset$, on considère que la place mémoire requise $data_i(fc_{j-1},fc_j)$ est occupée à la date $\tau_i = \max(st_i,fc_{j-1})$;

- pour toute expérience $i \in \mathcal{E}_2$, telle que $[fc_{j-1},fc_j] \cap [est_i,lft_i] \neq \emptyset$, on considère que la place mémoire maximale requise $\overline{data_i}(fc_{j-1},fc_j)$ est occupée à la date $\overline{\tau}_i = \max(est_i,fc_{j-1})$ (pour la courbe $\mathcal{C}_{inf}$);
- pour toute expérience $i \in \mathcal{E}_2$, telle que $est_i + \underline{du}_i > fc_{i-1}$ et $lft_i - \underline{du}_i < fc_i$, on considère que la place mémoire obligatoire requise $\underline{data}_i(fc_{j-1},fc_j)$ est occupée à la date $\underline{\tau}_i = \max(lft_i - \underline{du}_i,fc_{j-1})$ (pour la courbe $\mathcal{C}_{sup}$);
- la mémoire de masse est partiellement renouvelée à la date fc_j ; la place disponible est alors $\underline{M}_j$ pour la courbe $\mathcal{C}_{inf}$ et $\overline{M}_j$ pour la courbe $\mathcal{C}_{sup}$.

La figure 6.6 met en œuvre ces différentes règles pour la construction des courbes $\mathcal{C}_{inf}$ et $\mathcal{C}_{sup}$ sur un intervalle $[fc_1,fc_2]$. Le plan de travail en cours d'élaboration considéré est constitué de quatre expériences : les expériences 1, 2 et 3 appartiennent à $\mathcal{E}_2$ et l'expérience 4 appartient à $\mathcal{E}_1$.

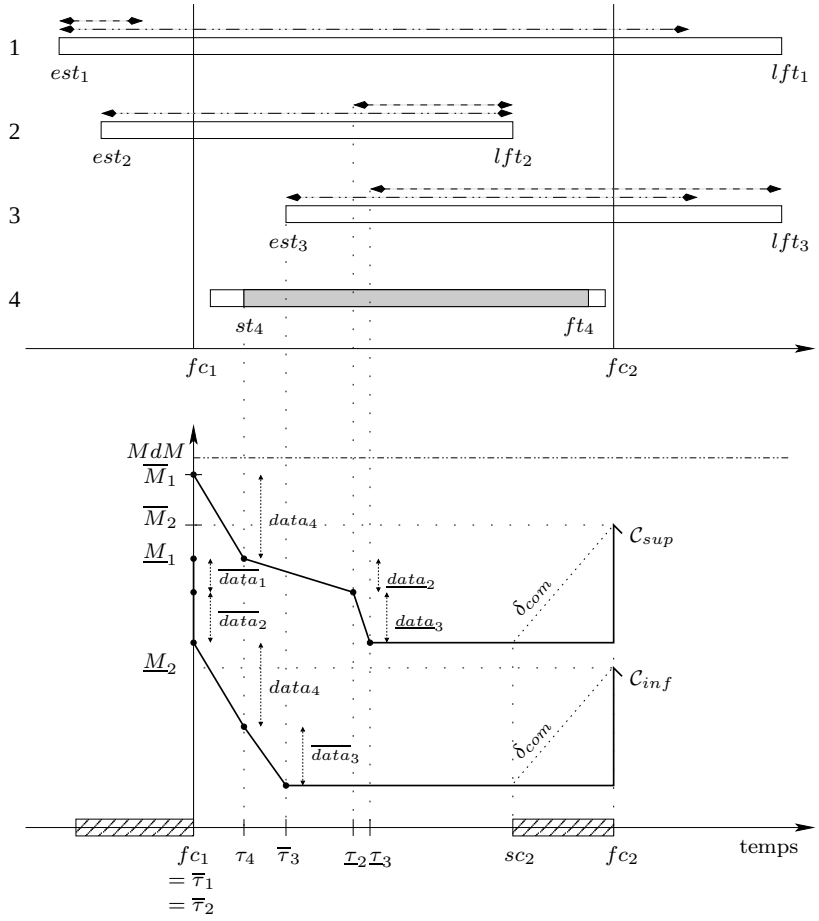


FIG. 6.6 – *Exemple de construction des courbes $\mathcal{C}_{inf}$ et $\mathcal{C}_{sup}$ sur un intervalle $[fc_1,fc_2]$ pour un ensemble de quatre expériences élémentaires*

Lorsque le plan de travail est figé (toutes les dates de début et fin d'exécution sont instanciées), les deux courbes se confondent et représentent l'évolution de la charge de la mémoire de masse pour ce plan de travail.

6.3.1.3 Analyse des courbes, règles de déduction

Sur chaque intervalle $[fc_{j-1},fc_j]$, la mémoire de masse est une ressource consommable, de taille M_{j-1}.

Afin de donner des indications sur la faisabilité (relativement à la contrainte de partage de la mémoire de masse) d'un plan de travail au cours de son élaboration, on utilise les règles de déduction suivantes, issues du raisonnement énergétique :

- S'il existe j tel que $\overline{M}_{j-1} < data_{\mathcal{E}_1}(fc_{j-1},fc_j) + \underline{data}_{\mathcal{E}_2}(fc_{j-1},fc_j)$ (la courbe $\mathcal{C}_{sup}$ passe sous l'axe des abscisses sur l'intervalle $[fc_{j-1},fc_j]$), le plan de travail ne peut pas être réalisable. En effet, dans cette évaluation optimiste de la place mémoire disponible, la place mémoire requise est supérieure à la place mémoire disponible sur cet intervalle ; on a donc nécessairement un viol de la contrainte de partage de la mémoire.
- À l'inverse, si pour tout $j = 1$ à N_{com}, $\underline{M}_{j-1} > data_{\mathcal{E}_1}(fc_{j-1},fc_j) + \overline{data}_{\mathcal{E}_2}(fc_{j-1},fc_j)$ (la courbe $\mathcal{C}_{inf}$ reste au-dessus de l'axe des abscisses sur tout l'horizon T), le plan de travail est réalisable quelles que soient les dates de début et fin d'exécution choisies pour les expériences de $\mathcal{E}_2$.
- S'il existe j tel que $\overline{M}_{j-1} < data_{\mathcal{E}_1}(fc_{j-1},fc_j)$, alors le plan de travail restreint aux expériences de $\mathcal{E}_1$ n'est pas réalisable. Il est donc nécessaire de modifier certaines décisions prises sur les expériences de $\mathcal{E}_1$ dans les intervalles $[fc_0,fc_1]$ à $[fc_{j-1},fc_j]$.
- S'il existe j tel que $\overline{M}_{j-1} < \underline{data}_{\mathcal{E}_2}(fc_{j-1},fc_j)$, alors le plan de travail restreint aux expériences de $\mathcal{E}_2$ n'est pas réalisable. Il est donc nécessaire de modifier certaines décisions prises sur les expériences de $\mathcal{E}_1$ dans les intervalles précédents, ou de supprimer certaines des expériences i de $\mathcal{E}$ telles que $est_i \leq fc_j$.
- Pour toute expérience $i \in \mathcal{E}_1$, soit
  $$\overline{M}^i_{j-1} = \overline{M}_{j-1} - \textstyle\sum_{e\in\mathcal{E}_1\setminus\{i\}} data_e(fc_{j-1},fc_j) - \underline{data}_{\mathcal{E}_2}(fc_{j-1},fc_j)$$
  la place mémoire disponible maximale pour l'exécution de i sur $[fc_{j-1},fc_j]$; s'il existe j tel que $0 < \overline{M}^i_{j-1} < data_i(fc_{j-1},fc_j)$, alors on ne peut pas exécuter à la fois i et l'ensemble d'expériences $\mathcal{E}_1 \setminus \{i\} \cup \mathcal{E}_2$. Si on peut décaler l'exécution de i, ou réduire sa durée de réalisation de telle façon que $[st_i,ft_i]\cap[fc_{j-1},fc_j]$ soit un intervalle de longueur inférieure à $\overline{M}^i_{j-1}/\delta_{mop_i}$, le plan devient alors réalisable sur cet intervalle. Sinon, il faut modifier certaines décisions prises sur les expériences de $\mathcal{E}_1$ qui débutent avant fc_j, ou bien supprimer certaines des expériences e de $\mathcal{E}$ telles que $est_e \leq fc_j$.
- Pour toute expérience $i \in \mathcal{E}_2$, soit
  $$\overline{M}^i_{j-1} = \overline{M}_{j-1} - data_{\mathcal{E}_1}(fc_{j-1},fc_j) - \textstyle\sum_{e\in\mathcal{E}_2\setminus\{i\}} \underline{data}_e(fc_{j-1},fc_j)$$
  la place mémoire disponible maximale pour l'exécution de i sur $[fc_{j-1},fc_j]$; s'il existe j tel que $0 < \overline{M}^i_{j-1} < \underline{data}_i(fc_{j-1},fc_j)$, alors on ne peut pas exécuter à la fois i et l'ensemble d'expériences $\mathcal{E}_1\cup\mathcal{E}_2\setminus\{i\}$. On doit donc modifier certaines décisions prises sur les expériences de $\mathcal{E}_1$ qui débutent avant fc_j, ou bien supprimer certaines des expériences e de $\mathcal{E}$ telles que $est_e \leq fc_j$.

À chaque étape de l'élaboration d'un plan de travail, on peut ainsi donner à l'utilisateur des informations sur sa faisabilité et éventuellement proposer un ensemble de modifications permettant de converger vers une solution réalisable.

6.3.2 Courbe de consommation de l'énergie électrique d'une sonde

6.3.2.1 L'énergie électrique d'une sonde

Le système d'alimentation électrique des sondes comporte des panneaux solaires et des batteries rechargeables. Le mode de consommation de la ressource électrique dépend ainsi du moment de la journée considéré :

- pendant la période d'ensoleillement des sondes (de l'aube au crépuscule), de l'énergie électrique est produite au niveau des panneaux solaires ; cette énergie est utilisée pour recharger la batterie d'une part et pour réaliser les différentes tâches consommatrices d'électricité d'autre part (principalement les expériences et les communications). Au besoin, on peut également puiser sur la batterie pendant le jour ;
- du crépuscule à l'aube, les panneaux solaires étant dans l'ombre, on ne peut utiliser que l'énergie électrique fournie par la batterie.

Comme la mémoire de masse, l'énergie électrique fournie par la batterie peut être vue comme une ressource doublement contrainte ; elle est en effet consommable et renouvelable à certains moments d'une journée. Cependant, son renouvellement est plus compliqué que celui de la mémoire de masse car il peut se faire pendant toute la période d'ensoleillement d'une journée et non à certains instants précis ; de plus, il requiert lui-même une certaine énergie électrique puisque c'est grâce à l'électricité fournie par les panneaux solaires que l'on peut recharger la batterie. Cette énergie solaire peut quant à elle être vue comme une ressource différente, uniquement consommable, disponible seulement pendant les périodes d'ensoleillement de la sonde et de capacité non constante.

On peut considérer que le mode d'affectation des différentes tâches à ces deux ressources consiste à utiliser en priorité l'énergie issue des panneaux solaires (E_{sol}) et uniquement en complément l'énergie issue de la batterie (E_{bat}). Des quantités d'énergie déchargées de la batterie, on peut ensuite déduire des tâches fictives correspondant à la recharge de la batterie ; ces tâches sont, elles, nécessairement exécutées par la ressource E_{sol}.

La puissance électrique délivrée au cours du temps par les panneaux solaires dépend principalement des conditions climatiques de Mars ainsi que de l'état de dégradation des panneaux solaires. On considère que l'on peut prévoir ces conditions sur tout horizon de planification T, on peut donc ainsi connaître la puissance délivrée par les panneaux solaires au cours du temps. D'autre part, on suppose connues les consommations électriques liées aux créneaux de communication sonde/orbiteur. On peut donc évaluer la puissance électrique disponible au cours du temps pour l'exécution d'expériences par la sonde.

6.3.2.2 Élaboration de la courbe d'utilisation de l'énergie électrique

Évaluation par intervalles de l'énergie électrique disponible

Le profil type de l'évolution de la puissance électrique produite au cours du temps, $P(t)$, est schématisé à la figure 6.7. De l'aube au zénith du soleil, la production électrique augmente, du zénith au crépuscule elle diminue, puis du crépuscule à l'aube suivante, elle est nulle (l'électricité est alors fournie uniquement par la batterie).

De ces courbes de puissance, on peut déduire des niveaux d'énergie totale disponible par

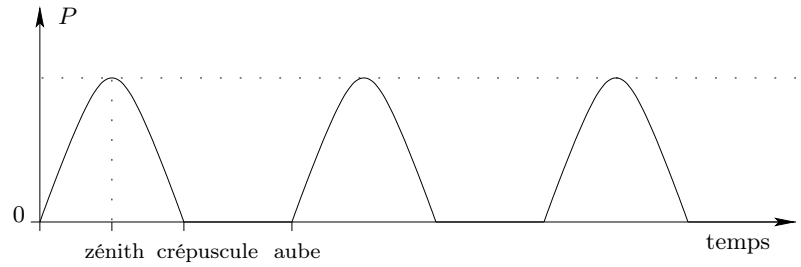


FIG. 6.7 – *Profil de la puissance électrique fournie par les panneaux solaires d'une sonde*

intervalle de temps, $E_{tot}(t,t')$, avec l'égalité :

$$E_{tot}(t,t') = \int_t^{t'} P(\tau).d\tau + E_{bat}$$

où E_{bat} est l'énergie de la batterie (lorsqu'elle est totalement chargée) dans l'intervalle de temps $[t,t']$. Cette évaluation des niveaux d'énergie agrège la ressource électrique par intervalles et permet de considérer qu'elle est de capacité constante sur chacun de ces intervalles. Ceci entraîne évidemment une perte d'information et constitue donc une approximation du comportement de la ressource électrique. Cette approximation est d'autant meilleure que les intervalles considérés pendant les périodes d'ensoleillement sont courts. En revanche, aucune puissance électrique n'étant produite de nuit, la capacité de la ressource électrique est constante sur ces périodes. On peut donc ne considérer qu'un seul intervalle par nuit.

Des niveaux d'énergie totale disponible ainsi obtenus, on retranche ensuite les consommations électriques dues aux communications sonde/orbiteur. Soit $E_{tot}(t,t')$ l'énergie totale disponible sur un intervalle d'agrégation $[t,t']$, soit $[sc_i,fc_i]$ un intervalle de communication sonde/orbiteur tel que $t \leq sc_i$ et $fc_i \leq t'$, l'énergie disponible pour les expériences, $E_d(t,t')$ est alors :

$$E_d(t,t') = E_{tot}(t,t') - P_{com}.(fc_i - sc_i)$$

où P_{com} est la puissance électrique requise pour exécuter une communication sonde/orbiteur.

On obtient ainsi une courbe des niveaux d'énergie disponible par intervalles pour les expériences, sur tout l'horizon T, comme illustré à la figure 6.8.

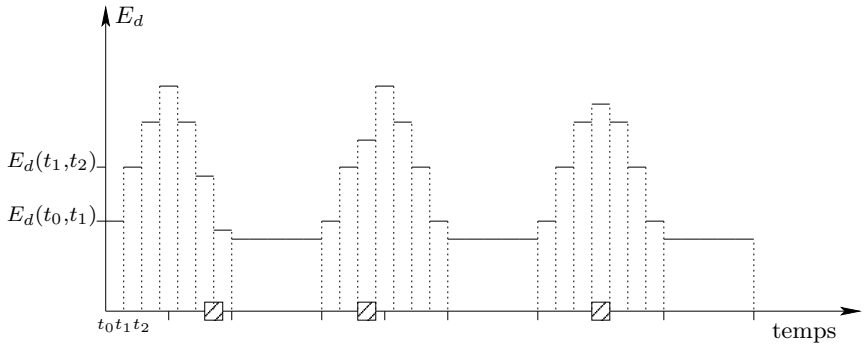


FIG. 6.8 – *Exemple de courbe de niveaux d'énergie disponible par intervalles*

Notons que les communications sonde/orbiteur n'interviennent que pendant les phases d'ensoleillement ; on n'a donc pas à utiliser la batterie pour les réaliser et donc il n'y a pas de phase de recharge de la batterie engendrée par l'exécution d'une communication.

Soit un plan de travail en cours d'élaboration pour la sonde s, sur l'horizon T et $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2$ l'ensemble associé des expériences élémentaires sélectionnées. Soit $I_T = \{[t_{j-1},t_j] \mid j = 1..n\}$ l'ensemble des intervalles sur lesquels on calcule les niveaux d'énergie disponible E_d (on a $t_0 = 0$ et $t_n = T$).

Bilan énergétiques

Sur tout intervalle $[t_{j-1},t_j] \in I_T$, on peut établir le bilan énergétique E_j suivant :

$$E_j = E_d(t_{j-1},t_j) - conso_{\mathcal{E}}(t_{j-1},t_j)$$

où $conso_{\mathcal{E}}(t_{j-1},t_j)$ est l'énergie consommée sur l'intervalle $[t_{j-1},t_j]$ par l'exécution des expériences de $\mathcal{E}$.

Dans la suite, nous notons $E_{d_j} = E_d(t_{j-1},t_j)$.

Si $E_j \geq 0$, E_j représente l'énergie électrique non utilisée sur l'intervalle $[t_{j-1},t_j]$, sinon E_j représente la quantité d'énergie qu'il manque sur l'intervalle $[t_{j-1},t_j]$ pour pouvoir réaliser les expériences de $\mathcal{E}$. De plus, si $E_j \geq E_{bat}$, la batterie n'est pas sollicitée sur $[t_{j-1},t_j]$; au contraire, si $E_j < E_{bat}$, on utilise la batterie sur cet intervalle, il faut alors prévoir une phase de recharge de la batterie dans les intervalles suivants. Cette phase constitue une nouvelle tâche à exécuter, $rech_j$, que l'on assimile à une expérience élémentaire avec les attributs suivants :

- $P_{rech_j} = P_{rech}$, la puissance de recharge de la batterie (supposée constante) ;
- $[est,lft]_{rech_j} = [t_j,tc_j]$, la fenêtre de réalisation, où tc_j est la date du crépuscule qui suit la date t_j (on force ainsi la batterie à se recharger avant la prochaine période de nuit) ;
- $du_{rech_j} = \alpha.(E_{bat} - E_j)/P_{rech}$, la durée d'exécution ($\overline{du}_{rech_j} = \underline{du}_{rech_j} = du_{rech_j}$), avec $\alpha \geq 1$, un certain coefficient multiplicateur.

Cette nouvelle expérience doit être ajoutée à l'ensemble $\mathcal{E}$ pour établir les bilans énergétiques E_k pour $k > j$. Comme on ne connaît pas, *a priori*, ses dates d'exécution (st_{rech_j} et ft_{rech_j} ne sont pas instanciées), $rech_j$ est ajouté à l'ensemble $\mathcal{E}_2 \subseteq \mathcal{E}$.

Décharge de la batterie, mise-à-jour des valeurs de E_d

Lorsque $E_j < E_{bat}$, l'énergie disponible, sur les intervalles $[t_{k-1},t_k]$ ($k \in \{j+1..n\}$) tels que $rech_j$ n'est pas encore réalisée à la date t_k, doit être *mise à jour*. En effet, sur un tel intervalle, l'énergie disponible pour réaliser les expériences de $\mathcal{E}$ n'est plus E_{d_k} mais $E_{d_k} - (E_{bat} - E_j)$.

Si $rech_j \in \mathcal{E}_1$ (*i.e.*, st_{rech_j} et ft_{rech_j} sont instanciées), alors pour tout $k \in \{j+1..n\}$ tel que $ft_{rech_j} > t_{k-1}$, on peut effectuer la mise-à-jour suivante :

$$E_{d_k} \longleftarrow E_{d_k} - (E_{bat} - E_j)$$

En revanche, tant que $rech_j \in \mathcal{E}_2$, on ne connaît pas de façon exacte les intervalles sur lesquels on peut effectuer cette mise-à-jour. On peut cependant en donner un "encadrement" en considérant d'une part, que la recharge de la batterie est effectuée au plus tard et, d'autre part, qu'elle est effectuée au plus tôt. Dans le premier cas, on effectue la mise-à-jour de E_{d_k}

pour tout $k \in \{j+1..n\}$ tel que $k \in \{j+1..n\}$ tel que $lft_{rech_j} > t_{k-1}$, dans le second cas, on effectue la mise-à-jour de E_{d_k} pour tout $est_{rech_j} + du_{rech_j} > t_{k-1}$.

Pour tout $j = 1$ à n tel que $E_j < E_{bat}$, on obtient ainsi un encadrement de E_{d_k} pour tout $k \in \{j+1..n\}$ (voir figure 6.9) :

$$\underline{E}_{d_k} \leq E_{d_k} \leq \overline{E}_{d_k}$$

avec

$$\underline{E}_{d_k} = \begin{cases} E_{d_k} - (E_{bat} - E_j) & \text{si } lft_{rech_j} > t_{k-1} \\ E_{d_k} & \text{sinon} \end{cases}$$

et

$$\overline{E}_{d_k} = \begin{cases} E_{d_k} - (E_{bat} - E_j) & \text{si } est_{rech_j} + du_{rech_j} > t_{k-1} \\ E_{d_k} & \text{sinon} \end{cases}$$

Cet encadrement doit être remis à jour après chaque bilan énergétique E_j, pour pouvoir établir le suivant.

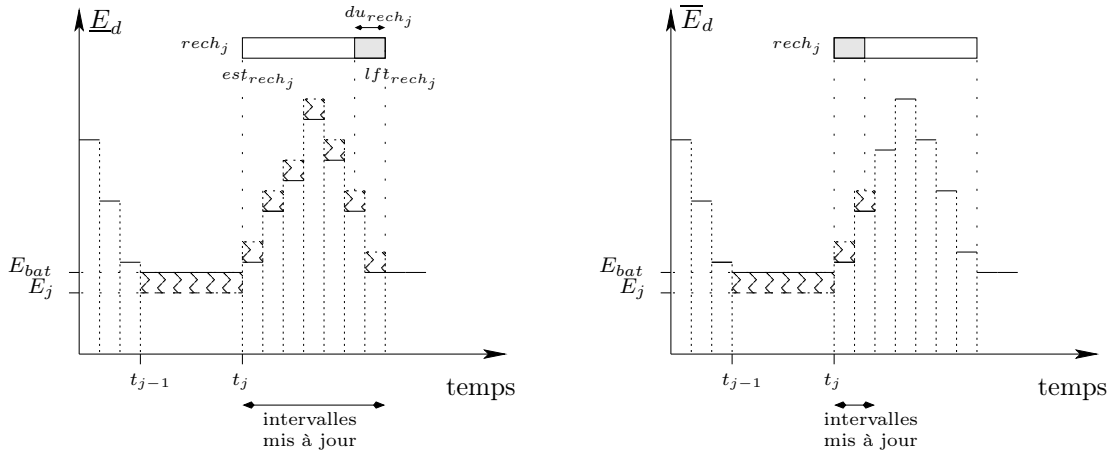


FIG. 6.9 – *Mise-à-jour des valeurs de E_d après le bilan énergétique E_j*

Évaluation de la consommation électrique sur chaque intervalle

L'énergie consommée par les expériences de $\mathcal{E}$ sur l'intervalle $[t_{j-1},t_j]$, $conso_{\mathcal{E}}(t_{j-1},t_j)$, peut s'écrire comme la somme de $conso_{\mathcal{E}_1}(t_{j-1},t_j)$ et $conso_{\mathcal{E}_2}(t_{j-1},t_j)$, où $conso_{\mathcal{E}_1}(t_{j-1},t_j)$ et $conso_{\mathcal{E}_2}(t_{j-1},t_j)$ sont, respectivement, l'énergie consommée sur $[t_{j-1},t_j]$ par les expériences de $\mathcal{E}_1$ et par les expériences de $\mathcal{E}_2$.

Comme pour la place mémoire requise par les différents types d'expériences, nous calculons de façon exacte $conso_{\mathcal{E}_1}(t_{j-1},t_j)$ pour tout intervalle $[t_{j-1},t_j]$ et nous proposons une borne inférieure $\underline{conso}_{\mathcal{E}_2}(t_{j-1},t_j)$ et une borne supérieure $\overline{conso}_{\mathcal{E}_2}(t_{j-1},t_j)$ pour $conso_{\mathcal{E}_2}(t_{j-1},t_j)$.

On a :

$$conso_{\mathcal{E}_1}(t_{j-1},t_j) = \sum_{i \in \mathcal{E}_1 \,|\, [t_{j-1},t_j] \cap [st_i,ft_i] \neq \emptyset} conso_i(t_{j-1},t_j)$$

où $conso_i(t_{j-1},t_j) = (\min(t_j,ft_i) - \max(t_{j-1},st_i)) \times P_{mop_i}$ est l'énergie consommée sur l'intervalle $[t_{j-1},t_j]$ pour l'exécution de i (rappelons que P_{mop_i} est la puissance requise pour l'exécution du mode opératoire associé à l'expérience i).

$\underline{conso}_{\mathcal{E}_2}(t_{j-1},t_j)$ est la consommation électrique *obligatoire* de $\mathcal{E}_2$ sur $[t_{j-1},t_j]$ et $\overline{conso}_{\mathcal{E}_2}(t_{j-1},t_j)$ est la consommation électrique *maximale* de $\mathcal{E}_2$ sur $[t_{j-1},t_j]$. On a :

$$\underline{conso}_{\mathcal{E}_2}(t_{j-1},t_j) = \sum_{\substack{i\in\mathcal{E}_2 \mid est_i+\underline{du}_i > t_{j-1} \\ \text{et } lft_i-\underline{du}_i < t_j}} \underline{conso}_i(t_{j-1},t_j)$$

et

$$\overline{conso}_{\mathcal{E}_2}(t_{j-1},t_j) = \sum_{i\in\mathcal{E}_2 \mid [t_{j-1},t_j]\cap[st_i,ft_i]\neq\emptyset} \overline{conso}_i(t_{j-1},t_j)$$

où

$$\underline{conso}_i(t_{j-1},t_j) = \min(est_i + \underline{du}_i - t_{j-1},\, t_j - lft_i + \underline{du}_i,\, t_j - t_{j-1},\, \underline{du}_i) \times P_{mop_i}$$

est la consommation électrique obligatoire de i sur l'intervalle $[t_{j-1},t_j]$ et

$$\overline{conso}_i(t_{j-1},t_j) = \min(lft_i - t_{j-1}\,,\, t_j - est_i\,,\, t_j - t_{j-1}\,,\, \overline{du_i}) \times P_{mop_i}$$

est la consommation électrique maximale de i sur l'intervalle $[t_{j-1},t_j]$.

On obtient ainsi un encadrement de la consommation électrique des expériences de $\mathcal{E}$ sur $[t_{j-1},t_j]$:

$$conso_{\mathcal{E}_1}(t_{j-1},t_j) + \underline{conso}_{\mathcal{E}_2}(t_{j-1},t_j) \leq\ conso_{\mathcal{E}}(t_{j-1},t_j)\ \leq conso_{\mathcal{E}_1}(t_{j-1},t_j) + \overline{conso}_{\mathcal{E}_2}(t_{j-1},t_j)$$

Ces formulations sont similaires à celles présentées précédemment pour l'élaboration des courbes de charge de la mémoire de masse d'une sonde.

Encadrement du bilan énergétique sur chaque intervalle

On obtient ainsi, un encadrement des bilans énergétiques E_j, définis précédemment pour tout intervalle $[t_{j-1},t_j]$ $(j = 1..n)$:

$$\underline{E}_j \leq E_j \leq \overline{E}_j$$

où, $\underline{E}_j$ et $\overline{E}_j$, les *bornes inférieures* et *supérieures* du bilan énergétique E_j, sont calculées itérativement pour $j = 1$ à n par l'algorithme 1.

Construction des courbes de bilans énergétiques

Soit un plan de travail en cours d'élaboration sur un horizon T et l'ensemble associé $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2$ des expériences élémentaires sélectionnées.

De même que pour la charge de la mémoire de masse, on peut établir deux courbes $\mathcal{B}_{sup}$ et $\mathcal{B}_{inf}$ représentant, respectivement, une borne supérieure et une borne inférieure de l'évolution de la ressource électrique sur T.

En revanche, contrairement à la mémoire de masse qui est une ressource consommable de capacité constante sur les intervalles définis par les dates successives de fins de communications, nous avons vu précédemment que la ressource électrique est de capacité constante

Algorithme 1 Calcul itératif des $\underline{E}_j$ et des $\overline{E}_j$ $(j = 1..n)$

Initialisation
  $\mathcal{E}'_2 \leftarrow \mathcal{E}_2$
  Pour $j = 1$ à n **faire**
    $\underline{E}_{d_j} \leftarrow E_{d_j}$
    $\overline{E}_{d_j} \leftarrow E_{d_j}$
  fin pour
fin initialisation
Pour $j = 1$ à n **faire**
  $\underline{E}_j \leftarrow \underline{E}_{d_j} - conso_{\mathcal{E}_1}(t_{j-1},t_j) - \overline{conso}_{\mathcal{E}_2}(t_{j-1},t_j)$
  $\overline{E}_j \leftarrow \overline{E}_{d_j} - conso_{\mathcal{E}_1}(t_{j-1},t_j) - \underline{conso}_{\mathcal{E}_2}(t_{j-1},t_j)$
  Si $E_j < E_{bat}$ **faire**
    créer $rech_j$
    $\mathcal{E}_2 \leftarrow \mathcal{E}_2 \cup \{rech_j\}$
    $k \leftarrow j + 1$
    // mise-à-jour des $\underline{E}_{d_k}$ et des $\overline{E}_{d_k}$
    Tant que $(t_{k-1} < est_{rech_j} + du_{rech_j})$ et $(k \leq n)$ **faire**
      $\underline{E}_{d_k} \leftarrow \underline{E}_{d_k} - (E_{bat} - E_j)$
      $\overline{E}_{d_k} \leftarrow \overline{E}_{d_k} - (E_{bat} - E_j)$
      $k \leftarrow k + 1$
    fin tant que
    Tant que $(t_{k-1} < lft_{rech_j})$ et $(k \leq n)$ **faire**
      $\underline{E}_{d_k} \leftarrow \underline{E}_{d_k} - (E_{bat} - E_j)$
      $k \leftarrow k + 1$
    fin tant que
  fin si
fin pour
$\mathcal{E}_2 \leftarrow \mathcal{E}'_2$

uniquement la nuit et évolue continûment de l'aube au crépuscule. On discrétise donc les périodes d'ensoleillement afin de pouvoir la considérer comme une ressource de capacité constante par intervalle. Il reste alors à choisir les intervalles $[t_{j-1},t_j]$ $(j = 1$ à $n)$ sur lesquels on va agréger la ressource électrique. Il s'agit d'établir un compromis entre une représentation précise du comportement de la ressource (qui tend à utiliser des intervalles très courts) et la qualité des bornes des bilans énergétiques calculées sur ces intervalles. Or, si on a des intervalles trop courts, les valeurs des consommations électriques obligatoires et maximales des expériences de $\mathcal{E}_2$ sur ces intervalles vont tendre à être très différentes les unes des autres ; les courbes $\mathcal{B}_{sup}$ et $\mathcal{B}_{inf}$ sur ces intervalles vont de même tendre à diverger (d'autant plus fortement que le sous-ensemble $\mathcal{E}_2$ est grand devant le sous-ensemble $\mathcal{E}_1$).

Du fait que l'on commence généralement l'élaboration d'un plan de travail avec peu de

connaissances sur les expériences ($|\mathcal{E}_2| >> |\mathcal{E}_1|$), nous proposons de choisir dans un premier temps de grands intervalles : soit $t_1 < \ldots < t_{n-1}$ les dates de levers, zéniths et couchers successifs du soleil sur l'horizon T, $t_0 = 0$ et $t_n = T$, on considère les intervalles $I_T = \{[t_{i-1},t_i] \mid i = 1..n\}$.

On peut envisager de resserrer ces intervalles au cours de l'élaboration du plan, en particulier, dès que les évaluations des bilans énergétiques ne permettent plus de déduire de nouvelles informations sur la validité du plan en cours. Notons toutefois qu'il est inutile de diviser les périodes de nuit en plus d'un intervalle puisque la ressource électrique est réellement de capacité constante pendant toute cette durée.

Pour tracer les courbes $\mathcal{B}_{sup}$ et $\mathcal{B}_{inf}$, pour tout intervalle $[t_{j-1},t_j]$ ($j = 1$ à n), on pose des règles similaires à celles utilisées pour les courbes de charge de la mémoire de masse :

- pour toute expérience $i \in \mathcal{E}_1$, telle que $[t_{j-1},t_j] \cap [st_i,ft_i] \neq \emptyset$, on considère que l'énergie $conso_i(t_{j-1},t_j)$ est consommée à la date $\tau = \max(st_i,t_{j-1})$;
- pour toute expérience $i \in \mathcal{E}_2$, telle que $[t_{j-1},t_j] \cap [est_i,lft_i] \neq \emptyset$, on considère que l'énergie $\overline{conso}_i(t_{j-1},t_j)$ est consommée à la date $\tau = \max(est_i,fc_{j-1})$ (pour la courbe $\mathcal{B}_{inf}$) ;
- pour toute expérience $i \in \mathcal{E}_2$, telle que $est_i + \underline{du}_i > fc_{i-1}$ et $lft_i - \underline{du}_i < fc_i$, on considère que l'énergie $\underline{data}_i(fc_{j-1},fc_j)$ est consommée à la date $\tau = \max(lft_i - \underline{du}_i, fc_{j-1})$ (pour la courbe $\mathcal{B}_{sup}$) ;
- L'énergie disponible sur l'intervalle $[t_{j-1},t_j]$ est $\overline{E}_{d_j}$ pour la courbe $\mathcal{B}_{sup}$ et $\underline{E}_{d_j}$ pour la courbe $\mathcal{B}_{inf}$, calculés dans l'algorithme 1.

La figure 6.10 met en œuvre ces règles pour établir les courbes $\mathcal{B}_{sup}$ et $\mathcal{B}_{inf}$ sur un intervalle $[t_0,t_1]$. Le plan de travail considéré est constitué de quatre expériences : les expériences 1, 2 et 3 appartiennent à $\mathcal{E}_2$ et l'expérience 4 appartient à $\mathcal{E}_1$.

6.3.2.3 Analyse des courbes, règles de déduction

Afin de donner des indications sur la faisabilité d'un plan de travail (relativement à la contrainte de partage de la ressource électrique) au cours de son élaboration, on utilise les règles de déduction suivantes, issues de l'analyse énergétique :

- S'il existe j tel que $\overline{E}_j < 0$ (la courbe $\mathcal{B}_{sup}$ passe sous l'axe des abscisses sur l'intervalle $[t_{j-1},t_j]$), le plan de travail ne peut pas être réalisable. En effet, le bilan énergétique est nécessairement négatif sur cet intervalle ; on a donc un viol de la contrainte de partage de la ressource électrique.
- À l'inverse, si pour tout $j = 1$ à n, $\underline{E}_j > 0$ (la courbe $\mathcal{B}_{inf}$ reste au-dessus de l'axe des abscisses sur tout l'horizon T), le plan de travail est réalisable, compte tenu des intervalles d'agrégation de la ressource choisis, quelles que soient les dates de début et fin d'exécution des expériences de $\mathcal{E}_2$. On peut alors décider de recommencer l'évaluation en resserrant les intervalles.
- S'il existe j tel que $\overline{E}_j < conso_{\mathcal{E}_1}(t_{j-1},t_j)$, alors le plan de travail restreint aux expériences de $\mathcal{E}_1$ n'est pas réalisable. Il est donc nécessaire de modifier certaines décisions prises sur les expériences de $\mathcal{E}_1$ dans les intervalles $[t_0,fc_1]$ à $[t_{j-1},fc_j]$.
- S'il existe j tel que $\overline{E}_j < \underline{conso}_{\mathcal{E}_2}(t_{j-1},t_j)$, alors le plan de travail restreint aux expériences de $\mathcal{E}_2$ n'est pas réalisable. Il est donc nécessaire de modifier certaines décisions prises sur les expériences de $\mathcal{E}_1$ dans les intervalles précédents, ou de supprimer certaines des expériences i de $\mathcal{E}$ telles que $est_i \leq fc_j$.

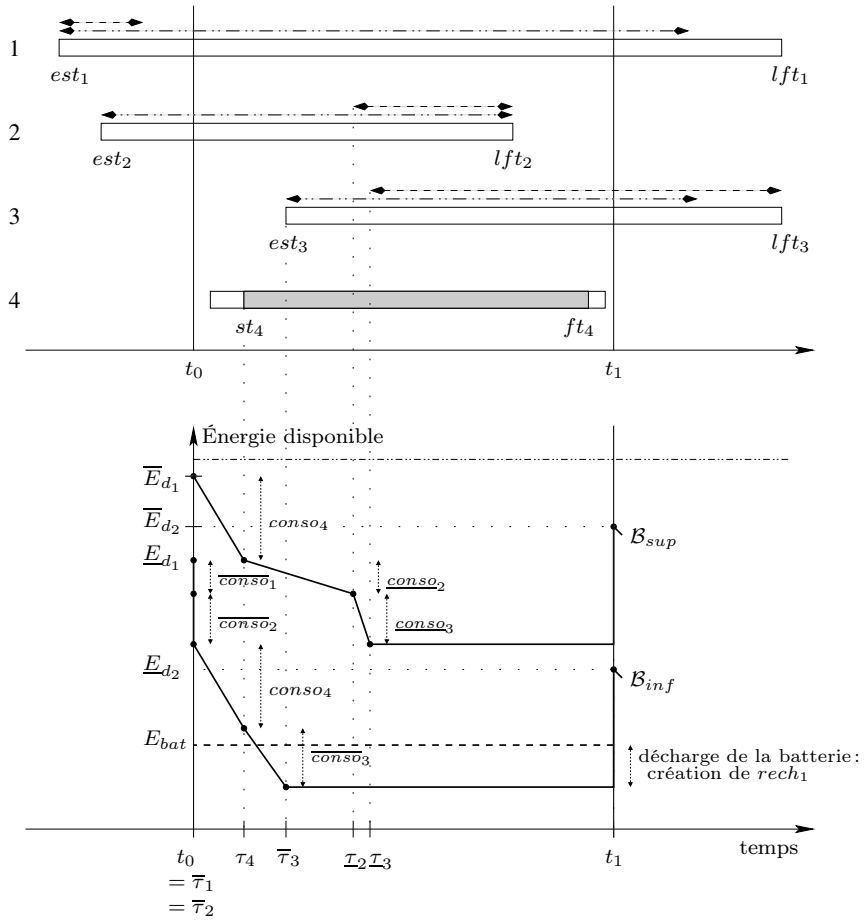


FIG. 6.10 – *Exemple de courbes $\mathcal{B}_{sup}$ et $\mathcal{B}_{inf}$ établies sur un intervalle $[t_0,t_1]$ pour un ensemble de quatre expériences élémentaires*

– Pour toute expérience $i \in \mathcal{E}_1$, soit
$$\overline{E}^i_{d_j} = \overline{E}_{d_j} - \sum_{e\in\mathcal{E}_1\setminus\{i\}} conso_e(t_{j-1},t_j) - \underline{conso}_{\mathcal{E}_2}(t_{j-1},t_j)$$
l'énergie maximale disponible pour l'exécution de i sur $[t_{j-1},t_j]$; si
$$0 < \overline{E}^i_{d_j} < conso_i(t_{j-1},t_j)$$
alors on ne peut pas exécuter à la fois i et l'ensemble d'expériences $\mathcal{E}_1 \setminus \{i\} \cup \mathcal{E}_2$. Si on peut décaler l'exécution de i, ou réduire sa durée de réalisation de telle façon que $[st_i,ft_i] \cap [t_{j-1},t_j]$ soit un intervalle de longueur inférieure à $\overline{E}^i_{d_j}/P_{mop_i}$, le plan devient alors réalisable sur cet intervalle. Sinon, il faut modifier certaines décisions prises sur les expériences de $\mathcal{E}_1$ qui débutent avant t_j, ou bien supprimer certaines des expériences e de $\mathcal{E}$ telles que $est_e \leq t_j$.

– Pour toute expérience $i \in \mathcal{E}_2$, soit
$$\overline{E}^i_{d_j} = \overline{E}_{d_j} - conso_{\mathcal{E}_1}(t_{j-1},t_j) - \sum_{e\in\mathcal{E}_2\setminus\{i\}} \underline{conso}_e(t_{j-1},t_j)$$
l'énergie maximale disponible pour l'exécution de i sur $[t_{j-1},t_j]$; si
$$0 < \overline{E}^i_{d_j} < \underline{conso}_i(t_{j-1},t_j)$$
alors on ne peut pas exécuter à la fois i et l'ensemble d'expériences $\mathcal{E}_1 \cup \mathcal{E}_2 \setminus \{i\}$. On doit donc modifier certaines décisions prises sur les expériences de $\mathcal{E}_1$ qui débutent avant t_j,

ou bien supprimer certaines des expériences e de $\mathcal{E}$ telles que $est_e \leq t_j$.

Lorsque $\mathcal{E} = \mathcal{E}_1$ (toutes les expériences élémentaires sélectionnées sont ordonnancées) et que la courbe $\mathcal{B}_{sup}$ ne permet de détecter aucun viol de la contrainte de partage de la ressource électrique ($\mathcal{B}_{sup}$ est au dessus de l'axe des abscisses sur T), il reste à instancier les dates de début et de fin des tâches de recharge de la batterie, $\{rech_j \mid j \in \{1..n\}\}$, éventuellement créées lors de la dernière évaluation des bilans énergétiques (par l'algorithme 1).

Notons que si la courbe $\mathcal{B}_{inf}$ ne passe jamais sous l'axe ($E = E_{bat}$) sur T, on n'a aucune tâche de recharge de la batterie. De plus, si $\mathcal{B}_{inf}$ ne passe jamais sous l'axe des abscisses sur T, le plan de travail est réalisable, quelles que soient les dates de début et de fin des $rech_j$ ($j \in \{1..n\}$) choisies. En revanche, s'il existe un intervalle sur lequel $\mathcal{B}_{inf}$ passe sous l'axe des abscisses, on n'est pas assuré de pouvoir trouver un plan réalisable. On peut être alors amené à modifier certaines décisions prises sur $\mathcal{E}_1$.

Lorsque $\mathcal{E} = \mathcal{E}_1$ et que toutes les dates de début et de fin des $rech_j$ ($j \in \{1..n\}$) sont instanciées, les courbes $\mathcal{B}_{sup}$ et $\mathcal{B}_{inf}$ se confondent (on a $\underline{E}_j = E_j = \overline{E}_j$, pour tout $j = 1$ à n).

6.3.3 Exemple

Afin d'illustrer l'utilisation que l'on peut faire des courbes présentées précédemment, pour l'élaboration d'un plan de travail d'une sonde, considérons l'exemple caractérisé par les éléments suivants :

- un horizon T de planification de 24 heures ($t_0 = 0$h et $T = 24$h) ;
- un unique créneau de communication sonde/orbiteur $[sc_1, fc_1] = [9h40,10h]$ (on a donc deux intervalles, [0h,10h] et [10h,24h], pour l'étude de la charge de la mémoire de masse) ;
- $\delta_{com} = 15$Mbit/h ;
- $M_0 = 8{,}5$Mbit, $MdM = 20$Mbit ;
- la puissance électrique fournie sur T, donnée par la courbe de la figure 6.11, permet de déterminer les niveaux d'énergie sur les intervalles $I_T = \{[0h,8h][8h,16h][16h,24h]\}$: $E_{d_1} = 9$Wh, $E_{d_2} = 8$Wh, $E_{d_3} = E_{bat} = 2{,}3$Wh ;
- $P_{rech} = 1W$, $\alpha = 1{,}3$;
- l'ensemble des expériences élémentaires candidates, $\mathcal{E}c$, est décrit dans le tableau 6.5.

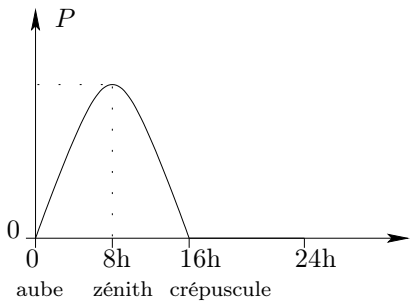


FIG. 6.11 – *Profil de la puissance fournie pour l'exemple*

Supposons que l'on commence l'élaboration d'un plan de travail sur T en posant $\mathcal{E} = \mathcal{E}c$, $\mathcal{E}_1 = \emptyset$ et $\mathcal{E}_2 = \mathcal{E}$.

expérience i	instrument	$[est_i,lft_i]$	$\frac{\underline{du}_{mop_i}}{\overline{du}_{mop_i}}$	δ_{mop_i} (kbit/h)	P_{mop_i} (mW)
1	magnétomètre	[0h,24h]	19h 24h	160	420
2	capteur de champ électrique	[0h,16h]	10h 16h	30	380
3	station météo	[0h,24h]	20h 24h	150	300
4		[0h,8h]	1h 2h	1000	580
5	capteur thermique	[0h,24h]	0h 24h	20	40
6		[5h,11h]	1h 2h	60	500
7	microphone	[0h,16h]	5h 10h	300	200
8	caméra panoramique	[0h,8h]	2'	30000	1800
9		[8h,16h]			

TAB. 6.5 – *Caractéristiques des expériences élémentaires candidates*

On obtient les consommations obligatoire et maximale par intervalle représentées aux figures 6.12 et 6.13 pour l'occupation de la mémoire de masse et 6.14 et 6.15 pour la consommation électrique.

Ceci nous permet de dresser les bilans $\underline{M}_1$, $\overline{M}_1$, $\underline{M}_2$ et $\overline{M}_2$ pour la mémoire de masse et $\underline{E}_1$, $\overline{E}_1$, $\underline{E}_2$, $\overline{E}_2$, $\underline{E}_3$ et $\overline{E}_3$ pour l'énergie électrique et de tracer les courbes $\mathcal{C}_{inf}$, $\mathcal{C}_{sup}$, $\mathcal{B}_{inf}$ $\mathcal{B}_{sup}$ correspondantes (voir figures 6.16 et 6.17).

À cette étape, le bilan $\overline{E}_3 = -1{,}6$Wh, nous permet de déduire que le plan en cours d'élaboration n'est pas réalisable. En effet, dans le meilleur des cas, l'exécution de l'ensemble des expériences de $\mathcal{E}$ entraîne une trop forte consommation énergétique sur l'intervalle [16h,24h].

De plus, pour $i = 1$ et $i = 3$, on a $0 < \overline{E}^i_{d_3} < \underline{conso}_i$(16h,24h), on ne peut donc pas exécuter à la fois 1 et l'ensemble d'expériences $\mathcal{E} \setminus \{1\}$ ni 3 et l'ensemble d'expériences $\mathcal{E} \setminus \{3\}$. Comme $\mathcal{E}_1 = \emptyset$, on doit donc supprimer l'expérience 1 ou l'expérience 3.

Supposons que l'on choisisse de retirer l'expérience 3 ($\mathcal{E} \leftarrow \mathcal{E} \setminus \{3\}$). On a donc $\mathcal{E} = \mathcal{E}_2 = \{1,2,4,5,6,7,8\}$.

Supposons qu'à une étape suivante de l'élaboration du plan de travail, on ait pris un ensemble de décisions tel que l'on ait maintenant $\mathcal{E}_1 = \{1,2,5,6\}$ (avec les dates de début et fin d'exécution des expériences de $\mathcal{E}_1$ définies dans le tableau 6.6) et $\mathcal{E}_2 = \{4,7,8,9\}$.

On obtient les nouvelles consommations obligatoire et maximale représentées aux figures 6.18, 6.19, 6.20 et 6.21. Ceci nous donne de nouvelles valeurs pour les bilans et les courbes correspondantes, $\mathcal{C}_{inf}$, $\mathcal{C}_{sup}$, $\mathcal{B}_{inf}$ $\mathcal{B}_{sup}$, représentées aux figures 6.22 et 6.23.

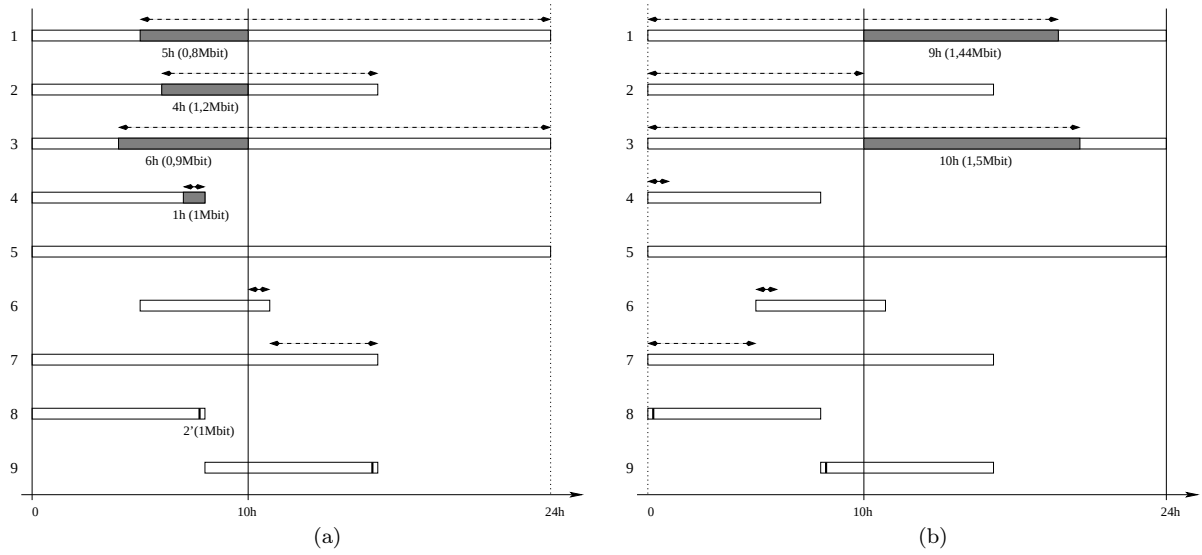


FIG. 6.12 – *Intervalles d'occupation obligatoire de la mémoire de masse sur (a)* [0*h*,10*h*] *et (b)* [10*h*,24*h*]

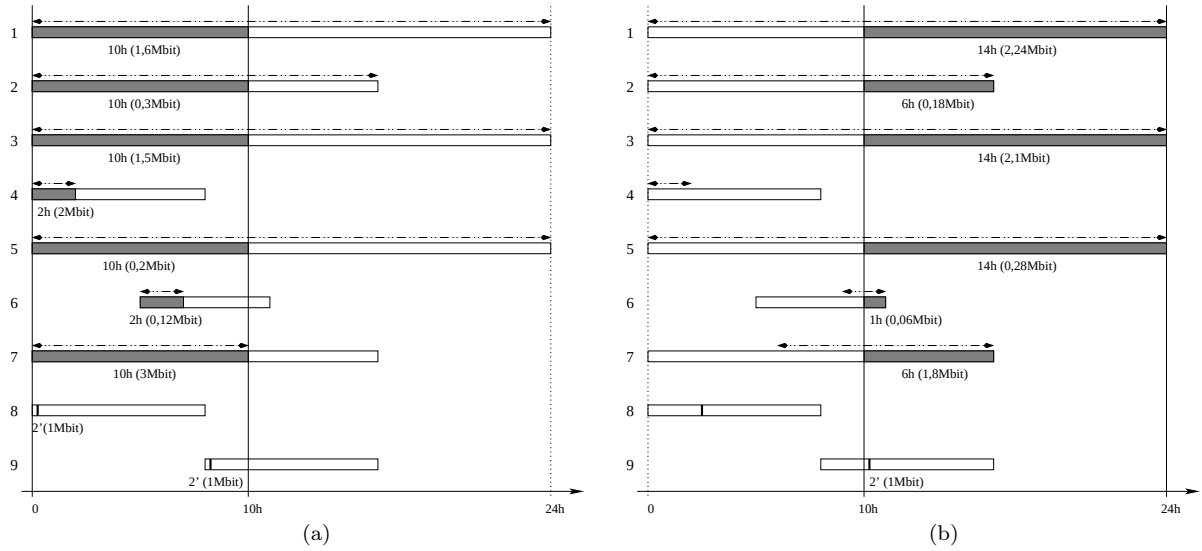


FIG. 6.13 – *Intervalles d'occupation maximale de la mémoire de masse sur (a)* [0*h*,10*h*] *et (b)* [10*h*,24*h*]

À cette étape, on constate d'une part que, bien que la courbe $\mathcal{C}_{inf}$ passe sous l'axe des abscisses sur les deux intervalles, le plan de travail restreint à $\mathcal{E} \setminus \{4\}$ est réalisable sur l'intervalle [0h,10h] et le plan de travail restreint à $\mathcal{E} \setminus \{7\}$ est réalisable sur l'intervalle [10h,24h]. Choisir une durée $du_4 \leq du_{mop_4}$, telle que :

$$du_4.\delta_{mop_4} < \underline{M}_0 - data_{\mathcal{E}_1}(0\text{h},10\text{h}) - \overline{data}_{\mathcal{E}_1\setminus\{4\}}(0\text{h},10\text{h})$$

(soit $du_4 < 1h20$), pour l'expérience 4 et une date de fin d'exécution $ft_7 < lft_7$, telle que :

$$(ft_7 - 10).\delta_{mop_7} < \underline{M}_0 - data_{\mathcal{E}_1}(10\text{h},24\text{h}) - \overline{data}_{\mathcal{E}_1\setminus\{7\}}(10\text{h},24\text{h})$$

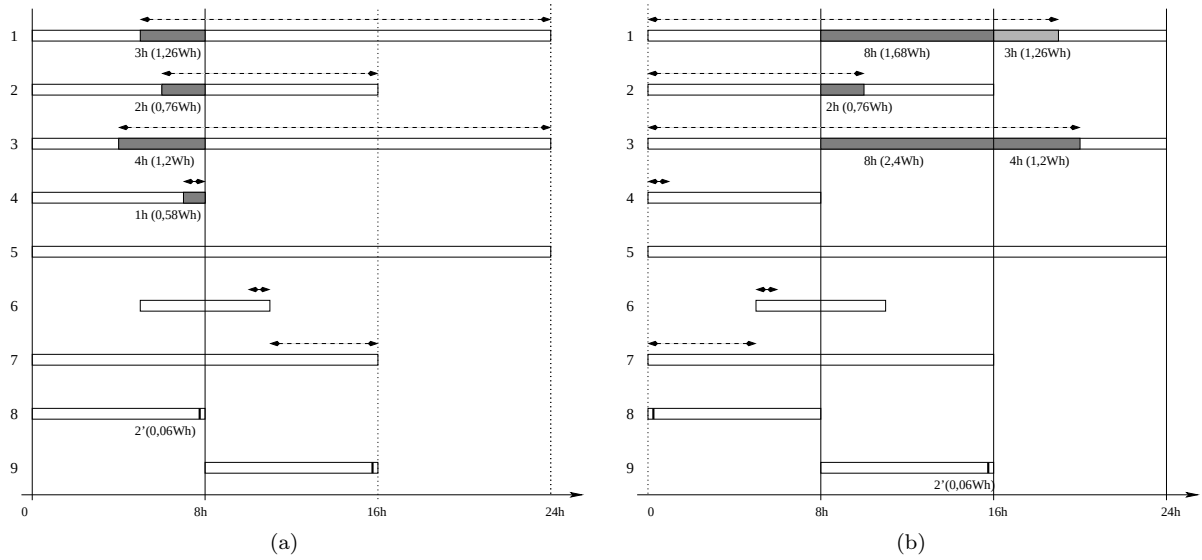


FIG. 6.14 – *Intervalles de consommation électrique obligatoire sur (a)* [0*h*,8*h*] *et (b)* [8*h*,16*h*] *et* [16*h*,24*h*]

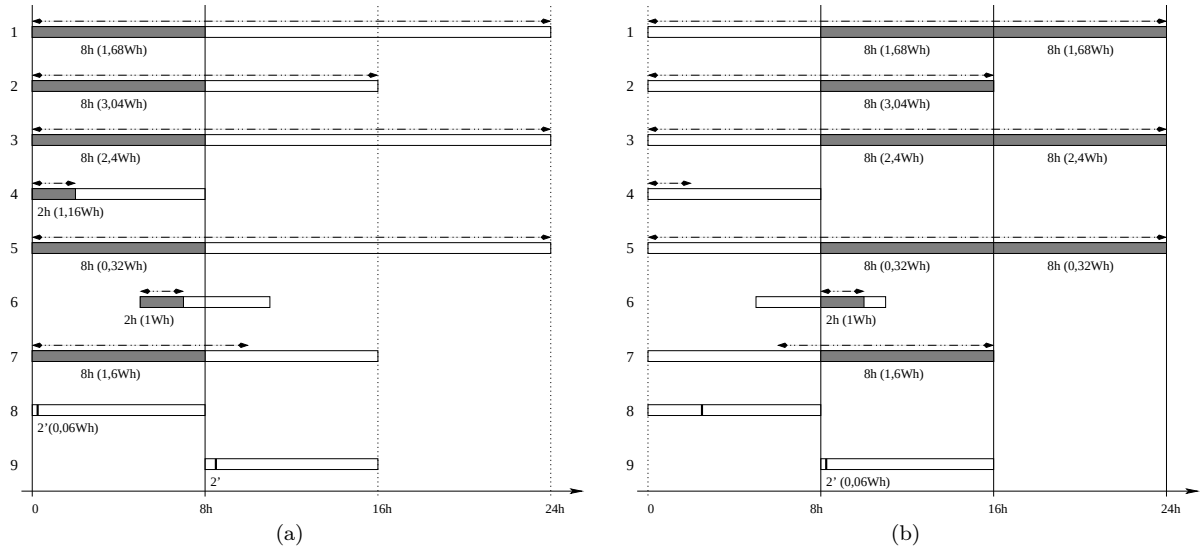


FIG. 6.15 – *Intervalles de consommation électrique maximale sur (a)* [0*h*,8*h*] *et (b)* [8*h*,16*h*] *et* [16*h*,24*h*]

(soit $ft_7 \leq 12h30$), pour l'expérience 7, permet d'assurer que le plan en cours est réalisable, du point de vue de la mémoire de masse.

On choisit alors d'instancier les dates de début et de fin d'exécution des expériences 4 et 7 comme indiqué dans le tableau 6.7. On a donc $\mathcal{E}_1 = \{1,2,4,5,6,7\}$ et $\mathcal{E}_2 = \{8,9\}$. On obtient alors les courbes $\mathcal{C}_{sup}$ et $\mathcal{C}_{inf}$ confondues (cf. figure 6.26) et les courbes $\mathcal{C}_{sup}$ et $\mathcal{C}_{inf}$ (cf. figure 6.25) qui permettent de dire que le plan de travail est réalisable, pour toutes dates de début et de fin d'exécution données aux expériences 8 et 9 de $\mathcal{E}_2$. La figure 6.24 présente ainsi une solution de plan de travail réalisable.

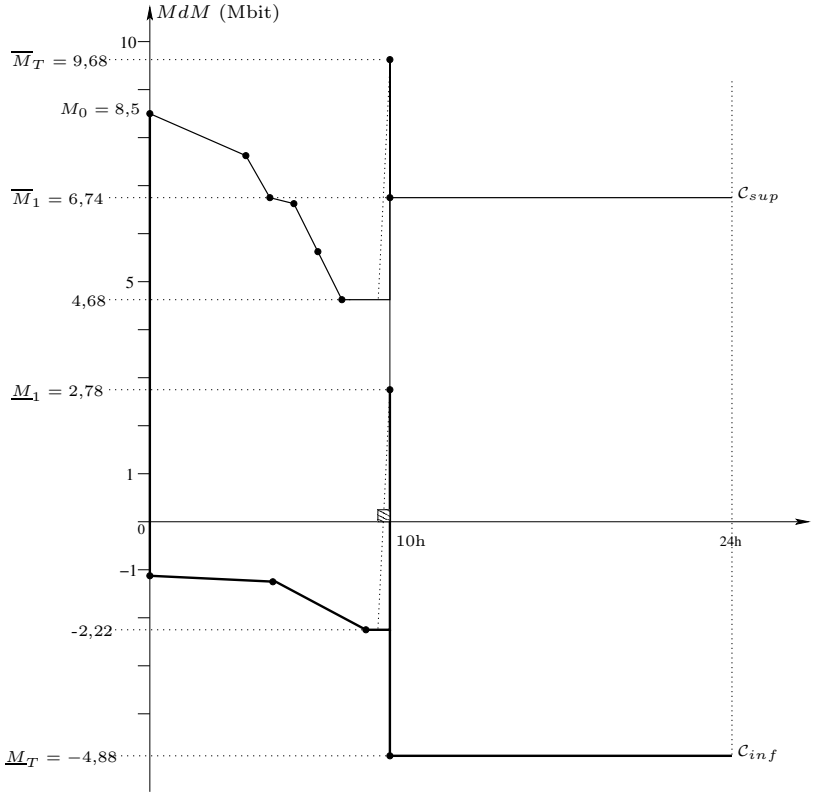


FIG. 6.16 – *Courbes de charge obligatoire, $\mathcal{C}_{sup}$, et de charge maximale, $\mathcal{C}_{inf}$ de la mémoire de masse*

i	st_i	ft_i
1	0h	24h
2	2h	12h
5	0h	24h
6	6h	8h

TAB. 6.6 – *Instanciation des dates de début et fin des expériences 1, 2, 5 et 6*

i	st_i	ft_i
4	6h	7h
7	2h30	12h30

TAB. 6.7 – *Instanciation des dates de début et fin des expériences 4 et 7*

Par la suite, on peut affiner l'analyse de la faisabilité de ce plan, en resserrant les intervalles d'étude de la ressource électrique. On peut, par exemple, poser :

$$I_T = \{[0\text{h},4\text{h}][4\text{h},8\text{h}][8\text{h},12\text{h}][12\text{h},16\text{h}][16\text{h},24\text{h}]\}$$

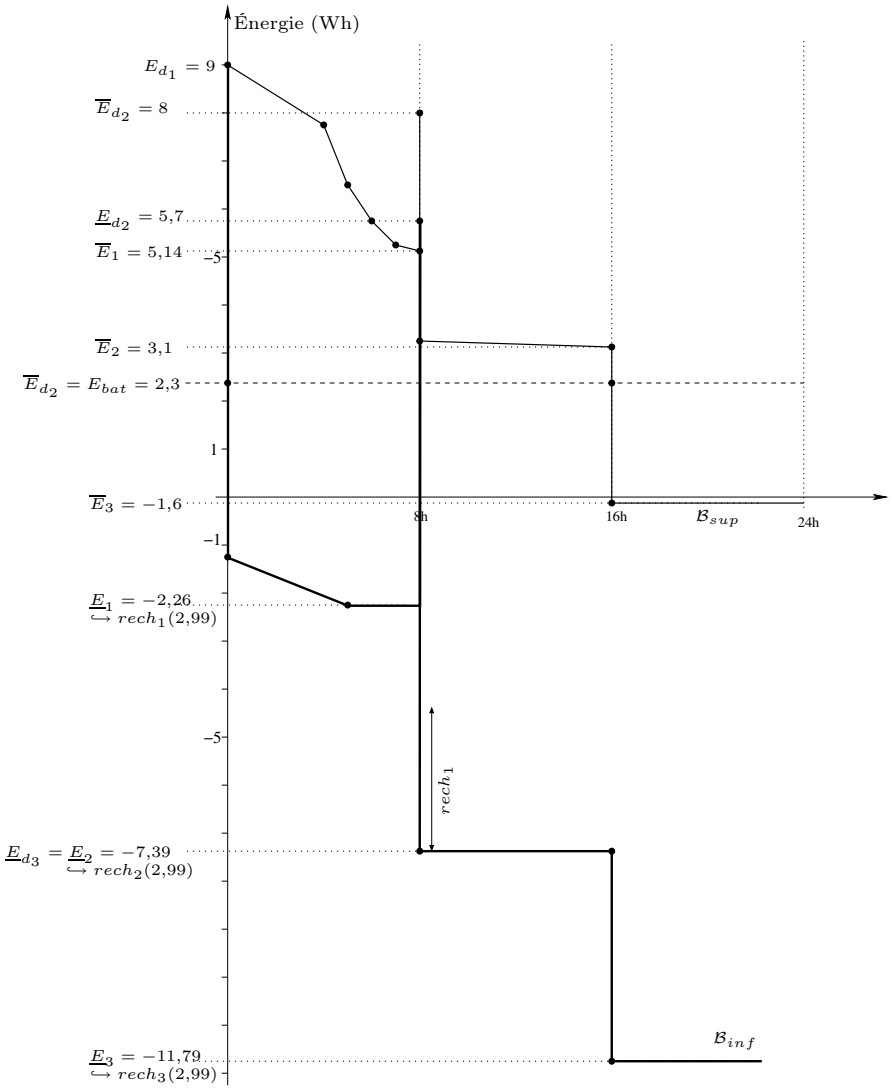


FIG. 6.17 – *Courbes de consommation électrique obligatoire, $\mathcal{B}_{sup}$, et maximale, $\mathcal{B}_{inf}$*

6.3.4 Conclusion

La planification des expériences des sondes du système Netlander émane d'une négociation entre différents experts engagés dans la mission. Compte tenu des connaissances actuelles sur le système et afin de fournir une aide pour l'élaboration des plans de travail des sondes, nous proposons le calcul de courbes d'évaluation de la charge de la mémoire de masse, ainsi que de l'énergie électrique, disponibles sur chaque sonde.

Pour établir ces courbes, nous considérons un ensemble d'expériences candidates sur un horizon de planification. Nous déterminons des intervalles temporels significatifs, sur lesquels nous dressons les bilans énergétiques liés à la consommation maximale et minimale de mémoire et d'électricité par les expériences candidates. Sur un exemple, nous montrons comment les courbes ainsi obtenues fournissent un encadrement des quantités de mémoire et d'électricité disponibles au cours du temps qui permet de guider l'élaboration d'un plan de travail.

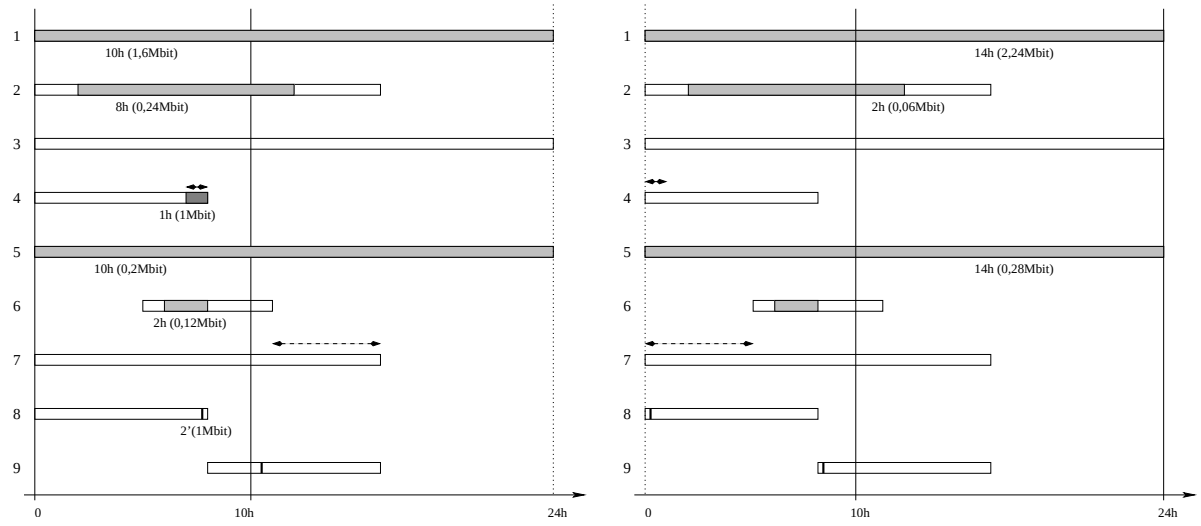


FIG. 6.18 – *Nouveaux intervalles d'occupation obligatoire de la mémoire de masse*

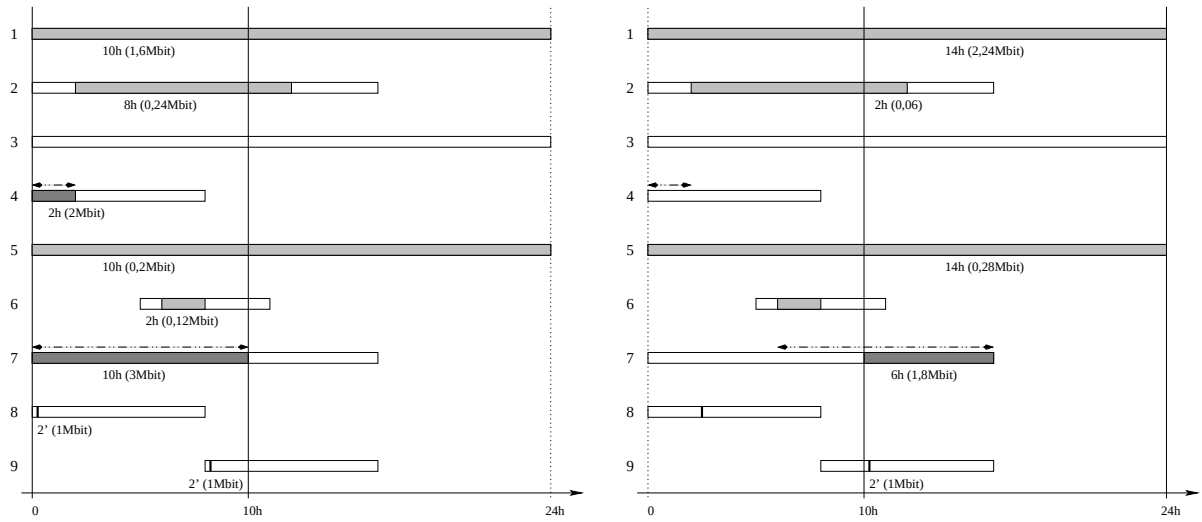


FIG. 6.19 – *Nouveaux intervalles d'occupation maximale de la mémoire de masse*

Pour poursuivre l'étude de ce problème de planification des expériences, il serait nécessaire de mieux définir les processus de négociation mis en œuvre dans le cadre de la mission Netlander. En particulier, il est très difficile dans l'état actuel des connaissances, de proposer des stratégies de prise de décision au cours de l'élaboration d'un plan de travail (sélection ou élimination d'expériences, choix sur leur durée, sur leur dates de début ou de fin). La modélisation des processus de négociation dans une organisation décisionnelle peut permettre, grâce à des mécanismes de propagation de contraintes, de renforcer les décisions proposées par des outils développés pour la gestion du temps et des ressources dans différents types de projets ou systèmes de production [Camalot 00].

Le travail présenté dans ce chapitre, a été réalisé en vue de la spécification d'un outil pour la planification des "boucles de programmation" dans le projet Netlander [Mancel *et al.* 01a]. Actuellement, l'outil MOST, conçu initialement pour la planification des expériences de la mis-

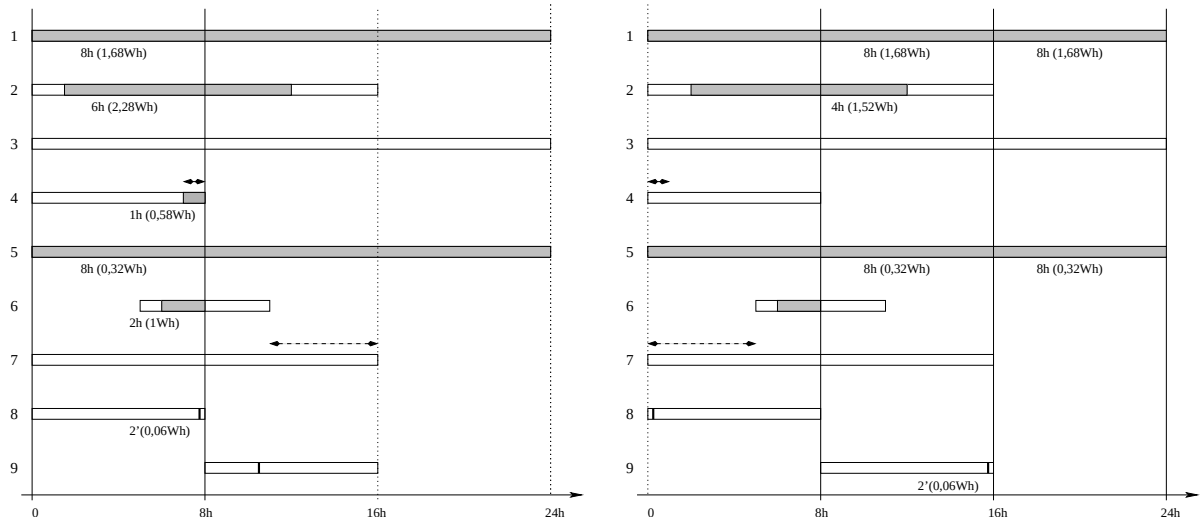


FIG. 6.20 – *Nouveaux intervalles de consommation électrique obligatoire*

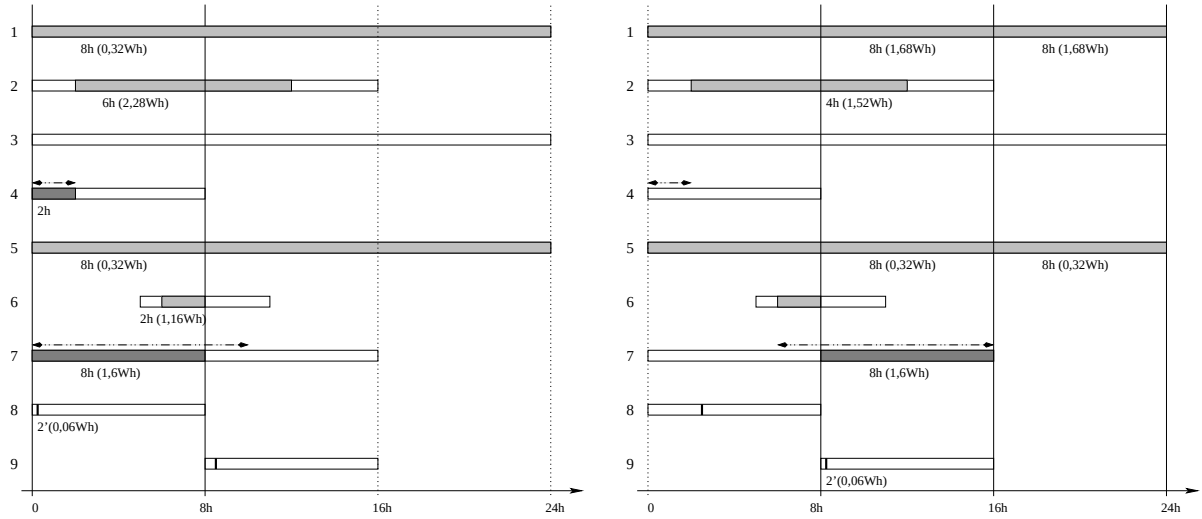


FIG. 6.21 – *Nouveaux intervalles de consommation électrique maximale*

sion Rosetta (sonde lancée le 2 mars 2004 pour l'étude de la comète Churyumov-Gerasimenko), est testé en interne au CNES afin de déterminer s'il peut intégrer aisément les méthodes que nous proposons pour le projet Netlander.

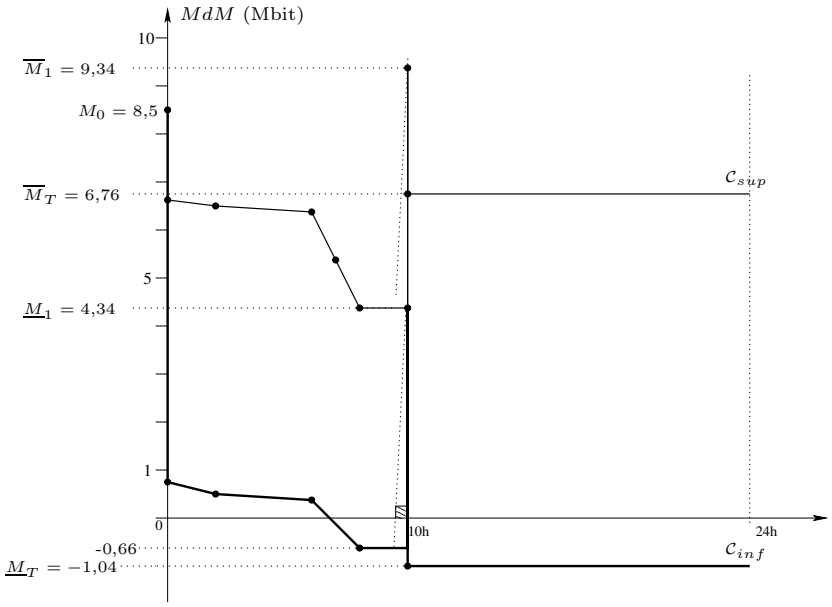


FIG. 6.22 – *Nouvelles courbes $\mathcal{C}_{sup}$ et $\mathcal{C}_{inf}$*

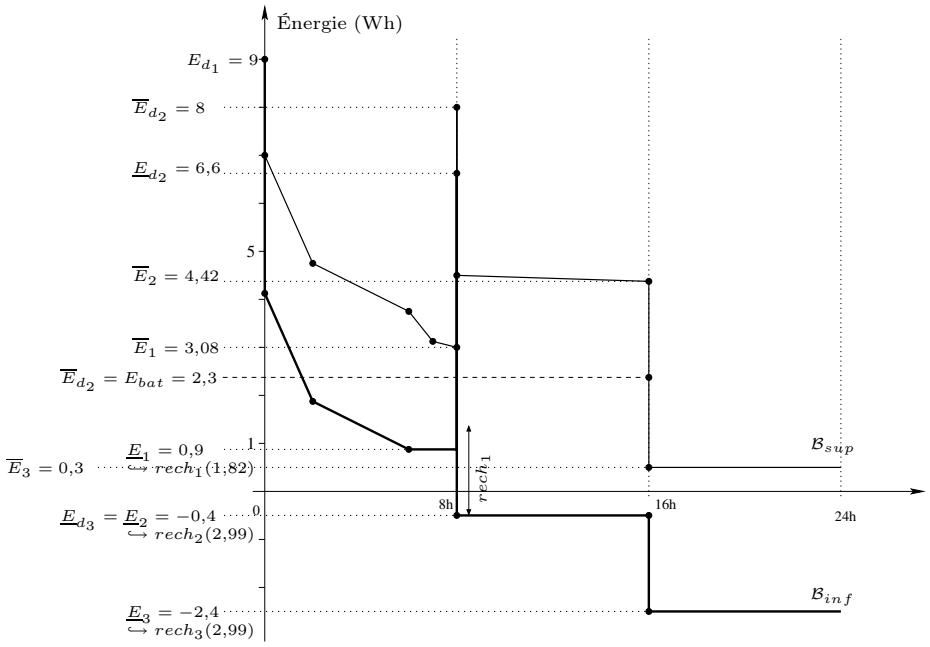


FIG. 6.23 – *Nouvelles courbes $\mathcal{B}_{sup}$ et $\mathcal{B}_{inf}$*

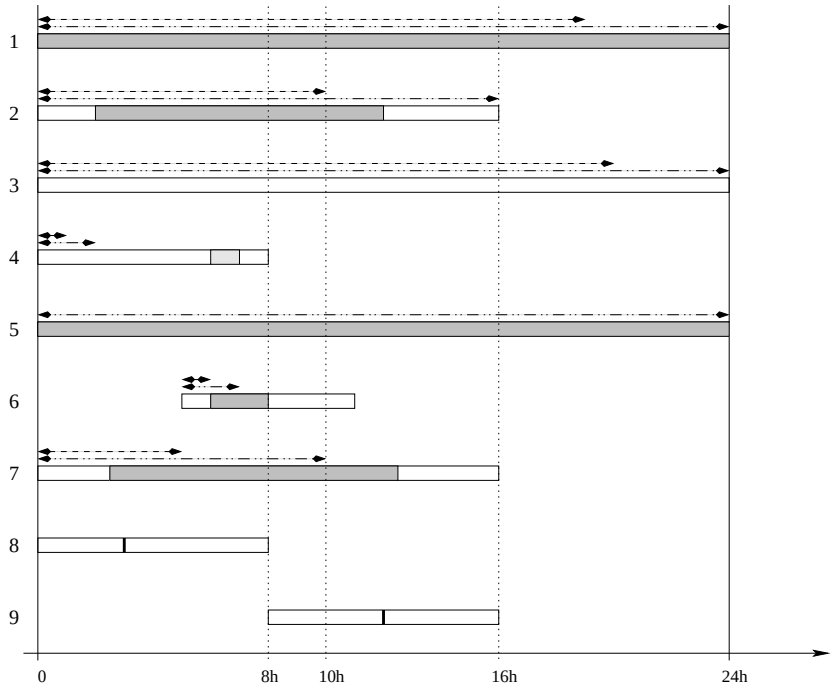


FIG. 6.24 – *Exemple de solution réalisable pour le plan de travail*

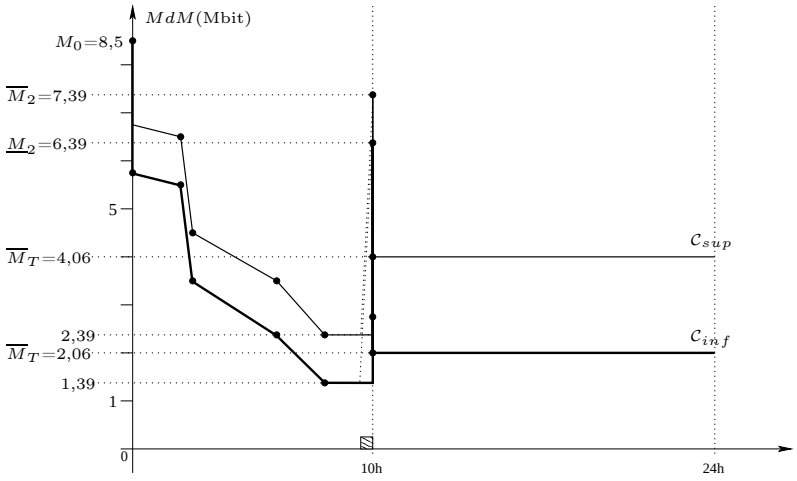


FIG. 6.25 – *Courbes $\mathcal{C}_{sup}$ et $\mathcal{C}_{inf}$ pour le plan solution*

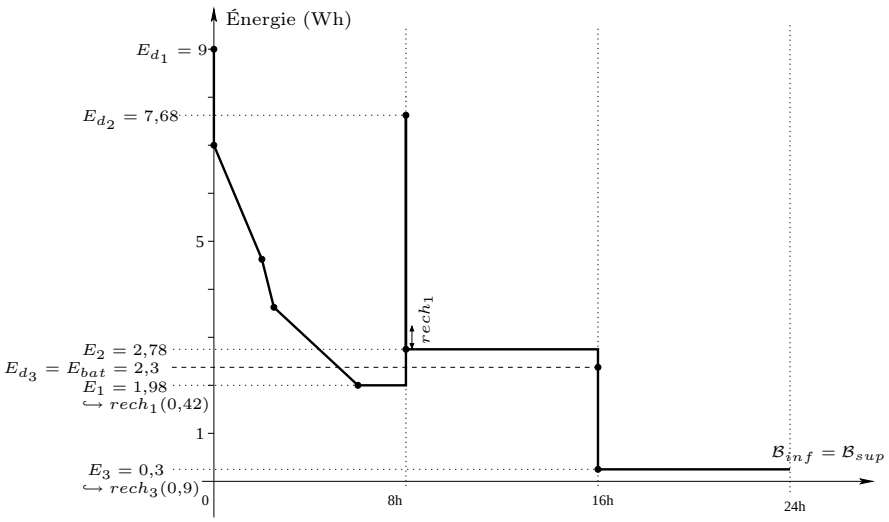


FIG. 6.26 – *Courbes $\mathcal{B}_{sup}$ et $\mathcal{B}_{inf}$ pour le plan solution*

Chapitre 7

Planification de prises de vues dans le projet Pléiades

7.1 Introduction

Le problème Pléiades, présenté au chapitre 5, consiste à planifier, c'est-à-dire sélectionner et ordonnancer, un ensemble de prises de vue devant être réalisées par un satellite au cours d'une demi-révolution autour de la Terre, de façon à maximiser un critère de qualité. Nous avons proposé un modèle, P, linéaire en variables mixtes pour ce problème et montré qu'il appartient à la classe des problèmes NP-dificiles.

Dans ce chapitre, nous proposons une décomposition du modèle P par la méthode de Dantzig-Wolfe et nous développons une procédure de génération de colonnes pour la recherche de bornes supérieures au problème Pléiades.

7.2 Positionnement bibliographique

En raison de la taille des instances étudiées et du temps de résolution fortement contraint, nous avons vu au chapitre 5 que seules des méthodes approchées sont actuellement proposées pour fournir des solutions au problème Pléiades. Il est cependant important également de pouvoir disposer d'informations sur la qualité de ces solutions. Dans ce contexte, notre objectif est, non pas de proposer une nouvelle heuristique pour la résolution du problème Pléiades, mais de founir un modèle et une méthode permettant de calculer des bornes supérieures pour évaluer la qualité des solutions approchées du problème Pléiades et, éventuellement, pour construire de meilleures solutions.

Notons que des méthodes de calcul de bornes supérieures ont été proposées dans [Benoist & Rottembourg 03] et [Vasquez *et al.* 03] à l'occasion du Challenge ROADEF 2003. La première méthode est basée sur le renforcement d'un modèle linéarisé du problème, par adjonction de coupes efficaces et sur l'utilisation d'une approche de type "poupées russes" qui consiste à résoudre un ensemble de sous-problèmes imbriqués avant de résoudre le problème global [Verfaillie *et al.* 96]. La méthode présentée dans [Vasquez *et al.* 03] considère un critère linéaire identique au critère (36) du modèle P et résout par programmation dynamique une relaxation du problème (en particulier, les contraintes stéréoscopiques sont ignorées).

Si l'on considère les contraintes de fenêtres temporelles ainsi que les temps de transitions entre les prises de vue (contraintes (27)–(30)), le problème Pléiades se rapproche d'un problème de tournée avec fenêtres temporelles (VRPTW pour *Vehicle Routing Problem with Time Windows*), où chaque prise de vue candidate peut être vue comme un client à visiter. Ce type de problème est traité avec succès par des méthodes de génération de colonnes, par exemple dans [Desrochers *et al.* 92] ou plus récemment dans [Chabrier *et al.* 02]. Dans ces travaux, le problème maître est un problème de recouvrement et l'algorithme générateur résout un problème de plus court chemin dans un graphe avec fenêtres de temps (SPPTW pour *Shortest Path Problem with Time Windows*) et contraintes de capacité. Dans [Desrochers & Soumis 88], les auteurs proposent un algorithme de programmation dynamique efficace pour résoudre le SPPTW, appelé *generalized permanent labelling algorithm* (algorithme à fixation d'étiquettes généralisé). Le principe de cet algorithme est largement utilisé pour divers problèmes de tournées traités par génération de colonnes.

Toutefois, dans le problème Pléiades, on cherche non pas à réaliser toutes les prises de vues mais à en sélectionner un sous-ensemble qui soit réalisable et qui maximise le gain obtenu. Comme nous l'avons établi au chapitre 5, on peut voir le problème Pléiades comme une généralisation du STSP (*Selective Travelling Salesman Problem*), qui est une variante à un véhicule des problèmes de tournées pour laquelle il n'est pas nécessaire de visiter tous les clients. Dans [Feillet 01], ces problèmes de tournées sélectives sont particulièrements étudiés. L'auteur définit un nouveau problème de graphe, le PTAP (Problème de Tournées avec Arcs Profitables), et propose une approche de résolution exacte par génération de colonnes et *branch and price*.

Bien qu'il en soit proche, le problème Pléiades se distingue du PTAP par deux principaux aspects. D'une part le PTAP ne contient pas de contrainte ensembliste comme la contrainte stéréoscopique ((31) et (32)). D'autre part, dans le PTAP ainsi que dans la plupart des problèmes de tournées traités par génération de colonnes, le nombre de ressources n'est pas fixé (on cherche en général à le minimiser) alors que dans le problème Pléiades, nous avons une unique ressource, le satellite d'observation.

Cependant, la génération de colonnes peut également se révéler efficace sur des problèmes à une unique machine, comme cela est montré dans [Van Den Akker 94] et [Van Den Akker *et al.* 00], où un problème d'ordonnancement est formulé par un programme linéaire indicé sur le temps et décomposé de sorte que l'algorithme générateur produise des *pseudo-ordonnancements* dans lesquels une tâche peut être exécutée zéro, une ou plusieurs fois. La contrainte de réalisation des tâches est alors laissée dans le problème maître. On peut exprimer le problème Pléiades en termes de problèmes d'ordonnancement classiquement étudiés dans la littérature. On a alors un problème à une machine (le satellite d'observation), avec un critère qui consiste à minimiser une "somme pondérée des tâches en retard" – les prises de vue non réalisées, pondérées de leur gain associé – et avec des "temps de préparation" dépendant de la séquence, qui correspondent aux durées de transition du satellite entre les prises de vue. Dans [Sourd 04], un problème d'ordonnancement présentant ces trois caractéristiques est étudié ; l'auteur propose un modèle indexé sur le temps et une relaxation lagrangienne est utilisée pour le calcul de bornes inférieures.

Enfin, comme nous l'avons évoqué aux chapitres 3 et 5, plusieurs publications font état de méthodes efficaces utilisant la génération de colonnes pour des problèmes concernant la gestion de missions de satellites [Lebbar 00, Gabrel 99].

On voit donc que le problème Pléiades considéré se rapproche par de nombreux aspects de problèmes connus, traités efficacement dans la littérature par la génération de colonnes. Il nous semble donc intéressant d'étudier les possibilités d'aborder le problème Pléiades par le biais de cette technique. Ainsi, dans le paragraphe suivant, nous appliquons la méthode de décomposition de Dantzig-Wolfe au modèle présenté au chapitre 5, puis nous proposons un processus de résolution par génération de colonnes pour le calcul de bornes supérieures.

7.3 Décomposition du problème

Nous proposons de décomposer le modèle linéaire en variables mixtes P du problème Pléiades en nous basant sur la décomposition présentée dans [Van Den Akker 94]. Nous obtenons, par la méthode de décomposition de Dantzig-Wolfe, la formulation suivante, dans laquelle les contraintes d'acquisition des bandes (24) et les contraintes stéréoscopiques ((31) et (32)) sont laissées dans le problème maître et où le sous-problème peut se ramener à un problème de recherche de plus longs chemins dans un graphe avec fenêtres de temps [Mancel *et al.* 03b].

Soit Ω l'ensemble des séquences de prises de vues satisfaisant les contraintes temporelles (contraintes (27)–(30)) du problème et que l'on nomme *pseudo-séquences* du fait qu'elles ne sont pas nécessairement réalisables (notamment vis-à-vis des contraintes stéréoscopiques). Ω contient un nombre fini d'éléments ; pour $s = 1$ à $|\Omega|$ la pseudo-séquence $se^s \in \Omega$ est une instanciation des variables $se(2j,i)$ et $se(2j+1,i)$, pour $j = 0$ à $N_B - 1$ et $i = 1$ à N_{sup}, satisfaisant les contraintes temporelles du modèle initial. À tout élément se^s de Ω, associons la variable binaire x^s égale à 1 si la colonne s, correspondant à se^s, est sélectionnée dans la solution, égale à 0 sinon.

Le problème suivant, que nous appelons Problème Maître (PM) est équivalent au problème P :

$$
PM \begin{cases}
Max \sum_s & x^s.g^s & & (38)\\
\text{s.c.} & \sum_s x^s.(se^s_{2j} + se^s_{2j+1}) \leq 1 & {\scriptstyle j=0..N_B-1} & (39)\\
& \sum_s x^s.(se^s_{2j} - se^s_{2Tw(j)}) = 0 & {\scriptstyle j=0..N_B-1 \mid Tw(j)>j} & (40)\\
& \sum_s x^s.(se^s_{2j+1} - se^s_{2Tw(j)+1}) = 0 & {\scriptstyle j=0..N_B-1 \mid Tw(j)>j} & (41)\\
& \sum_s x^s = 1 & & (42)\\
& x^s \in \{0,1\} & &\\
& se^s \in \Omega & &
\end{cases}
$$

où $se^s_k = \sum_i se^s(k,i)$ pour $k = 0$ à $2N_B - 1$ et $g^s = \sum_j gain(j).(se^s_{2j} + se^s_{2j+1})$ pour $s = 1$ à $|\Omega|$.

Le problème PM consiste à rechercher une pseudo-séquence qui soit réalisable (*i.e.,* qui sélectionne au plus une fois chaque bande candidate – contrainte (39) – et qui vérifie les contraintes stéréoscopiques – contraintes (40) et (41)) et qui permette de maximiser la somme des gains des prises de vues acquises.

On constate que le programme linéaire en nombres entiers PM a un nombre réduit de contraintes par rapport au modèle initial. En revanche, l'ensemble Ω, qui détermine le nombre de ses colonnes, est de taille exponentielle vis-à-vis du nombre de prises de vues candidates du problème, ce qui empêche une résolution de la relaxation linéaire par des algorithmes classiques

comme le simplexe. En effet, à moins de considérer de très petites instances, on ne peut pas déterminer l'ensemble des pseudo-séquences possibles. On va donc utiliser la technique de génération de colonnes, adaptée à ce type de problèmes comportant un grand nombre de variables, pour résoudre la relaxation linéaire de PM.

Comme nous l'avons vu au chapitre 2, la résolution par génération de colonnes repose sur la possibilité d'identifier dans un sous-problème (ou problème esclave) les colonnes susceptibles d'appartenir à la solution optimale. Considérons la restriction $PMR_{\Omega'}$ (Problème Maître Restreint), du problème PM à un sous-ensemble $\Omega' \subset \Omega$ de ses colonnes, suffisamment petit pour que l'on puisse résoudre sa relaxation linéaire par l'algorithme du simplexe. S'il existe dans $\Omega \backslash \Omega'$ une colonne de profit marginal positif, alors on peut espérer augmenter la fonction de maximisation de $PMR_{\Omega'}$ en ajoutant à Ω' la variable x^s correspondante. Dans le cas contraire, la solution optimale obtenue pour la relaxation linéaire de $PMR_{\Omega'}$ est également la solution optimale de la relaxation linéaire de PM.

Le sous-problème (SP) consiste donc ici à chercher des pseudo-séquences de prises de vues de profit marginal positif.

Soient π_j les variables duales associées aux contraintes (39) de PM pour $j = 0$ à $N_B - 1$, λ_{2j} et λ_{2j+1} les variables duales associées respectivement aux contraintes (40) et (41) pour $j = 0$ à $N_B - 1$ tel que $Tw(j) > j$ et μ la variable duale associée à la contrainte (42).

Le profit marginal d'une colonne s, $\tilde{g}^s$, est défini par l'égalité :

$$\tilde{g}^s = g^s - \sum_j \pi_j (se^s_{2j} + se^s_{2j+1}) - \sum_j \lambda_{2j} (se^s_{2j} - se^s_{2Tw(j)}) - \sum_j \lambda_{2j+1} (se^s_{2j+1} - se^s_{2Tw(j)+1}) - \mu$$

Posons, pour tout $j = 0$ à $N_B - 1$:

$$\lambda'_{2j} = \begin{cases} \lambda_{2j} & \text{si } Tw(j) > j \\ -\lambda_{2j} & \text{si } 0 < Tw(j) < j \\ 0 & \text{sinon } (i.e.,\ Tw(j) = -1, \text{ signifiant que } j \text{ n'a pas de bande jumelle}) \end{cases}$$

et

$$\lambda'_{2j+1} = \begin{cases} \lambda_{2j+1} & \text{si } Tw(j) > j \\ -\lambda_{2j+1} & \text{si } 0 < Tw(j) < j \\ 0 & \text{sinon} \end{cases}$$

On peut alors reformuler l'expression de $\tilde{g}^s$ par :

$$\tilde{g}^s = g^s - \sum_k se^s_k (\pi_{|k/2|} + \lambda'_k) - \mu$$

ou encore :

$$g^s = se^s . \tilde{g} - \mu$$

où $\tilde{g}_k = gain(|k/2|) - \pi_{|k/2|} - \lambda'_k$ est le profit marginal associé à une prise de vue k et $se^s . \tilde{g} = \sum_k se^s_k \times \tilde{g}_k$.

Ainsi, le sous-problème consiste à rechercher des instanciations des variables $se(k,i)$ qui vérifient les contraintes (25) à (30) du modèle initial et de profit marginal positif, c'est-à-dire tel que : $\sum_k se_k . \tilde{g}_k > \mu$.

Si on exprime SP comme un problème de maximisation avec le critère :

$$Max\ z = \sum_{k} se_k \times \tilde{g}_k \tag{43}$$

pour un $\Omega' \subset \Omega$ donné, si la solution optimale z^* de SP est inférieure ou égale à μ, alors il n'existe pas de colonne permettant d'améliorer le critère de $PMR_{\Omega'}$ et on prouve ainsi que la solution optimale de la relaxation linéaire de PM est égale à celle de $PMR_{\Omega'}$.

7.4 Description du problème de génération des pseudo-séquences de prises de vue

Comme nous l'avons établi précédemment, le sous-problème SP consiste à chercher des pseudo-séquences de prises de vues de profit marginal positif. Nous formulons ce problème comme un problème de recherche de plus longs chemins dans un graphe de la façon suivante.

Soit $G = (X,U)$ un graphe orienté où X, l'ensemble des sommets du graphe, représente l'ensemble des prises de vue candidates et U, l'ensemble des arcs, représente les successions possibles entre les prises de vue. À chaque sommet k de X, on associe la fenêtre temporelle $[est_k\ ,\ lst_k]$ de début d'acquisition de la prise de vue correspondante ($[est_k\ ,\ lst_k] = [Tmin0(|k/2|)\ ,\ Tmax0(|k/2|)]$ si k est paire et $[est_k\ ,\ lst_k] = [Tmin1(|k/2|)\ ,\ Tmax1(|k/2|)]$ si k est impaire), ainsi que le gain $\tilde{g}_k$, défini précédemment comme le profit marginal lié à l'acquisition de la prise de vue k. Notons que ce gain $\tilde{g}_k$ peut prendre une valeur positive ou négative.

On associe à chaque arc (k,l) de U la durée $d_{k,l} = Du(|k/2|) + Dt(k,l)$ qui représente la somme de la durée d'acquisition de la prise de vue k et de la durée de transition entre les prises de vue k et l.

Étant donné que l'acquisition de chaque prise de vue k ne peut débuter que dans sa fenêtre $[est_k\ ,\ lst_k]$, un arc (k,l) appartient à U si et seulement s'il existe une possibilité d'exécuter la prise de vue k puis la prise de vue l, c'est-à-dire si on a l'inégalité suivante :

$$est_k + d_{k,l} \leq lst_l$$

Ce point est illustré à la figure 7.1 qui donne le graphe G représentant un exemple de sous-problème SP comportant cinq prises de vue candidates. Les durées $d_{k,l}$ entre les différentes prises de vue sont présentées au tableau 7.1. Pour chaque arc du graphe, on peut vérifier l'inégalité ci-dessus. À l'inverse, s'il n'y a pas d'arc entre deux prises de vue k et l, on peut vérifier que même en choisissant est_k comme date de début de la prise de vue k, il n'est pas possible de réaliser la prise de vue l ensuite. C'est le cas par exemple pour les prises de vue 2 et 3 pour lesquelles on a $est_2 + d_{2,3} = 10$ et $lst_3 = 9$ et donc $est_2 + d_{2,3} > lst_3$.

On ajoute à X deux sommets fictifs, o et f, "origine" et "fin", de gain nul, de fenêtre temporelle $[0,T]$ et tels que :

- $d_{k,o} = \infty\ \forall\, k \in X$,
- $d_{o,k} = 0\ \forall\, k \in X \backslash \{o\}$,
- $d_{k,f} = Du(|k/2|)\ \forall\, k \in X \backslash \{f\}$,
- $d_{f,k} = \infty\ \forall\, k \in X$.

	1	**2**	**3**	**4**	**5**
1	∞	6	6	7	8
2	4	∞	5	3	5
3	5	4	∞	3	7
4	7	4	5	∞	5
5	5	6	4	5	∞

TAB. 7.1 – *Tableau des durées $d_{k,l}$ entre cinq prises de vue*

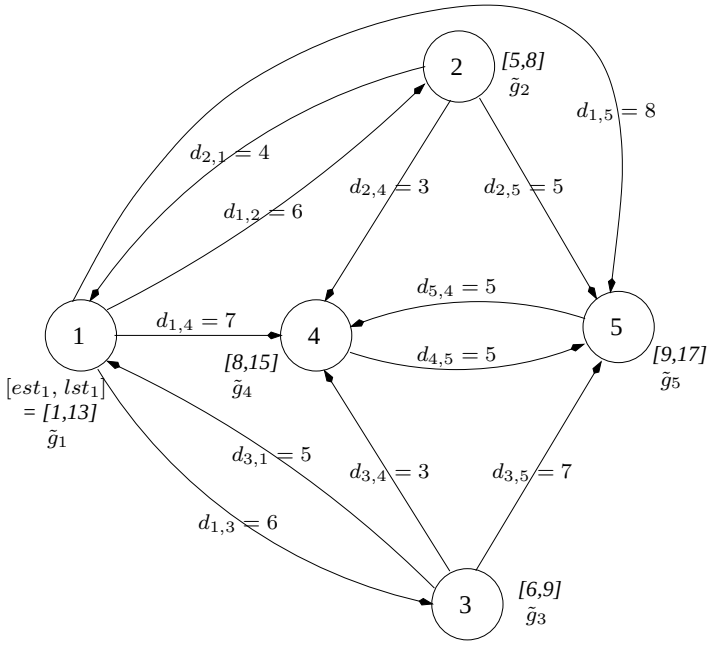


FIG. 7.1 – *Graphe G représentant un exemple à cinq prises de vue*

Un chemin élémentaire de o à f dans le graphe G correspond à une séquence de prises de vue vérifiant les contraintes temporelles du problème mais pas nécessairement les contraintes d'acquisition des bandes (deux prises de vues correspondant à la même bande peuvent appartenir à un même chemin) ni les contraintes stéréoscopiques. Un chemin élémentaire de o à f dans le graphe G est donc une pseudo-séquence de prises de vue telle que définie précédemment.

Appelons *longueur* d'un chemin du graphe G la somme des gains associés à ses sommets. La recherche d'une pseudo-séquence de prises de vue de profit marginal positif se ramène alors à la recherche d'un chemin élémentaire de longueur supérieure à μ entre les sommets o et f du graphe G. Le sous-problème SP peut donc être formulé comme un problème de plus longs chemins élémentaire dans un graphe avec fenêtres de temps, ce qui est équivalent à un problème de plus court chemin élémentaire dans un graphe avec fenêtres de temps (ESPPTW pour *Elementary Shortest Path Problem With Time Windows*).

Pour la plupart des applications de la génération de colonnes à des problèmes de tournées de véhicules, le sous-problème est un ESPPTW dont on relâche la contrainte imposant d'avoir des chemins élémentaires, afin d'en accélérer la résolution (le sous-problème devient un SPPTW). Cette relaxation nécessite parfois une modification du problème maître qui doit

pouvoir prendre en compte des colonnes correspondant éventuellement à des chemins non élémentaires générés par le sous-problème.

Dans le cas du problème Pléiades, il est clair que relaxer la contrainte d'élémentarité des chemins dans le sous-problème peut amener à générer des pseudo-séquences réalisant plusieurs fois une même prise de vue (si elle est de fort gain par exemple). Cependant, l'existence d'une fenêtre de réalisation pour chaque sommet du graphe et d'une durée positive sur chaque arc assure qu'il n'y a pas de cycle infini. De plus, relaxer cette contrainte n'entraîne aucune modification dans la formulation du problème maître PM ; en effet, la contrainte (39) qui assure qu'une bande est acquise par au plus une des deux prises de vues qui lui sont associées est toujours valide dans le cas où, pour certaines colonnes, le coefficient de cette contrainte est un entier supérieur à 1 (se_k^s est égal au nombre de fois où k est visité dans le chemin correspondant à la colonne s ; dans le cas où on autorise les cycles dans les chemins, se_k^s ne prend plus ses valeurs seulement dans $\{0;1\}$, mais dans l'ensemble des entiers naturels).

Dans le cadre d'une méthode de génération de colonnes, ce type de problème est généralement résolu par des algorithmes de programmation dynamique. En particulier, dans [Larsen 99] l'auteur propose un algorithme à fixation d'étiquettes de type Dijkstra pour résoudre un sous-problème ESPPTW avec des contraintes de ressources supplémentaires dans un contexte de résolution d'un problème de tournée de véhicules par génération de colonnes. Dans la suite, nous proposons une adaptation de cet algorithme à la résolution de SP.

7.5 Algorithme de génération de colonnes pour le problème Pléiades

7.5.1 Génération des pseudo-séquences de prises de vue

7.5.1.1 Chemins élémentaires, algorithme PLPSE

Dans les algorithmes de programmation dynamique pour des problèmes de plus court (ou plus long) chemin avec contraintes supplémentaires, comme le ESPPTW, la notion d'optimalité d'un chemin est liée à la définition d'une relation d'ordre partiel entre les chemins du graphe. Cette relation est appelée *critère de dominance* ; dans un graphe sans cycle, un premier chemin du sommet origine o à un sommet quelconque s du graphe *domine* un second chemin de o à s s'il est de coût inférieur ou égal au coût du second tout en consommant moins de ressources. Dans un graphe contenant des cycles de coût négatif (ou positif dans le cas d'un problème de plus longs chemins), un premier chemin élémentaire de o à s domine un second si les conditions précédentes sont observées et si, de plus, toute "extension" du second chemin à d'autres sommets est également une extension possible du premier chemin (voir par exemple [Chabrier *et al.* 02], qui modifie ainsi le critère de dominance d'un algorithme de plus court chemin élémentaire à fixation d'étiquettes pour générer des tournées sans cycle de véhicules dans un procesus de génération de colonnes).

L'algorithme présenté dans [Larsen 99] se base sur le principe de l'algorithme de Dijkstra pour résoudre un SPPTW avec contraintes de ressources supplémentaires (SPPTWCC, pour *Shortest Path Problem with Time Windows and Capacity Constraints*). Cet algorithme maintient un ensemble de chemins "partiels" dominants qui partent du sommet origine o et les étend itérativement jusqu'au sommet final f du graphe de façon à construire un ensemble

de chemins dominants entre o et f. Le chemin de plus petit coût dans cet ensemble est alors le chemin optimal. Nous proposons d'adapter cet algorithme à la recherche d'un plus long chemin élémentaire dans le graphe G, afin de résoudre le sous-problème SP [Mancel 03].

Définition d'un premier critère de dominance

Soit c_k un chemin élémentaire partiel de o à un sommet $k \in X$ dans le graphe G. Appelons $S(c_k)$ la liste des sommets du chemin c_k. On associe l'étiquette $E^{c_k} = (k,\, \tilde{g}c_k,\, dc_k)$ au chemin c_k, où $\tilde{g}c_k$ est la longueur de c_k, $\tilde{g}c_k = \sum_{l \in S(c_k)} \tilde{g}_l$, égale au profit marginal de la pseudo-séquence de prises de vues correspondante et dc_k est la date de début d'exécution de la prise de vue associée à k dans le chemin c_k.

Introduisons un premier critère de dominance : soit c_k et c'_k deux chemins élémentaires partiels du sommet o à un sommet k de G, caractérisés par les étiquettes $(k,\, \tilde{g}c_k,\, dc_k)$ et $(k,\, \tilde{g}c'_k,\, dc'_k)$; c_k domine (au sens 1) c'_k si et seulement si les trois conditions suivantes sont vérifiées :

- $\tilde{g}c_k \geq \tilde{g}c'_k$ (c_k est plus long que c'_k),
- $dc_k \leq dc'_k$ (le parcours de c_k nécessite moins de temps que celui de c'_k, donc on peut dire que c_k utilise moins de ressources que c'_k si on considère le temps comme une ressource) et
- $S(c_k) \subseteq S(c'_k)$ (on a donc $X \backslash S(c'_k) \subseteq X \backslash S(c_k)$, ce qui signifie que si on peut étendre c'_k à un sommet l du graphe $G - l$ appartient nécessairement à $X \backslash S(c'_k)$ –, alors on peut également étendre c_k à l).

Définition d'un second critère de dominance

On peut renforcer la définition précédente de la dominance en considérant, non pas la liste $S(c_k)$ des sommets visités par un chemin partiel c_k, mais la liste $NA(c_k)$ des sommets "non-accessibles" pour c_k. Dans [Feillet *et al.* 03], un nouveau critère de dominance utilisant cette notion de sommets non-accessibles est présenté et utilisé pour la résolution par programmation dynamique d'un problème de plus court chemin élémentaire avec contraintes de ressources. Les auteurs définissent la notion de sommet non-accessible et montrent que le critère de dominance qui en découle permet d'établir potentiellement un plus grand nombre de relations d'ordre "dominé/dominant" entre les chemins partiels élémentaires d'un graphe.

Nous reprenons ici les définitions de [Feillet *et al.* 03], pour le problème qui nous concerne. Soit c_k un chemin partiel, un sommet l de X est *non-accessible* pour c_k s'il a déjà été visité par c_k ($l \in S(c_k)$), ou bien s'il n'est plus possible (compte tenu de la fenêtre temporelle $[est_l, lst_l]$) de visiter l après avoir visité k, le dernier sommet de c_k, à la date dc_k ($lst_l < dc_k + d_{k,l}$).

On appelle $NA(c_k)$ l'ensemble des sommets non-accessibles pour c_k, on a donc :

$$NA(c_k) = S(c_k) \cup \{l \in X \mid lst_l < dc_k + d_{k,l}\}$$

Nous définissons ainsi le second critère de dominance : soit c_k et c'_k deux chemins élémentaires partiels du sommet o à un sommet k de G, caractérisés par les étiquettes $(k,\, \tilde{g}c_k,\, dc_k)$ et $(k,\, \tilde{g}c'_k,\, dc'_k)$; c_k domine (au sens 2) c'_k si et seulement si les trois conditions suivantes sont vérifiées :

- $\tilde{g}c_k \geq \tilde{g}c'_k$,
- $dc_k \leq dc'_k$ et

– $NA(c_k) \subseteq NA(c'_k)$.

Afin d'illustrer la différence entre les deux critères de dominance, reprenons le graphe G de la figure 7.1 auquel on ajoute un sommet fictif initial o et considérons les chemins élémentaires partiels c_1 et c'_1 de o à 1 où $S(c_1) = \{o,2,1\}$ et $S(c'_1) = \{o,3,1\}$. Les étiquettes E^{c_1} et $E^{c'_1}$ associées sont :

$$E^{c_1} = (1,\, \tilde{g}_2 + \tilde{g}_1,\, 9) \quad \text{et} \quad E^{c'_1} = (1,\, \tilde{g}_3 + \tilde{g}_1,\, 11)$$

Supposons $\tilde{g}_2 > \tilde{g}_3$, on a alors $\tilde{g}c > \tilde{g}c'$ et $dc_1 < dc'_1$, mais $S(c_1) \not\subseteq S(c'_1)$; il n'y a donc pas de relation de dominance au sens 1 entre c_1 et c'_1. En revanche, les ensembles des sommets non accessibles sont :

$$NA(c_1) = \{o,1,2,3,4\} \quad \text{et} \quad NA(c'_1) = \{o,1,2,3,4,5\}$$

On a donc $NA(c_1) \subseteq NA(c'_1)$ et c_1 domine donc c'_1 au sens 2.

Description de l'algorithme PLPSE

À partir d'un ensemble Ep de chemins partiels élémentaires, l'algorithme de résolution de SP consiste à trouver de nouveaux plus longs chemins partiels élémentaires qui étendent les chemins de Ep à des sommets non encore visités, jusqu'à atteindre le sommet final f. Notons que par construction du sommet f dans le graphe G, quand un chemin visite le sommet f, alors f est nécessairement le dernier sommet de ce chemin ; inversement, tout chemin ne contenant pas f peut être étendu à f. Quand tous les chemins de Ep atteignent le sommet f (*i.e., Ep* ne contient plus de chemin "partiel"), le chemin de Ep le plus long donne la solution du problème SP, c'est-à-dire la pseudo-séquence de prises de vue de plus grand profit marginal $\tilde{g}$. Afin de ne pas considérer la totalité des chemins partiels élémentaires, seul les chemins dominants, au sens 1 ou au sens 2, présentés ci-avant, sont conservés dans l'ensemble E tout au long de l'algorithme.

L'algorithme 2, nommé PLPSE($\tilde{g}$), détaille cette procédure de résolution de SP. PLPSE prend en paramètre le vecteur $\tilde{g}$, des profits marginaux des prises de vues, car ce sont ces valeurs qui caractérisent SP au cours du processus de génération de colonnes.

L'ensemble Ep est initialisé avec le chemin nul c_o ($S(c_o) = \{o\}$) associé à l'étiquette $(o, 0, 0)$ (ligne 1) et la première étiquette traitée (étiquette courante) est $(o, 0, 0)$ (ligne 2).

À chaque itération, l'étiquette courante $E^{c_k} = (k, \tilde{g}c_k, dc_k)$, associée au chemin partiel élémentaire c_k, est marquée dans l'ensemble Ep (ligne 4) de façon à ne plus être traitée par la suite. L'ensemble $Sc(k)$ des successeurs possibles du sommet k est mis à jour en fonction de la valeur dc_k (ligne 5) grâce à des règles de filtrage des données sur lesquelles nous revenons par la suite. Le chemin courant est étendu à chacun de ses successeurs, créant ainsi autant de nouvelles étiquettes E^{c_l} avec $\tilde{g}c_l = \tilde{g}c_k + \tilde{g}_l$ et $dc_l = \max(est_l, dc_k + d_{k,l})$ (ligne 7). Parmi ces nouvelles étiquettes, seules celles correspondant à des chemins partiels dominants (ligne 8) sont ajoutées à l'ensemble Ep, auquel on retire les étiquettes associées à d'éventuels chemins partiels désormais dominés (ligne 10). Lorsque le traitement de l'étiquette E^{c_k} est terminé, on choisit une nouvelle étiquette courante parmi les éléments non marqués de Ep (ligne 13). On peut choisir par exemple l'étiquette non marquée E^c de plus petite date dc ; nous revenons par la suite sur ce critère de choix. Lorsque Ep ne contient plus d'étiquette non marquée correspondant à un chemin partiel, l'étiquette courante E^{c_k} est telle que $k = f$ et le processus itératif s'arrête (ligne 3).

Algorithme 2 Algorithme PLPSE($\tilde{g}$) : recherche de la Plus Longue Pseudo-Séquence Élémentaire

1: $Ep \leftarrow \{(o, 0, 0)\}$
2: $(k, \tilde{g}c_k, dc_k) \leftarrow (o, 0, 0)$ *(étiquette courante)*
3: **Tant que** $(k \neq f)$ **faire**
4: `Marquer`$(Ep, (k, \tilde{g}c_k, dc_k))$
5: $Sc(k) \leftarrow$ `Successeurs` $((k, \tilde{g}c_k, dc_k))$
6: **Pour** tout $l \in Sc(k)$ **faire**
7: $(l, \tilde{g}c_l, dc_l) \leftarrow (l, \tilde{g}c_k + \tilde{g}_l, \max(est_l, dc_k + d_{k,l}))$ *(nouvelle étiquette)*
8: **Si** `Est_Dominant` $(Ep, (l, \tilde{g}c_l, dc_l))$ **faire**
9: $Ep \leftarrow E \cup \{(l, \tilde{g}c_l, dc_l)\}$
10: `Mise_À_Jour` (Ep)
11: **fin si**
12: **fin pour**
13: $(k, \tilde{g}c_k, dc_k) \leftarrow$ `Choix_Etiquette`(Ep)
14: **fin tant que**
15: **Retourner** `Plus_Long_Chemin`(Ep)

L'étiquette E^{c_k}, de plus grande valeur $\tilde{g}c_k$ parmi les éléments non marqués de Ep, donne alors le plus long chemin du graphe G et donc la pseudo-séquence de prises de vues de profit marginal maximum (ligne 15).

fonction `Successeurs`

La fonction `Successeurs` établit la liste $Sc(k)$ des sommets auxquels un chemin c_k, associé à une étiquette E^{c_k}, peut être étendu, c'est-à-dire, l'ensemble des successeurs possibles de k, pour le chemin partiel c_k.

Soit ω_k^+ l'ensemble des sommets de X tels que $(k,l) \in U$, $\forall\, l \in \omega_k^+$ dans le graphe G. Il est clair que l'on a $Sc(k) \subseteq \omega_k^+$. On peut déduire de la liste $S(c_k)$ des sommets de c_k, ainsi que de la date dc_k, un certain nombre de sommets de ω_k^+ qui peuvent être exclus de $Sc(k)$, ceci afin de réduire l'espace de recherche des chemins possibles.

D'une part, comme on cherche à construire uniquement des chemins élémentaires, un sommet appartenant déjà au chemin c_k ne peut pas être à nouveau visité. On a donc :

$$\forall\, l \in \omega_k^+,\, l \in S(c_k) \Longrightarrow l \notin Sc(k) \tag{44}$$

D'autre part, si $dc_k > est_k$, alors il peut exister des arcs $(k,l) \in U$ tels qu'il n'est plus possible de visiter l après avoir visité k à la date dc_k ($lst_l < dc_k + d_{k,l}$, l est non-accessible pour c_k) ; dans ce cas, l n'appartient pas à $Sc(k)$. On a donc :

$$\forall\, l \in \omega_k^+,\, (lst_l < dc_k + d_{k,l}) \Longrightarrow l \notin Sc(k) \tag{45}$$

Enfin, pour tout sommet $l \in \omega_k^+ \backslash S(c_k)$, s'il existe un autre sommet $m \in \omega_k^+ \backslash S(c_k)$ de gain $\tilde{g}_m$ positif qu'on peut, à partir de c_k, visiter avant l sans modifier la date d'exécution de la

prise de vue l, alors l peut être éliminé de l'ensemble des successeurs de k dans c_k car le chemin visitant les sommets $(S(c_k),\ l)$ sera nécessairement dominé par le chemin $(S(c_k),\ m,\ l)$. Ceci se traduit par la relation suivante :

$$\forall\,(m,l) \in (\omega_k^+ \backslash S(c_k))^2 \mid \tilde{g}_m > 0,\ \max(est_m, dc_k + d_{k,m}) + d_{m,l} \leq est_l \Longrightarrow l \notin Sc(k) \tag{46}$$

Cette règle d'élimination peut être coûteuse à réaliser dans le cadre de l'algorithme PLPSE car elle teste tous les couples de sommets de ω_k^+ et ce, pour chaque élément de E_p. Nous proposons une simplification de cette règle. Soit $\overline{d} = \max_{(k,l)\in U | l \neq f}(d_{k,l})$ la durée maximale des arcs du graphe G (ne contenant pas f). Notons que $\overline{d}$ est calculé une seule fois. On définit les ensembles ω_k^+1 et ω_k^+2 par :

$$\omega_k^+1 = \{l \in \omega_k^+ \backslash S(c_k) \mid est_l \ \leq\ dc_k + d_{kl} \text{ et } \tilde{g}_l > 0\}$$

et

$$\omega_k^+2 = \{l \in \omega_k^+ \backslash S(c_k) \mid est_l \ >\ dc_k + d_{kl}\}$$

Soit $\overline{dc} = \max_{l \in \omega_k^+1}(dc_k + d_{k,l})$ la date maximale dc_l des chemins étendus à partir de c_k à un sommet l de ω_k^+1. Si $\omega_k^+1 \neq \emptyset$, on pose alors la règle suivante :

$$\forall\, l \in \omega_k^+2,\ \overline{dc} + \overline{d} < est_l \Longrightarrow l \notin Sc(k) \tag{47}$$

Cette règle nécessite moins de tests que la précédente, elle reste néanmoins valide dans le sens où elle n'élimine pas de sommets susceptibles d'être des successeurs de k. En effet, soit $l \in \omega_k^+2$ tel que $\overline{dc} + \overline{d} < est_l$. On a alors la relation $dc_k + d_{k,m} + d_{m,l} < est_l,\ \forall\, m \in \omega_k^+1$. La définition de ω_k^+1 nous permet donc d'écrire : $\max(est_m, dc_k + d_{k,m}) + d_{m,l} < est_l,\ \forall\, m \in \omega_k^+1$. Il existe donc $m \in \omega_k^+ \backslash S(c_k)$ tel que $\tilde{g}_m > 0$ et $\max(est_m, dc_k + d_{k,m}) + d_{m,l} < est_l$, ce qui signifie que l'on peut toujours réaliser avantageusement la prise de vue m entre k et l et que l'on peut donc éliminer l des successeurs de k ($l \notin Sc(k)$).

Ainsi, pour chaque chemin partiel exploré par l'algorithme PLPSE, la fonction `Successeurs` met en œuvre les règles (44), (45) et (47) pour déterminer l'ensemble de ses extensions possibles.

fonction `Choix_Etiquette`

La fonction `Choix_Etiquette` choisit, à chaque itération, la prochaine étiquette (le prochain chemin), non marquée, dans l'ensemble Ep, à traiter. Le processus itératif de l'algorithme s'arrête lorsque `choix_Etiquette` retourne une étiquette E^{c_f} correspondant à un chemin entier de Ep (*i.e.,* le dernier sommet visité est f), ce qui signifie que tous les chemins partiels c_k, $k \neq f$ de Ep ont été traités et qu'il ne reste donc plus de possibilité de construire de meilleur chemin de o à f.

Si, globalement, tous les éléments de Ep doivent être traités, l'ordre dans lequel ces éléments sont traités (et donc le choix de la prochaine étiquette à chaque itération) est important. En effet, l'existence d'un critère de dominance entre les chemins permet de n'ajouter à Ep que les "meilleurs" chemins partiels et permet également d'éliminer certains éléments de Ep lorsqu'un nouveau chemin est ajouté. On a donc tout intérêt à générer le plus tôt possible de "bons" chemins partiels dans Ep si on veut limiter la taille de cet ensemble. Différentes stratégies de choix peuvent être retenues, nous en proposons plusieurs, basées sur des tests simples de comparaison. Tout d'abord, nous proposons de choisir la prochaine étiquette parmi

celles qui ont été ajoutées à Ep lors de l'itération courante, on tend donc à construire rapidement un chemin atteignant le sommet final f. Parmi ces nouvelles étiquettes, on peut choisir l'étiquette E^{c_k} :

- de plus petite date dc_k, ce qui revient à choisir le chemin partiel le plus "rapide" ;
- de plus petite "marge" de visite de k, $lst_k - dc_k$, ce qui revient à choisir le chemin partiel dans lequel le dernier sommet visité, k, aurait le moins de chance de pouvoir être visité plus tard ;
- de plus grand gain $\tilde{g}c_k$, ce qui revient à choisir le plus long chemin partiel ;
- de plus petite valeur $dc_k/\tilde{g}c_k$, ce qui revient à choisir le chemin partiel réalisant le "meilleur compromis" entre la date de visite du dernier sommet k et la longueur ;
- de plus petite valeur $(lst_k - dc_k)/\tilde{g}c_k$, ce qui revient à choisir le chemin partiel réalisant le meilleur compromis entre la marge de visite de k et la longueur.

Lorsqu'à l'itération courante, aucun nouveau chemin partiel n'a été ajouté à E_p, on étend alors le choix aux autres éléments de E_p, avec là encore, un des critères de sélection ci-dessus.

On pourrait adopter d'autres types de stratégies, utilisant des tests plus complexes, basées par exemple, sur l'observation du "regret" associé à un chemin partiel, qui correspond au gain des prises de vues qui ne peuvent plus être acquises à partir de ce chemin partiel. Cette notion de "moindre regret" a été introduite par [Little *et al.* 63] pour la résolution du problème de voyageur de commerce et est utilisée de façon plus générale en tant qu'heuristique d'instanciation de variables dans différents algorithmes de recherche arborescente. On la retrouve ainsi dans les stratégies mises en œuvre dans les systèmes de Programmation Par Contraintes, comme ILOG Solver ou CHIP [Le Pape 94, Wallace & Schimpf 02].

7.5.1.2 Chemins avec cycles autorisés, algorithme PLPSC

Nous avons vu qu'on peut, sans modifier la formulation du problème maître, relaxer la contrainte du sous-problème SP qui force à rechercher uniquement des chemins élémentaires.

Pour résoudre ce sous-problème, on peut reprendre l'algorithme PLPSE précédent en adaptant simplement la fonction `Successeurs` qui ne doit plus retirer à l'ensemble des successeurs possibles d'un chemin partiel c la liste $S(c)$ des sommets déjà visités par ce chemin. De plus, on fait appel à un critère de dominance beaucoup plus simple et classique dans la résolution de problèmes de plus courts chemins dans des graphes sans cycles. Soit c et c' deux chemins partiels du graphe G, associés aux étiquettes E^c et $E^{c'}$ définies au pragraphe précédent, c domine c' si et seulement si les deux conditions suivantes sont vérifiées :

- $\tilde{g}c \geq \tilde{g}c'$ et
- $dc \leq dc'$.

La relation d'ordre est ici largement renforcée puisqu'on ne compare plus, ni les sommets visités, ni les sommets non-accessibles ; en revanche l'espace de recherche de l'algorithme est élargi puisqu'on autorise la construction de chemins comportant des cycles et que, de toute évidence, le graphe G comporte des cycles de longueur positive. Cependant, on est assuré que l'algorithme converge encore, grâce aux fenêtres temporelles qui contraignent la visite de chaque sommet du graphe et qui empêchent ainsi la formation de cycles "infinis".

On nomme la variante ainsi obtenue de l'algorithme PLPSE, algorithme PLPSC (pour Plus Longue Pseudo-Séquence avec Cycles autorisés).

7.5.2 Résolution de la relaxation linéaire de PM

Pour résoudre la relaxation linéaire du problème maître PM, nous définissons dans l'algorithme 3, une procédure de génération de colonnes GC, qui prend en paramètre un ensemble initial de colonnes Ω_0 (dont nous discutons la construction par la suite) et qui fait appel à l'un des algorithmes précédemment présentés pour la résolution du sous-problème SP.

Algorithme 3 GC(Ω_0) : Génération de Colonnes pour la relaxation linéaire de PM

1: `Construction_Graphe`(SP)
2: **Pour** (tout sommet $k \in X$) **faire**
3: $\tilde{g}_k \leftarrow gain(|k/2|)$
4: $\Omega \leftarrow \Omega_0$
5: fin $\leftarrow 0$
6: **Répéter**
7: `Résoudre_R_L`(PMR_Ω)
8: **Pour** (tout sommet $k \in X$) **faire**
9: $\tilde{g}_k \leftarrow gain(|k/2|) - \pi_{|k/2|} - \lambda'_k$ *(mise-à-jour des profits marginaux)*
10: $n_{PS} \leftarrow$ PLPSE ($\tilde{g}$) *(nouvelle pseudo-séquence de prises de vues)*
11: **Si** ($\tilde{g}n_{PS} > \mu$) **faire**
12: $\Omega \leftarrow \Omega \cup \{$`colonne` $(n_{PS})\}$
13: **Sinon** fin $\leftarrow 1$
14: **Tant que** (non fin)

Tout d'abord, le graphe G, qui modélise le sous-problème SP (voir paragraphe (7.4)), est construit par la fonction `Construction_Graphe` (ligne 1), sur laquelle nous revenons par la suite. Les gains associés aux sommets du graphe sont ensuite initialisés (ligne 3), ainsi que l'ensemble courant Ω des colonnes considérées pour la résolution de PM (avec l'ensemble Ω_0, ligne 4).

Le processus itératif commence par la résolution de PMR_Ω, la restriction de PM à Ω, par un algorithme de programmation linéaire comme le simplexe dans (`Résoudre_R_L`(PMR_Ω), ligne 7). La solution obtenue permet de mettre à jour les gains des sommets du graphe G selon la définition, donnée au paragraphe (7.3), des profits marginaux associés aux prises de vue (ligne 9). Rappelons que π et λ' sont les vecteurs des valeurs duales des contraintes (39) à (41) de PM. Le sous-problème SP avec les gains ainsi modifiés est alors résolu. Dans l'algorithme 3, on fait appel à PLPSE (qui recherche des chemins élémentaires) pour résoudre SP. On peut, sans modification de GC, utiliser l'algorithme PLPSC (qui autorise les cycles dans les chemins de G). La variable n_{PS} reçoit donc le nouveau plus long chemin de G (*i.e.,* la nouvelle pseudo-séquence de plus grand profit marginal) (ligne 10).

Si le profit marginal $\tilde{g}n_{PS}$ du chemin n_{PS} est supérieur à μ, la valeur duale associée à la contrainte (42) de PM, alors n_{PS} est susceptible d'améliorer la solution de PM, on construit

donc la colonne correspondant à ce chemin (`colonne`(n_{PS}), ligne 12), on l'ajoute à l'ensemble Ω et on commence l'itération suivante en résolvant la relaxation du nouveau PMR_Ω. Si, au contraire, le profit marginal $\tilde{g}n_{PS}$ du chemin n_{PS} est inférieur ou égal à μ (ligne 13), nous avons vu au paragraphe (7.3) qu'il n'existe alors plus de colonne pouvant améliorer la solution de PM. Le processus itératif se termine et la solution courante de la relaxation de PMR_Ω fournit la solution pour PM.

7.5.2.1 Fonction `Construction_Graphe`(SP)

La fonction `Construction_Graphe` construit les ensembles X et U des sommets et des arcs du graphe G correspondant au sous problème SP. À chaque prise de vue k, on associe un sommet k dans X ainsi qu'une fenêtre temporelle $[est_k\ ,\ lst_k]$ comme défini au paragraphe (7.4).

Afin de réduire la taille du graphe ainsi généré, nous mettons en place les règles suivantes, permettant d'éliminer certains sommets du graphe d'une part et certains arcs d'autre part.

Pour tout couple de bandes jumelles j et $Tw(j)$, s'il est impossible de réaliser les deux prises de vue associées $2j$ et $2Tw(j)$ (ou $2j+1$ et $2Tw(j)+1$), l'une à la suite de l'autre, alors ces prises de vue ne pourront pas faire partie d'une solution optimale du problème. On peut donc exclure les sommets $2j$ et $2Tw(j)$ (ou $2j+1$ et $2Tw(j)+1$) de l'ensemble X. Ceci se traduit par les règles :

$$\begin{aligned}
&\forall\, j = 0..N_B - 1 \mid Tw(j) > j \\
&\quad (\ Tmin0(j) + Du(j) + Dt(2j, 2Tw(j)) > Tmax0(Tw(j)) \\
&\quad \text{et } (Tmin0(Tw(j)) + Du(Tw(j)) + Dt(2Tw(j), 2j) > Tmax0(j)\ ) \\
&\quad \Longrightarrow 2j\ ,\ 2Tw(j)\ \notin X \qquad (48) \\
&\quad (\ Tmin1(j) + Du(j) + Dt(2j+1, 2Tw(j)+1) > Tmax1(Tw(j)) \\
&\quad \text{et } (Tmin1(Tw(j)) + Du(Tw(j)) + Dt(2Tw(j)+1, 2j+1) > Tmax1(j)\ ) \\
&\quad \Longrightarrow 2j+1\ ,\ 2Tw(j)+1\ \notin X \qquad (49)
\end{aligned}$$

Nous avons vu au paragraphe 7.4 qu'à tout couple de sommets (k,l) dans X, on peut associer un arc (k,l) dans U s'il existe une possibilité de réaliser la prise de vue k puis la prise de vue l en respectant les fenêtres temporelles associées ($est_k + d_{k,l} \leq lst_l$). Or, ceci peut entraîner l'existence d'arcs (k,l) $\in U$ où est_l est suffisamment supérieure à lst_k pour qu'on puisse toujours réaliser une autre prise de vue m entre k et l. Dans ce cas, la "séquence" (k,l) n'appartiendra jamais à une solution optimale du problème. On peut donc exclure de U de tels arcs (k,l). Pour tout sommet k de X, posons :

$$\omega_k^+ 3\ =\ \{l \in X | est_k + d_{k,l} \leq lst_l \text{ et } lst_k - d_{l,k} \leq est_l\}$$

$\omega_k^+ 3$ est l'ensemble des prises de vues l telles qu'on peut réaliser k puis l mais pas l puis k. On a alors la relation suivante :

$$\begin{aligned}
&\forall\, k \in X\ ,\ \forall\, l \in \omega_k^+ 3 \mid (est_l > lst_k + d_{k,l}) \\
&(\ \exists\ m \in \omega_k^+ 3, \mid m \neq l \text{ et } (\max(est_m\ ,\ lst_k + d_{k,m}) + d_{m,l}) \leq est_l\ ) \Longrightarrow (k,l) \notin U \qquad (50)
\end{aligned}$$

Cette règle est très proche de la règle (46) présentée dans le cadre de l'algorithme PLPSE. La règle (46) a été simplifiée pour être intégrée à l'algorithme PLPSE du fait de sa complexité

jugée trop grande. Ici, la règle (50) nécessite également un grand nombre d'opérations (de l'ordre de N_B^3) mais, du fait qu'elle n'est appliquée qu'une fois dans le programme, on la conserve en l'état.

Ainsi, les règles (48) à (50) sont appliquées lors de la construction du graphe G afin d'obtenir un graphe de taille réduite pour représenter le sous-problème SP.

7.5.2.2 Construction de Ω_0

Dans la plupart des problèmes traités par génération de colonnes, la détermination d'un ensemble de colonnes initial, Ω_0, assurant l'existence d'une solution réalisable au problème maître, présente une première difficulté de la méthode. Dans notre cas, du fait qu'il n'y a aucun "service minimum" requis dans le problème Pléiades, on peut facilement fournir une solution réalisable (la solution consistant à ne sélectionner aucune prise de vue, par exemple) et donc cette difficulté ne se pose pas.

Cependant, la composition de l'ensemble initial influe sur le nombre d'itérations nécéssaires au processus de génération de colonnes pour converger. Même si on peut choisir un ensemble Ω_0 vide pour l'algorithme GC, il est donc préférable de générer un certain nombre de colonnes initiales.

Nous choisissons dans un premier temps de construire Ω_0 de façon simple, en associant une colonne à chaque bande candidate (ou couple de bandes stéréoscopiques jumelles, sous certaines conditions). En effet, pour chaque bande candidate j, la solution qui consiste à sélectionner uniquement la prise de vue $k = 2j$ (ou bien uniquement $k' = 2j+1$) est réalisable. De même, si j et $Tw(j)$ sont deux bandes candidates jumelles, la solution qui consiste à ne sélectionner que les prises de vue $k = 2j$ et $l = 2Tw(j)$ (ou bien $k' = 2j + 1$ et $l' = 2Tw(j) + 1$) est réalisable si $Tmin0(j) + Du(j) + Dt(k,l) < Tmax0(Tw(j))$ (ou bien si $Tmin1(j) + Du(j) + Dt(k',l') < Tmax1(Tw(j))$).

Nous générons également des colonnes initiales par des algorithmes gloutons qui sélectionnent une première prise de vue parmi celles de plus petite date de début au plus tôt et les suivantes selon un des critères de choix présentés précédemment pour la fonction `Choix_Etiquette` de l'algorithme PLPSE (prise de vue de plus petite date d'exécution, ou de plus petite marge, ou de plus grand profit associé, ou encore de plus petit rapport entre la date d'exécution et le gain, ou entre la marge et le gain).

Notons qu'il serait intéressant de faire appel à des méthodes approchées plus élaborées pour générer de meilleures colonnes initiales dans Ω_0, par exemple, à des algorithmes de recherche Tabou qui ont fourni de bons résultats au Challenge ROADEF'03.

Des méthodes approchées peuvent être également utilisées pour résoudre plus rapidement le sous-problème SP, à l'intérieur du processus de génération de colonnes. On peut par exemple, utiliser une variante de l'algorithme PLPSE (ou PLPSC) pour la résolution du sous-problème, dans laquelle le temps de résolution maximum est fixé. Cet algorithme modifié (que nous nommons PLPSE*) retourne éventuellement la solution exacte de SP (si la résolution se termine avant la limite de temps fixée), ou bien la meilleure pseudo-séquence trouvée dans le temps limité. On doit alors modifier la condition d'arrêt de l'algorithme GC(Ω_0). En effet, si la pseudo-séquence n_{PS} trouvée par l'algorithme PLPSE* a un profit marginal $\tilde{g}n_{PS} \leq \mu$, il est nécessaire de résoudre SP au moins une fois par un algorithme exact (PLPSE) pour prouver qu'il n'existe pas de meilleure solution que n_{PS}. Si PLPSE ne fournit pas de meilleure

solution, alors le processus s'arrête, sinon la solution trouvée par PLPSE est ajoutée à Ω, l'ensemble courant de colonnes, et on passe à l'itération suivante.

Enfin, rappelons que l'algorithme GC(Ω_0) résout optimalement la relaxation linéaire du problème maître PM. Il permet donc de calculer une borne supérieure de la solution optimale du problème Pléiades mais pas nécessairement une solution réalisable. Pour obtenir la solution optimale entière du problème, il faudrait intégrer cet algorithme dans une recherche arborescente de type *branch and price*.

7.6 Expérimentations

7.6.1 Modèle linéaire initial

Les premiers tests que nous avons réalisés sur le problème Pléiades ont consisté à résoudre la relaxation du modèle initial P, présenté au chapitre 5 (pages 87 à 90), par programmation linéaire.

Le modèle est implémenté en C++ avec les bibliothèques ILOG CONCERT 1.2 et compilé avec Visual C++ 6.0, sous Windows XP. Le solveur ILOG CPLEX 7.5 est utilisé pour résoudre la relaxation linéaire de P par l'algorithme du simplexe. Les tests ont été réalisés sur un PC Pentium 4 à 2GHz, avec 248Mo de RAM.

Les instances testées sont présentées pour partie dans l'article [Lemaître *et al.* 02] qui rapporte les résultats de la première étude du problème Pléiades, réalisée par l'ONERA. Ces instances constituent la "base A" de données qui ont été soumises par la suite pour la phase de qualification du challenge ROADEF'03. Elles se composent de deux instances de très petites tailles et de huit autres, de tailles moyennes à grandes (de 25 à 534 bandes candidates), représentatives de la demande attendue pour le début de vie du système Pléiades.

7.6.1.1 Calcul d'une valeur de N_{sup}

Pour chaque instance, il faut fixer le nombre maximal, N_{sup}, de prises de vue pouvant être sélectionnées dans une solution. Par défaut, on peut poser $N_{sup} = N_B$ puisqu'on sait qu'on ne peut pas sélectionner plus d'une fois chaque bande. Afin de réduire la combinatoire du problème traité, on cherche cependant à évaluer plus précisément la valeur de N_{sup}. Pour cela, nous proposons de procéder comme suit. Introduisons les notations suivantes :

- $Dt\uparrow$: la liste des durées de transition entre les prises de vue, triée par ordre croissant (on note $Dt\uparrow[i]$ le $i^{\text{ème}}$ élément de $Dt\uparrow$) ;
- $Du\uparrow$: la liste des durées d'acquisition des bandes, triée par ordre croissant (on note $Du\uparrow[i]$ le $i^{\text{ème}}$ élément de $Du\uparrow$) ;
- $\overline{Tmax}$: la plus grande date de début au plus tard des prises de vues ;
- $\underline{Tmin}$: la plus petite date de début au plus tôt des prises de vues.

Soit $nsup$ le nombre réel de prises de vue sélectionnées dans une séquence solution. La différence $(\overline{Tmax} - \underline{Tmin})$ donne l'intervalle maximum sur lequel les prises de vue en position 1 à $(nsup-1)$ dans la séquence peuvent être réalisées. La durée totale d'acquisition des bandes sur cet intervalle est nécessairement supérieure à la valeur $\sum_{i=0}^{nsup-1} Du\uparrow[i]$ et, de même, la durée cumulée des transitions entre les prises de vue sur cet intervalle est nécessairement supérieure à la valeur $\sum_{i=0}^{nsup-1} Dt\uparrow[i]$.

On a donc l'inégalité :

$$(\overline{Tmax} - \underline{Tmin}) - \sum_{i=0}^{nsup-1} (Du\uparrow[i] + Dt\uparrow[i]) \geq 0$$

et donc, pour tout entier n, la relation d'implication :

$$(\overline{Tmax} - \underline{Tmin}) - \sum_{i=0}^{n-1} (Du\uparrow[i] + Dt\uparrow[i]) \leq 0 \implies n \geq nsup$$

Nous choisissons donc, comme valeur de N_{sup}, le plus petit entier n qui vérifie cette inégalité (ou N_B si $N_B < n$).

7.6.1.2 Bornes supérieures obtenues par programmation linéaire

Le tableau 7.2 donne une synthèse des principales données des instances testées. Il indique pour chacune d'entre-elles le nombre de requêtes et de bandes candidates, le nombre de requêtes et de bandes stéréoscopiques, ainsi que l'intervalle $(\overline{Tmax} - \underline{Tmin})$ (en secondes, arrondi à l'entier supérieur) défini ci-avant et la valeur de N_{sup}, calculée comme nous venons de le voir. Le nom des instances est composé de trois nombres qui correspondent à des paramètres liés à la position orbitale du satellite et à la base de données des demandes.

Instance	nombre requêtes (N_R)	nombre bandes (N_B)	nombre requêtes stéréo	nombre bandes stéréo	$(\overline{Tmax} - \underline{Tmin})$	N_{sup}
2_9_36	2	2	0	0	114	2
2_9_66	4	7	0	0	272	7
2_9_170	12	25	2	10	633	25
3_8_155	12	28	9	20	265	28
2_13_111	68	106	12	24	624	89
4_17_186	77	147	40	104	1065	123
2_15_170	218	295	39	78	606	94
3_25_22	150	342	87	236	549	82
2_26_96	336	483	126	55	576	92
2_27_22	375	534	63	138	567	89

TAB. 7.2 – *Description des instances testées*

La résolution de ces instances par programmation linéaire nous a permis d'obtenir les bornes supérieures présentées dans le tableau 7.3. Pour chaque instance, les deux premières colonnes donnent la solution du modèle P relâché et le temps CPU de calcul (en secondes). À titre de comparaison, la dernière colonne du tableau indique les résultats présentés lors du challenge ROADEF'03 par [Vasquez *et al.* 03], qui calcule des bornes supérieures du problème Pléiades en considérant un critère de maximisation linéaire identique au critère (36) du modèle P. Les résultats marqués d'une étoile indiquent que la borne a été prouvée être également la solution optimale du problème en variables mixtes. Pour les instances 2_26_96 et 2_27_22, notons que la relaxation des seules contraintes d'intégrité du problème ne permet pas de trouver une borne. En effet, le programme linéaire ainsi généré est trop grand pour être traité par CPLEX sur la machine utilisée. Pour ces deux instances, nous avons été amené à relâcher

également les contraintes de durées de transition entre les prises de vue, en éliminant les composantes $Dt(.,.)$ des contraintes (29) et (30) de P. Ceci permet de réduire suffisamment le nombre de contraintes pour obtenir une borne.

Instance	borne sup.	temps CPU (secondes)	borne sup. [Vasquez *et al.* 03]
2_9_36	$1.04234.10^8$(*)	0.75	$1.04234.10^8$(*)
2_9_66	$1.15711.10^8$(*)	0.05	$1.15711.10^8$(*)
2_9_170	$4.63949.10^8$	0.38	$1.91358.10^8$(*)
3_8_155	$3.41113.10^8$	0.64	$1.21680.10^8$(*)
2_13_111	$1.16543.10^9$	34	$7.50675.10^8$
4_17_186	$6.48827.10^8$	312	$2.33270.10^8$
2_15_170	$1.36837.10^9$	635	$1.00758.10^9$
3_25_22	$1.68483.10^9$	1671	$5.95724.10^8$
2_26_96	$2.52505.10^9$	62	$1.26219.10^9$
2_27_22	$2.64702.10^9$	1143	$1.22587.10^9$

TAB. 7.3 – *Résultats de la relaxation linéaire de P*

Comme on pouvait le pressentir, en particulier du fait de la relaxation des contraintes d'intégrité sur des variables booléennes doublement indicées ainsi que le recours à des paramètres de type "grand M" dans les contraintes (29) et (30), les bornes obtenues par relaxation linéaire du modèle P sont de médiocre qualité.

Par la suite, la reformulation du modèle P en PM et l'utilisation de techniques de génération de colonnes pour sa résolution ont pour but d'améliorer ces bornes. Avant cela, nous présentons au paragraphe suivant des résultats concernant la génération de séquences solutions qui ont étés proposés pour le Challenge ROADEF'03.

7.6.2 Génération de séquences solutions de prises de vue

Dans le contexte du Challenge ROADEF'03, nous avons testé une adaptation de l'algorithme PLPSE sur les différentes instances soumises.

Afin de satisfaire aux différentes contraintes de ce challenge, nous avons apporté trois principales modifications à l'algorithme PLPSE. D'une part, nous avons modifié le mode de calcul du gain d'une séquence de prises de vue de façon à prendre en compte le critère non linéaire (critère (35), voir chapitre 5, page 90) du problème Pléiades. Le gain total d'une séquence solution est ainsi la somme des gains associés aux fractions acquises des requêtes candidates, pondérées non-linéairement. Notons que l'application du critère de dominance peut alors entrainer l'élimination de "bons" chemins partiels dans le graphe G et donc l'élimination de "bonnes" séquences de prises de vue. D'autre part, afin de prendre en compte la contrainte stéréoscopique du problème, nous retirons de chaque séquence solution trouvée, les éventuelles prises de vue stéréoscopiques dont la "jumelle" ne se trouve pas dans la séquence. Enfin, on limite le temps de recherche à 300 secondes.

Ces différentes modifications font de cet algorihtme PLPSE modifié une méthode approchée pour le problème Pléiades, qui n'est naturellement pas comptétitive avec les meilleures heuristiques fournies lors du challenge (algorithmes Tabou, Recuits Simulés, voir chapitre 5, page 87) sur les grandes instances, mais qui nous a permis de valider le fonctionnement de l'algorithme PLPSE. Le tableau 7.4 présente les résultats ainsi obtenus avec, pour comparai-

son, les résultats trouvés par le vainqueur du Challenge ROADEF'03 (par un algorithme de Recuit Simulé [Kuipers 03]).

Instance	solution (gain total)	[Kuipers 03]
2_9_36	$1.04234.10^8$	$1.04234.10^8$
2_9_66	$1.15711.10^8$	$1.15711.10^8$
2_9_170	$1.91358.10^8$	$1.91358.10^8$
3_8_155	$9.75361.10^7$	$1.21680.10^8$
2_13_111	$5.49226.10^8$	$5.63597.10^8$
4_17_186	$1.65615.10^8$	$1.85407.10^8$
2_15_170	$4.40836.10^8$	$7.16818.10^8$
3_25_22	$3.67666.10^8$	$4.25970.10^8$
2_26_96	$5.49087.10^8$	$1.00443.10^9$
2_27_22	$4.43815.10^8$	$9.65415.10^8$

TAB. 7.4 – *Solutions obtenues par PLPSE modifié pour le problème Pléiades non linéaire*

Nous avons testé les différentes stratégies de choix de "prochaine étiquette" introduites page 140 pour la fonction `Choix_Etiquette`. Les résultats obtenus diffèrent peu sensiblement d'une stratégie à l'autre, pour les tests réalisés dans le contexte particulier du challenge. Cependant, nous avons choisi de conserver pour la suite la quatrième stratégie, qui consiste à choisir l'étiquette E^{c_k} de plus petit rapport $(lst_k - dc_k)/\tilde{g}c_k$ et qui, en moyenne, donne les meilleurs résultats.

7.6.3 Procédure de génération de colonnes

7.6.3.1 Prétraitement des données : réduction du graphe G

Instance	nombre de sommets	sommets éliminés	nombre d'arcs	arcs éliminés	pourcentage d'arcs éliminés
2_9_36	6	0	24	0	0%
2_9_66	16	0	168	28	16%
2_9_170	52	0	1512	753	49%
3_8_155	58	0	1693	940	55%
2_13_111	214	0	26917	15310	56%
4_17_186	296	0	41815	30104	71%
2_15_170	592	0	212912	96798	45%
3_25_22	686	2	231267	138789	60%
2_26_96	968	0	553228	264792	47%
2_27_22	1070	0	714020	267262	37%

TAB. 7.5 – *Prétraitement des données pour le graphe G*

Nous évaluons tout d'abord l'efficacité des techniques de filtrage des données utilisées dans la fonction `Construction_Graphe`(SP), qui ont pour but de réduire la taille du graphe représentant le sous-problème SP. Le tableau 7.5 indique, pour chaque instance, le nombre initial de sommets et le nombre de sommets éliminés par l'application des règles (48) et (49), dans les deux premières colonnes, ainsi que le nombre initial d'arcs, le nombre et le pourcentage d'arcs éliminés par l'application de la règle (50), dans les trois dernières colonnes.

On remarque que les règles d'élimination des sommets correspondant à des prises de vue stéréoscopiques non réalisables sont inefficaces puisqu'elles ne permettent de supprimer qu'une paire de sommets, pour l'instance 3_25_22. Même si on pouvait espérer supprimer un nombre plus significatif de sommets par l'application de ces règles, ce test nous permet de vérifier la consistance des données concernant la réalisabilité des bandes stéréoscopiques.

En revanche, la règle de suppression d'arcs permet de réduire considérablement la densité du graphe puisque son application entraîne la suppression de 37% à 71 % des arcs pour les instances de taille moyenne à grande.

7.6.3.2 Procédure GC(Ω_0)

Nous avons pu valider la procédure GC(Ω_0) sur les plus petites instances, malheureusement, sur les intances de tailles moyennes ou grandes, nous sommes confronté à des problèmes de dépassement mémoire pour la résolution du sous-problème avec l'algorithme PLPSE.

Nous voyons actuellement plusieurs raisons à cela. Tout d'abord, le sous-problème SP considéré est de grande taille ; il comporte effectivement un nombre de nœuds supérieur à ce qui est généralement rapporté dans la littérature concernant la résolution de problèmes par génération de colonnes (environ une centaine de noeuds au maximum lorsque les graphes contiennent des cycles). On peut observer dans le tableau 7.5 que le graphe G compte de 52 à 1070 noeuds pour les instances 3_8_155 à 2_27_22.

De plus, il est à noter que la spécificité des données dans les instances considérées rend le problème de recherche de chemins partiels dominants dans le graphe G particulièrement combinatoire. En effet, on constate d'une part, que la durée des fenêtres d'exécution d'une prise de vue (de l'ordre de quelques minutes) est toujours grande en comparaison avec sa durée de réalisation (cinq secondes en moyenne) et avec sa durée de transition vers une autre prise de vue (moins de vingt secondes en général). D'autre part, les requêtes de prises de vue sont concentrées sur un intervalle temporel relativement restreint (de 265 à 1065 secondes, voir tableau 7.2) en regard de l'horizon temporel réel sur lequel on planifie les prises de vue (rappelons que l'horizon de planification correspond à une demi-révolution du satellite autour de la Terre, soit environ 3000 secondes). Ainsi, les fenêtres d'exécution des prises de vue candidates se chevauchent largement les unes les autres et il existe un grand nombre de combinaisons de choix de sélections de prises de vues possible. L'espace de recherche du sous-problème SP est donc très grand et les solutions de SP peuvent comporter un grand nombre de sommets.

Enfin, la règle de dominance utilisée dans l'algorithme PLPSE, qui interdit les chemins comportant des cycles, compare les ensembles de sommets non accessibles entre les chemins partiels. Cette comparaison est non seulement coûteuse en temps de calcul, mais elle affaiblit la règle de dominance qui identifie ainsi très peu de relations dominant/dominé entre les chemins partiels du graphe. L'ensemble des chemins partiels dominants générés par PLPSE est donc très grand.

Nous avons également testé la procédure GC(Ω_0) en faisant appel à l'algorithme PLPSC, qui autorise la présence de cycles dans les chemins pour la résolution du sous-problème. Le processus itératif est sensiblement plus rapide sur les instances de taille moyenne 3_8_155 et 2_9_170, mais la procédure converge dans des temps totalement prohibitifs. Pour les instances de plus grande taille, comme dans les tests précédents, on aboutit très rapidement

à un dépassement de la mémoire.

La rapidité gagnée à chaque itération est due au fait que l'algorithme PLPSC applique une règle de dominance plus forte que celle utilisée par l'algorithme PLPSE. Il y a moins de chemins partiels dominants et ils sont plus rapidement identifiés.

En revanche, compte tenu de la spécificité des données évoquée ci-dessus, le fait d'autoriser les cycles dans les chemins de G tend à créer un grand nombre de 2-cycles qui correspondent à des couples (k,l) de prises de vue de fort gain associé, dont les fenêtres d'exécution sont quasiment identiques et dont la durée $d_{k,l}$ est petite. Ainsi, le sous-problème génère des colonnes ne permettant pas d'améliorer suffisamment la solution du problème maître à chaque itération.

Notons enfin que les valeurs duales du problème maître évoluent très lentement au fil des itérations, ce qui ne permet pas à l'algorithme de résolution du sous-problème d'être bien "guidé". Le sous-problème à résoudre varie en effet très peu d'une itération à l'autre, ce qui conduit l'algorithme PLPSE ou PLPSC à générer des colonnes quasiment identiques d'une itération à l'autre.

7.7 Conclusions et perspectives

Dans ce chapitre, nous proposons une approche par génération de colonnes pour le problème Pléiades.

Nous décomposons le modèle P en un problème maître dans lequel les contraintes stéréoscopiques sont conservées et en un sous-problème consistant à rechercher des "pseudo-séquences" de prises de vues et qui s'exprime comme un problème de plus longs chemins dans un graphe avec fenêtres temporelles.

Nous décrivons ensuite une procédure de génération de colonnes pour le calcul de bornes supérieures du problème Pléiades. Deux algorithmes exacts sont développés pour la résolution du sous-problème, basés sur la résolution d'un ESPPTW (*Elementary Shortest Path Problem with Time Windows*).

Nous validons notre approche sur des instances de petite taille mais nous sommes rapidement confrontés à une explosion combinatoire du sous-problème. Il reste donc plusieurs améliorations à apporter à la méthode pour qu'elle fonctionne de façon satisfaisante. Trois voies nous semblent prioritaires dans cette optique.

D'une part, afin de réduire la taille du sous-problème considéré, il serait utile d'en proposer une décomposition efficace en plusieurs sous-problèmes indépendants. Notons à ce sujet, que nous avons tenté d'appliquer la méthode de décomposition temporelle proposée dans [Baptiste *et al.* 03], pour des problèmes d'ordonnancement à une machine où l'objectif est de minimiser le nombre de tâches en retard. Cette méthode consiste à rechercher, dans l'horizon considéré, des instants t qui n'appartiennent à aucune fenêtre temporelle et qui permettent donc de séparer les tâches à réaliser en deux groupes indépendants (tâches se terminant avant t et tâches débutant après t). On peut adapter cette méthode au problème Pléiades, en tenant compte des durées de transition, mais, en pratique sur les instances testées, cette méthode ne permet pas de décomposer le problème du fait des trop nombreux chevauchements des fenêtres entre-elles. On peut cependant penser à d'autres méthodes de décomposition temporelle, réalisant par exemple, non pas une partition, mais un recouvrement des fenêtres d'exécution des prises de vue, comme cela est proposé dans [Lebbar 00, Levy 96] pour le problème Skybridge.

Par ailleurs, il apparaît que l'algorithme de programmation dynamique mis en œuvre à chaque itération n'est pas assez efficace. Nous allons donc étudier la possibilité de développer un meilleur algorithme pour la résolution du sous-problème. Nous envisageons également de faire appel à des méthodes approchées de bonne qualité pour la résolution du sous-problème, ce qui devrait permettre de réduire sensiblement la durée de calcul à chaque itération du processus. Les méthodes proposées dans le cadre du challenge ROADEF'03 pour la résolution du problème Pléiades pourraient peut-être être intégrées bénéfiquement à notre approche du problème.

Enfin, nous pensons que le développement informatique de nos algorithmes pourrait être amélioré, les rendant ainsi plus performants. En effet, nos travaux ont porté essentiellement sur des aspects de modélisation des problèmes et de méthodologie de résolution, mais nous nous sommes peu penchés sur les détails d'implémentation. Cependant, des techniques pour l'optimisation de l'implémentation d'algorithmes existent et font l'objet de travaux de recherche, comme par exemple [Desaulniers *et al.* 99] pour le cas des algorithmes de génération de colonnes. Il nous semble donc important d'étudier les possibilités d'intégrer de telles considérations dans nos travaux.

Par la suite, notre but est d'intégrer cette procédure de génération de colonnes dans un processus de recherche de solutions entières pour le problème Pléiades. En particulier, une recherche arborescente de type *branch and price*, qui utilise la génération de colonnes comme procédure d'évaluation à chaque nœud, permettrait de construire une méthode exacte pour le problème considéré. Si, dans le cadre d'une planification "en ligne" d'un satellite d'observation, il ne semble pas raisonnable ou réellement utile de rechercher une solution optimale au problème, ce type d'étude serait toutefois intéressant pour l'évaluation de solutions approchées sur des instances de test, ou pour la génération de solutions pour des demi-révolutions particulières, sur lesquelles, par exemple, la demande est connue suffisamment à l'avance. De plus, cela permet d'explorer l'adéquation des techniques de génération de colonnes et de *branch and price* à des problèmes sortant de leur champ classique d'application.

Conclusion

La planification des missions spatiales nécessite d'importants efforts de recherche dans de nombreux domaines. Nous présentons en début de ce mémoire un tour d'horizon des principaux problèmes abordés et des méthodologies adoptées pour les modéliser et les résoudre. Notre travail s'est focalisé sur l'étude de trois projets : l'allocation des liens de communication Terre/satellites dans la constellation Skybridge, la planification des expériences et des communications de la mission Netlander, ainsi que la planification des prises de vue d'un satellite Pléiades. Nous nous sommes ainsi intéressé aux problèmes d'optimisation combinatoire sous-jacents, en privilégiant des méthodes exploitant les techniques générales que sont la Programmation Linéaire et la Propagation de Contraintes.

Nos travaux ont porté principalement sur les problèmes issus des projets Netlander et Pléiades. Netlander est un projet d'exploration de Mars par un ensemble de quatre sondes posées à sa surface et réalisant différents types d'expériences. Dans cette mission, il se pose en particulier le problème de la planification des expériences, ainsi que des communications des sondes avec un orbiteur martien qui assure le relais des échanges d'informations avec la Terre. Le projet Netlander est en tout début de vie ; aussi, ce problème n'a fait l'objet d'aucune étude préalable et l'effort de formalisation que nous avons fourni a été important. Nous proposons une décomposition en deux sous-problèmes indépendants. D'une part, nous modélisons et résolvons le problème de planification des communications orbiteur/sondes par Programmation Linéaire en Nombres Entiers. Nous proposons un modèle en variables booléennes, basé sur une discrétisation de l'horizon temporel de planification ; l'utilisation d'un outil commercial de PLNE a permis d'en résoudre efficacement des instances de taille réelle. Nous considérons également un cas généralisé dans lequel plusieurs orbiteurs martiens peuvent être utilisés pour communiquer avec les sondes. Nous adaptons le modèle pour prendre en compte les contraintes liées à cette modification et nous montrons que la PLNE peut encore fournir des résultats satisfaisants, avec cependant une limite sur les horizons de planification. D'autre part, nous proposons une approche d'aide à la décision pour la planification des expériences des sondes. En effet, dans ce second sous-problème, pour différentes raisons (liées aux connaissances actuelles, partielles, des caractéristiques physiques et fonctionnelles du système et aux intérêts divergents des différents acteurs engagés dans la mission), une résolution automatique ne semble pas envisageable ou utile actuellement. Ainsi, nous développons une méthode de construction de courbes d'évaluation de la charge des ressources critiques des sondes (place mémoire et énergie électrique) sur un horizon de planification. Ces évaluations sont déduites de l'application de techniques de Propagation de Contraintes basées sur un raisonnement énergétique, permettant de calculer des bornes supérieures et inférieures de la consommation d'une ressource par intervalle de temps. Au stade de l'avant projet, cette approche permet aux scientifiques d'évaluer la faisabilité de solutions types.

Pour poursuivre ces travaux, en proposant d'autres fonctionnalités d'aide à la décision (concernant la gestion des différents instruments de mesure des sondes, la synchronisation des quatres sondes, ...), en affinant celles présentées ici, ou encore en cherchant des stratégies d'élaboration de solutions, il nous semble nécessaire que le CNES soit en mesure de mieux caractériser l'ensemble des expériences susceptibles d'être réalisées, ainsi que les contraintes qui leur sont liées, et de définir plus précisément les processus de négociation qui interviendront pour l'élaboration des plans de travail des sondes.

Le projet Pléiades s'inscrit dans la continuité du programme d'observation de la Terre mené par le CNES (actuellement les satellites SPOT), avec l'exploitation de satellites de nouvelle génération, dits "agiles". Le problème soulevé ici consiste à sélectioner et ordonnancer les prises de vue d'un de ces satellites sur un horizon correspondant à une demi-révolution autour de la Terre, de façon à maximiser un gain associé à la réalisation des prises de vues. Ce problème est soumis à des contraintes fortes de ressource et de temps caractéristiques des problèmes de planification dans les missions spatiales, mais également à des contraintes logiques particulièrement complexes. Nous modélisons ce problème par un programme linéaire en variables mixtes et nous montrons qu'il appartient à la classe des problèmes NP-difficiles. Les méthodes actuellement envisagées pour la recherche de solutions au problème Pléiades font appel à des heuristiques. Dans cette étude, nous proposons une approche de résolution de ce problème par génération de colonnes. Le but est d'abord de pouvoir calculer des bornes supérieures de bonnes qualité, permettant par exemple l'évaluation des heuristiques proposées. On peut également envisager d'utiliser cette méthode dans des processus, exacts ou approchés, de recherche de solutions. La méthode se base sur une décomposition du modèle en, d'une part, un programme linéaire, appelé problème maître et, d'autre part, en un sous-problème qui consiste à générer les colonnes du problème maître. Dans notre cas, les colonnes du problème maître correspondent à des "pseudo-séquences" de prises de vue, qui vérifient toutes les contraintes temporelles du problème mais pas nécessairement certaines contraintes logiques, qui sont laissées dans le problème maître. La recherche des pseudo-séquences de prises de vue peut se ramener à celle de plus longs chemins dans un graphe avec fenêtres temporelles. Nous avons développé deux algorithmes de programmation dynamique pour la recherche des pseudo-séquences et nous décrivons le processus itératif de résolution du problème maître relâché.

Cette méthode nécessite encore plusieurs améliorations pour être satisfaisante. Nous envisageons différentes pistes pour renforcer l'efficacité de la recherche de pseudo-séquences, avec en particulier la possibilité de décomposer le problème global de façon à obtenir plusieurs sous-problèmes de moindre combinatoire. Une autre voie d'amélioration nous semble envisageable dans la gestion des contraintes du problème maître : d'une part, par la recherche de contraintes agrégées, par exemple en étudiant le problème d'un point de vue de ses "requêtes" (ensembles de prises de vue candidates) et en appliquant des techniques de propagation de contraintes, de façon à renforcer le modèle ; d'autre part, en adoptant une démarche par génération de contraintes, c'est-à-dire en incluant les contraintes du problème maître seulement lorsqu'elles sont nécessaires, de façon à réduire la taille de la matrice engendrée par le programme linéaire.

Une leçon que nous pouvons certainement tirer de notre travail est que face à la grande complexité des systèmes spatiaux, il n'est pas évident de trouver "la" bonne démarche. Nous avons vu par exemple que, bien que plusieurs approches soient généralement envisageables, la décomposition d'un problème ne s'impose pas toujours de façon naturelle. Ainsi, une autre perspective de travail pour la résolution du problème Pléiades, consiste à l'aborder d'un point

de vue différent, en le considérant par exemple comme un problème d'ordonnancement à une machine avec temps de préparation et dans lequel on souhaite minimiser une somme pondérée des tâches en retard. Ceci fait référence à certains types de problèmes faisant l'objet de travaux de recherche spécifiques en ordonnancement. Des méthodes de calcul de bornes, par relaxation lagrangienne et Programmation par Contraintes par exemple, sont proposées, ainsi que des algorithmes pour la recherche de solutions. Il serait intéressant de voir dans quelle mesure ces approches pourraient être appliquées au cas du problème Pléiades, de façon indépendante sur le problème global, ou bien intégrées dans un processus basé sur une décomposition, comme la génération de colonnes.

Enfin, une troisième perspective de travail concerne l'intégration du calcul de bornes dans un processus de recherche de solutions entières pour le problème Pléiades. En particulier, une méthode arborescente de type *branch and price*, qui utilise la Génération de Colonnes pour évaluer chaque nœud de l'arbre de recherche, permet de garantir l'optimalité de la solution entière obtenue. Naturellement, ce type de méthodes peut présenter des difficultés, liées en particulier aux temps de calcul nécessaires. La Génération de Colonnes peut également être employée dans des procédures heuristiques et ce à deux niveaux : en se limitant à une résolution approchée du sous-problème (donnant ainsi une évaluation du problème relaché, et non plus une borne) ou en utilisant la borne – ou l'évaluation – ainsi calculée dans une méthode heuristique de recherche de solutions entières (métaheuristique, algorithme *ad hoc*, recherche arborescente tronquée, ...).

À plus long terme, nous souhaitons pouvoir étendre l'application des méthodes de résolution mises en œuvre pour le problème Pléiades à d'autres problèmes. D'une part dans le domaine des applications spatiales, en traitant par exemple des instances multi-orbiteurs du problème de planification des communications du projet Netlander. D'autre part, dans d'autres domaines, en considérant par exemple des problèmes d'ordonnancement de projet ou d'ateliers comportant des contraintes de fenêtres temporelles d'exécution et de temps de préparation entre les tâches. Il nous semble en effet que les approches développées ici s'inscrivent dans la lignée d'un certain nombre de travaux récents, qui mettent en évidence l'utilité des méthodes basées sur une coopération de méthodes exactes comme la PLNE avec des techniques de réduction ou simplification de l'espace de recherche (décomposition des modèles, techniques de Propagation de Contraintes), pour aborder les problèmes d'optimisation combinatoire de grande taille [Chabrier 99, Lebbar 00, Demassey 03, Oliva 04, Danna 04].

Bibliographie

[Aktur & Kiliç 99] M.S. Aktur & K. Kiliç. *Generating short-term observation schedules for space mission projects.* Journal of Intelligent Manufacturing, vol. 10, pages 387–404, 1999.

[Alliot & Schiex 94] J.M. Alliot & T. Schiex. Intelligence artificielle et informatique théorique. Cépaduès éditions, Toulouse, 1994.

[Applegate & Cook 91] D. Applegate & W. Cook. *A computational study of the job-shop scheduling problem.* ORSA Journal on Computing, vol. 3, no. 2, pages 149–156, 1991.

[Baptiste & Le Pape 96] P. Baptiste & C. Le Pape. *Edge-finding constraint propagation algorithms for disjunctive and cumulative scheduling.* In Proc. of U.K. planning and scheduling SIG meeting, Liverpool, Novembre 1996.

[Baptiste *et al.* 03] P. Baptiste, L. Péridy & E. Pinson. *A Branch an Bound to minimize the number of late jobs on a single machine with release time constraints.* European Journal of Operational Research, vol. 144, no. 1, pages 1–11, 2003.

[Barnhart *et al.* 98a] C. Barnhart, N.L. Boland, L.W. CLarke, E.L. Johnson, G.L. Nemhauser & R.G. Shenoi. *Flight string models for aircraft fleeting and routing.* Transportation Science, vol. 32, no. 3, pages 208–220, 1998.

[Barnhart *et al.* 98b] C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh & P.H. Vance. *Branch and price : column generation for solving huge integer programs.* Operations Research, vol. 46, no. 3, pages 316–329, 1998.

[Ben Amor 02] H. Ben Amor. *Stabilisation de l'algorithme de génération de colonnes.* Phd thesis, École Polytechnique, Montréal, 2002.

[Benders 62] J.F. Benders. *Partitioning procedures for solving mixed variables programming problems.* Numerische Mathematik, vol. 4, pages 238–252, 1962.

[Benoist & Rottembourg 03] T. Benoist & B. Rottembourg. *Calcul de bornes supérieures du revenu d'un satellite d'observation.* In 5ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'2003), Avignon, France, 26-28 février 2003.

[Bensana *et al.* 99] E. Bensana, M. Lemaître & G. Verfaillie. *Earth observation satellite management.* Constraints, vol. 4, no. 3, pages 293–299, 1999.

[Blazewicz *et al.* 01] J. Blazewicz, K.H. Ecker, E. Pesch, G. Schmidt & J. Weglarz. Scheduling computer and manufacturing processes, second edition. Springer, Verlag, 2001.

[Bresina *et al.* 97] J.L. Bresina, R.A. Morris & W.R. Edgington. *Optimizing observation scheduling objectives.* In 1st NASA Workshop on Planning and Scheduling for Space, Oxnard, USA, oct 1997.

[Brochet 99] C. Brochet. *Planification et maintien à poste de constellations de satellites : modélisation, optimisation, simulation.* Thèse, Université Paul Sabatier, Toulouse, 1999.

[Brucker *et al.* 94] P. Brucker, B. Jurisch & A. Kraemer. *The job shop scheduling problem and immediate selections.* Annals of Operations Research, vol. 50, pages 73–114, 1994.

[Camalot 00] J.P. Camalot. *Aide à la décision et à la coopération en gestion du temps et des ressources.* Thèse, Institut National des Sciences Appliquées, Toulouse, 2000.

[Carlier & Pinson 94] J. Carlier & E. Pinson. *Adjustment of heads and tails for the job-shop problem.* European Journal of Operational Research, vol. 78, pages 146–161, 1994.

[Carlier 75] J. Carlier. *Ordonnancement à contraintes disjonctives.* Thèse de troisième cycle, Université Paris VI, 1975.

[Caseau 03] Y. Caseau. *Challenge ROADEF03 : Combinaison de Propagation de Contraintes et de Métaheuristiques.* In 5ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'2003) – Challenge ROADEF'2003, Avignon, France, 26-28 février 2003.

[Cesta *et al.* 02a] A. Cesta, G. Cortellessa, A. Oddi & N. Policella. *Interaction services for mission planning in Mars-Express.* In 3rd NASA Workshop on Planning and Scheduling for Space, Houston, USA, oct 2002.

[Cesta *et al.* 02b] A. Cesta, G. Cortellessa, A. Oddi & N. Policella. *MEXAR: An Intelligent Support for Space Mission Planning.* In Proceedings of the Workshop on AI Planning and Scheduling for Autonomy in Space Applications, Manchester, July 6, 2002.

[Chabrier *et al.* 02] A. Chabrier, E. Danna & C. Le Pape. *Coopération entre génération de colonnes avec tournées sans cycle et recherche locale appliquée au routage de véhicules.* In Actes de JNPC'02, Nice, France, mai 2002.

[Chabrier 99] A. Chabrier. *Une approche coopérative PPC-PL pour le problème de génération de pairings.* In 2ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'99), Autrans, France, 13-15 janvier 1999.

[Chang *et al.* 95] H.S. Chang, B.W. Kim, c.G. Lee, Y. Choi, H.S. Yang S.L. Min & C.S. Kim. *Topological design and routing for low earth orbit satellite networks.* IEEE Globecom, 1995.

[Cordeau & Laporte 03] J.F. Cordeau & G. Laporte. *Une méthode de recherche abec tabous pour la gestion de la mission de satellites d'observation de la*

Terre. In 5ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'2003) – Challenge ROADEF'2003, Avignon, France, 26-28 février 2003.

[Dafir 99] M. Dafir. *Modélisation par réseaux de Petri et optimisation des handovers dans les constellations de satellitaires*. Rapport de DEA ENSEEIHT, 1999.

[Danna 04] E. Danna. *Intégration des techniques de recherche locale à la Programmation Linéaire en Nombres Entiers*. Thèse de doctorat, Université d'Avignon et des Pays de Vaucluse, Avignon, 2004.

[de Givry & Jeannin 03] S. de Givry & L. Jeannin. *Modélisation et résolution en programmation par contraintes*. In 5ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'2003) – Challenge ROADEF'2003, Avignon, France, 26-28 février 2003.

[Debruyne & Bessière 97] R. Debruyne & C. Bessière. *Some practical filtering techniques for the constraint satisfaction*. In Proc. of the 15^{th} IJCAI, pages 412–417, Nagoya, Japan, 1997.

[Dechter *et al.* 91] R. Dechter, I. Meiri & J. Pearl. *Temporal constraint networks*. Artificial Intelligence, vol. 49, pages 61–95, 1991.

[Demassey 03] S. Demassey. *Méthodes hybrides de programmation par contraintes et programmation linéaire pour le problème d'ordonnancement de projet à contraintes de ressources*. Thèse de doctorat, Université d'Avignon et des Pays de Vaucluse, Avignon, 2003.

[Desaulniers *et al.* 97] G. Desaulniers, J. Desrosiers, Y. Dumas, M.M. Solomon & F. Soumis. *Daily aircraft routing and scheduling*. Management Science, vol. 43, no. 6, pages 841–855, 1997.

[Desaulniers *et al.* 99] G. Desaulniers, J. Desrosiers & M.M. Solomon. *Accelerating strategies in column generation methods for vehicle routing and crew scheduling problems*. Les cahiers du GERAD, aug 1999.

[Desrochers & Soumis 88] M. Desrochers & F. Soumis. *A generalized permanent labelling algorithm for the shortest path problem with time windows*. INFOR, vol. 26, no. 3, pages 191–212, 1988.

[Desrochers *et al.* 92] M. Desrochers, J. Desrosiers & M.M. Solomon. *A new Optimization algorithm for the vehicle routing problem with time windows*. Operations Research, vol. 40, no. 2, pages 342–354, 1992.

[Desrosiers *et al.* 84] J. Desrosiers, F. Soumis & M. Desrochers. *Routing with time windows by column generation*. Networks, vol. 14, pages 545–565, 1984.

[Erschler 76] J. Erschler. *Analyse sous contraintes et aide à la décision pour certains problèmes d'ordonnancement*. Thèse de Doctorat d'Etat, Université Paul Sabatier, Toulouse, 1976.

[Esquirol & Lopez 99] P. Esquirol & P. Lopez. L'ordonnancement. Economica, Paris, 1999.

[Esquirol *et al.* 01] P. Esquirol, P. Lopez & M.J. Huguet. *Propagation de contraintes en ordonnancement*. In P. Lopez & F. Roubellat, éditeurs, Ordonnancement de la production, pages 131–167. Hermès Science Publications, Paris, 2001.

[Estlin & Gaines 02] T.A. Estlin & D.M. Gaines. *An optimization framework for independent planning goals.* In 6th International Conference on Artificial Intelligence Planning and Scheduling (AIPS'02), Toulouse, France, avril 2002.

[Feillet *et al.* 03] D. Feillet, P. Dejax, M. Gendreau & C. Guegen. *An exact algorithm for the elementary shortest path problem with resource constraints : application to some vehicle routing problems.* submitted to Networks, 2003.

[Feillet 01] D. Feillet. *Problèmes de tournées avec gains : étude et application au transport inter-usines.* Thèse de doctorat, École Centrale, Paris, 2001.

[Ferreira *et al.* 00] A. Ferreira, J. Galtier, P. Mahey, G. Mateus, A. Oliveira & M. Pichereau. *An Optimization model for routing in low earth orbit satellite constellation.* In International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN'00), Dallas/Richardson, USA, dec 2000.

[Ferreira *et al.* 01a] A. Ferreira, J. Galtier & P. Pena. *Topological design, routing and hand-over in satellite networks.* In Handbook of Wireless Networks and Mobile Computing. John Wiley and Sons, 2001.

[Ferreira *et al.* 01b] A. Ferreira, J. Galtier, J.-N. Petit & H. Rivano. *Re-routing algorithms in a meshed satellite constellation.* Annales des Telecommunications, vol. 56, pages 169–174, 2001.

[Frank *et al.* 01] J. Frank, A. Jónsson, R. Morris & D. Smith. *Planning and scheduling for fleets of Earth observing satellites.* In 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2001), Montréal, Canada, juin 2001.

[Fukunaga *et al.* 97] A.S. Fukunaga, G. Rabideau, S. Chien & D. Yan. *ASPEN : A framework for automated planning and scheduling of spacecraft control and operations.* In International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS), Tokyo, Japan, 1997.

[Gabrel & Vanderpooten 02] V. Gabrel & D. Vanderpooten. *Enumeration and interactive selection of efficient paths in a multiple criteria graph for scheduling an earth observing satellite.* European Journal of Operational Research, vol. 139, pages 533–542, 2002.

[Gabrel 99] V. Gabrel. *Improved linear programming bounds via column generation for daily scheduling of earth observation satellite.* Rapport LIPN, janv 1999.

[Garey & Johnson 79] M.R. Garey & D.S. Johnson. Computers and intractability, a guide to the theory of NP-Completeness. Freeman, San Francisco, 1979.

[Gilmore & Gomory 61] P.C. Gilmore & R.E. Gomory. *A linear programming approach to the cutting stock problem.* Operations Research, vol. 9, pages 849–859, 1961.

[Gilmore & Gomory 63] P.C. Gilmore & R.E. Gomory. *A linear programming approach to the cutting stock problem – Part II.* Operations Research, vol. 11, pages 863–888, 1963.

[Gilmore & Gomory 65] P.C. Gilmore & R.E. Gomory. *Multistage cutting-stock problems in two or more dimensions.* Operations Research, vol. 13, pages 94–120, 1965.

[Golden *et al.* 87] B.L. Golden, L. Levy & R. Vohra. *The Orienteering Problem.* Naval Research Logistics, vol. 34, no. 3, pages 307–318, 1987.

[Goldfarb & Todd 89] D. Goldfarb & M.J. Todd. *Linear programming.* In M.J. Todd G.L. Nemhauser A.H.G. Rinnooy Kan, éditeur, Handbooks in operations research and management science, 1, pages 73–170. Elsevier, 1989.

[Gomory 58] R.E. Gomory. *Outline of an algorithm for integer solutions to linear programs.* Bulletin of the American Mathematical society, vol. 64, pages 275–278, 1958.

[Gondran & Minoux 95] M. Gondran & M. Minoux. Graphes et algorithmes. Eyrolles, Paris, 1995.

[Guéret *et al.* 00] C. Guéret, C. Prins & M. Sevaux. Programmation linéaire. Eyrolles, Paris, 2000.

[Gutin & Punnen 02] G. Gutin & A.P. Punnen. The traveling salesman problem and its variations. Kluwer Academic Publishers, Dordrecht, 2002.

[Habet & Vasquez 03] D. Habet & M. Vasquez. *Tabu algorithm on a saturated and consistent neighborhood for selecting and scheduling photographs of agile Earth observing satellite.* Bulletin de la ROADEF, no. 10, pages 11–14, été 2003.

[Hall & Magazine 94] N.G. Hall & M.J. Magazine. *Maximizing the value of a space mission.* European Journal of Operational Research, vol. 78, pages 224–241, 1994.

[Harrison & Price 99] S.A. Harrison & M.E. Price. *Task scheduling for satellite based imagery.* In 18th Workshop of UK Planning and Scheduling Special Interest Group, pages 64–78, Salford, UK, dec 1999.

[Harvey & Ginsberg 95] W. Harvey & M. Ginsberg. *Limited Discrepancy Search.* In 14th IJCAI, pages 607–615, Montréal, Canada, 1995.

[Jaumard *et al.* 03] B. Jaumard, R. Morales & M. Gendreau. *Gestion optimisée des prises de vues d'un satellite par une méthode de recherche tabou.* In 5ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'2003) – Challenge ROADEF'2003, Avignon, France, 26-28 février 2003.

[Johnson *et al.* 00] E.L. Johnson, G.L. Nemhauser & M.W.P Savelsbergh. *Progress in linear programming-based algorithms for integer programming : an exposition.* INFORMS Journal on Computing, vol. 12, no. 1, pages 2–23, 2000.

[Jónsson *et al.* 00] A.K. Jónsson, P. Morris, N. Muscettola & K. Rajan. *Planning in interplanetary space : theory and practice.* In 5th International Conference on Artificial Intelligence Planning and Scheduling (AIPS'00), Beckenridge, USA, avril 2000.

[Karmarkar 84] N. Karmarkar. *A new polynomial time algorithm for linear programming.* Combinatorica, vol. 4, pages 373–395, 1984.

[Khachian 79] L.G. Khachian. *A polynomial algorithm in linear programming.* Soviet Mathmatics Doklady, vol. 20, pages 191–194, 1979.

[Klee & Minty 72] V. Klee & G. J. Minty. *How good is the simplex algorithm.* In Shisha Os, éditeur, Inequalities III, pages 159–175. Academic Press, New-York, 1972.

[Kuipers 03] E.J. Kuipers. *An algorithm for selecting and timetabling requests for an Earth Observation Satellite.* Bulletin de la ROADEF, no. 11, pages 7–10, Décembre 2003.

[Laburthe 98] F. Laburthe. *Contraintes et algorithmes en optimisation combinatoire.* Thèse de doctorat, Université Paris VII- Denis Diderot, Paris, 1998.

[Landes 02] E. Landes. *Allocation de ressources de télécommunications dans une constellation de satellites à orbite basse.* Thèse de doctorat, Institut National des Sciences Appliquées, Toulouse, 2002.

[Laporte & Martello 90] G. Laporte & S. Martello. *The selective travelling salesman problem.* Discrete Applied Mathematics, vol. 26, pages 193–207, 1990.

[Laporte 92] G. Laporte. *The Travelling Salesman Problem.* European Journal of Operational Research, vol. 59, no. 2, pages 231–247, 1992.

[Larsen 99] J. Larsen. *Vehicle routing with time windows – finding optimal solutions efficiently.* DORSnyt (newsletter of the Danish OR society ; in english) no. 116, 1999.

[Lasdon 70] L.S. Lasdon. Optimization theory for large systems. Macmillan, London, 1970.

[Lübbecke & Desrosiers 02] M.E. Lübbecke & J. Desrosiers. *Selected topics in column generation.* http://www.optimization-online.org, dec 2002.

[Le Pape 94] C. Le Pape. *Constraint-Based Programming for scheduling : an historical perspective.* Working paper, Operations Research Society Seminar on Constraint Handling Techniques, 1994.

[Lebbar 00] M. Lebbar. *Résolution de problèmes combinatoires dans l'industrie, apport de la programmation mathématique et des techniques de décomposition.* Thèse de doctorat, École Centrale, Paris, 2000.

[Lee & Park 01] T. Lee & S. Park. *An integer programming approach to the time slot assignment problem in SS/TDMA systems with intersatellite links.* European Journal of Operational Research, vol. 135, pages 57–66, 2001.

[Lemaître *et al.* 02] M. Lemaître, G. Verfaillie, F. Jouhaud, J.M. Lachiver & N. Bataille. *Selecting and scheduling observations of agile satellites.* Aerospace Science and Technology, vol. 6, pages 367–381, 2002.

[Levy 96] M.-L. Levy. *Méthodes par décomposition temporelle et problèmes d'ordonnancement.* Thèse de doctorat, Institut National Polytechnique de Toulouse, 1996.

[Little *et al.* 63] J. Little, K. Murty, D. Sweeney & C. Karel. *An algorithm for the Travelling Salesman Problem.* Operations Research, vol. 11, no. 6, pages 972–989, 1963.

[Lopez & Roubellat 01] P. Lopez & F. Roubellat. Ordonnancement de la production. Hermes, Paris, 2001.

[Lopez *et al.* 92] P. Lopez, J. Erschler & P. Esquirol. *Ordonnancement de tâches sous contraintes : une approche énergétique.* RAIRO Automatique, Productique, Informatique Industrielle, vol. 26, pages 453–481, 1992.

[Lopez 91] P. Lopez. *Approche énergétique pour l'ordonnancement de tâches sous contraintes de temps et de ressources.* Thèse de Doctorat, Université Paul Sabatier, Toulouse, 1991.

[Lopez 03] P. Lopez. *Approche par contraintes des problèmes d'ordonnancement et d'affectation : structures temporelles et mécanismes de propagation.* Habilitation à Diriger des Recherches, Institut National Polytechnique de Toulouse, dec 2003.

[Mancel & Saleman 01] C. Mancel & M. Saleman. *Formalisation des besoins d'optimisation de la boucle de programmation Netlander.* Contrat CNES, Rapport CNES DSO/RC/IA/NC/01-076, avril 2001.

[Mancel *et al.* 01a] C. Mancel, P. Lopez & R. Valette. *Formalisation des besoins pour la spécification fonctionnelle d'un outil d'aide à l'élaboration de plans de travail dans le cadre du projet Netlander.* Contrat CNES, rapport LAAS No 01473, nov 2001.

[Mancel *et al.* 01b] C. Mancel, P. Lopez, R. Valette, M. Saleman & N. Bataille. *Approaches for the optimization of the resources management in a satellite constellation.* In 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS 2001), Montréal, Canada, jun 2001.

[Mancel *et al.* 01c] C. Mancel, M. Saleman, M. Faup, P. Lopez & R. Valette. *Optimization of the resource management in a satellite constellation: Progressive approaches and perspective for a hybrid global method.* In 2nd International Workshop on Satellite Constellations and Formation Flying, pages 1–8, Haifa, Israel, feb 2001.

[Mancel *et al.* 02] C. Mancel, P. Lopez, N. Rivière & R. Valette. *Relationships between Petri nets and constraint graphs: application to manufacturing.* In 15th IFAC World congress on Automatic Control, Barcelona, Spain, jul 2002.

[Mancel *et al.* 03a] C. Mancel, M.-J. Huguet, P. Lopez, M. Medjoudj & R. Valette. *Résolution intégrée d'ordonnancement et d'affectation : adaptation d'une méthode de recherche à divergence limitée à une approche fondée sur les réseaux de Petri et la logique linéaire.* In 4ème Conférence Francophone de MOdélisation et SIMulation (MOSIM'03), pages 293–298, Toulouse, 23-25 Avril 2003.

[Mancel *et al.* 03b] C. Mancel, P. Lopez & R. Valette. *Approche par génération de colonnes d'un problème d'optimisation de prises de vue satellite.* In 4ème Conférence Francophone de MOdélisation et SIMulation (MOSIM'03), pages 122–127, Toulouse, 23-25 Avril 2003.

[Mancel 00] C. Mancel. *Optimisation de la gestion de ressources dans des constellations de satellites.* Rapport de DEA, CNES, sept 2000.

[Mancel 03] C. Mancel. *A column generation approach for Earth observing satellites.* In 2nd Operational Research Peripatetic Postgraduate Programme Conference (ORP3), pages 337–349, Lambrecht, Germany, Sept 2003.

[Marchand *et al.* 99] H. Marchand, A. Martin, R. Weismantel & L. Wolsey. *Cutting planes in integer and mixed integer programming.* CORE discussion paper, oct 1999.

[Marsal *et al.* 01] O. Marsal, J.B. Dubois, B. Laborde & P. Pacholczyk. *Netlander Mission and System Technical Specification.* rapport CNES MARS-TS-00-026-CNES, février 2001.

[Martello *et al.* 99] S. Martello, D. Pisinger & P. Toth. *Dynamic programming and strong bounds for the 0-1 knapsack problem.* Management Science, vol. 45, pages 414–424, 1999.

[Martello *et al.* 00] S. Martello, D. Pisinger & P. Toth. *New trends in exact algorithms for the 0-1 knapsack problem.* European Journal of Operational Research, vol. 123, no. 2, pages 325–332, 2000.

[Martin & Shmoys 96] P. Martin & D.B. Shmoys. *A new approach to computing optimal schedules for the job-shop scheduling problem.* In Proc. of the 5^{th} International Conference on Integer Programming and Combinatorial Optimization, IPCO'96, pages 389–403, Vancouver, British Columbia, Juin 1996. LNCS, volume 1084, Springer Verlag. W.H. Cunningham, S.T. McCormick, and M. Queyranne (editors).

[Merle *et al.* 99] O. Du Merle, D. Villeneuve, J. Desrosiers & P. Hansen. *Stabilized column generation.* Discrete Mathematics, vol. 194, pages 229–237, 1999.

[Minoux 83a] M. Minoux. Programmation mathématique, théorie et algorithmes, tome 1. Dunod, Paris, 1983.

[Minoux 83b] M. Minoux. Programmation mathématique, théorie et algorithmes, tome 2. Dunod, Paris, 1983.

[Muscettola *et al.* 95] N. Muscettola, B. Pell, O. Hansson & S. Mohan. *Automating mission scheduling for space-based observatories.* In G.W. Henry & J.A. Eaton, éditeurs, Robotic Telescopes: current capabilities, present developpements and future prospects for automated astronomy. Astronomical Society of the Pacific, Provo, UT, 1995.

[Nemhauser & Wolsey 88] G.L. Nemhauser & L.A. Wolsey. Integer and combinatorial optimization. Wiley, New-York, 1988.

[Nemhauser & Wolsey 89] G.L. Nemhauser & L.A. Wolsey. *Integer programming.* In M.J. Todd G.L. Nemhauser A.H.G. Rinnooy Kan, éditeur, Handbooks in operations research and management science, 1, pages 447–528. Elsevier, 1989.

[Nuijten 94] W.P.M. Nuijten. *Time and resource constrained scheduling - A constraint satisfaction approach.* PhD thesis, Eindhoven University of Technology, 1994.

[Oddi *et al.* 03] A. Oddi, N. Policella, A. Cesta & G. Cortellessa. *Generating high quality schedules for spacecraft memory downlink problem.* In Lectures notes in Computer Science, éditeur, Principles and practice

of Constraint Programming, pages 570–584. Springer – Verlag Heidelberg, 2003.

[Oliva *et al.* 01] C. Oliva, P. Michelon & C. Artigues. *Constraint and Linear Programming : Using Reduced Costs for solving the Zero/One Multiple Knapsack Problem.* In International Conference on Constraint Programming, Workshop on Cooperative Solvers in Constraint Programming (CoSolv 01), pages 87–98, Paphos, Cyprus, 2001.

[Oliva 04] C. Oliva. *Techniques hybrides de propagation de contraintes et de programmation mathématique.* Thèse de doctorat, Université d'Avignon et des Pays de Vaucluse, Avignon, 2004.

[Paschos *et al.* 01] V. Paschos, C. Murat & V. Gabrel. *La Recherche Opérationnelle pour la planification de prises de vue par satellite.* Document no. 121 du LAMSADE, sept 2001.

[Pemberton 00] J.C. Pemberton. *Toward scheduling over-constrained remote-sensing satellites.* In 2nd NASA Workshop on Planning and Scheduling for Space, San Francisco, USA, march 2000.

[Pradin-Chézalviel *et al.* 99] B. Pradin-Chézalviel, R. Valette & L.A. Künzle. *Scenario duration characterization of t-timed Petri nets using linear logic.* In 8th International Workshop on Petri Nets and Performance Models (IEEE PNPM'99), pages 208–217, Zaragoza, Spain, sept 1999.

[Pratt *et al.* 99] S.R. Pratt, R.A. Raines, C.E. Fossa & M.A. Temple. *An operational and performance overview of the IRIDIUM low earth orbit satellite system.* IEEE Communications surveys and tutorials, vol. 2, no. 2, 1999.

[Péridy 96] L. Péridy. *Le problème de job-shop : arbitrages et ajustements.* Thèse de Doctorat, Université de Technologie de Compiègne, 1996.

[Prins 91] C. Prins. *Deux problèmes d'ordonnancement en télécommunications par satellite.* RAIRO-Recherche Opérationnelle/Operations Research, vol. 25, no. 3, pages 341–358, 1991.

[Prins 94] C. Prins. Algorithmes de graphes. Eyrolles, Paris, 1994.

[Ribeiro & Soumis 94] C.C. Ribeiro & F. Soumis. *A column generation approach to the multiple-depot vehicle scheduling problem.* Operations Research, vol. 42, no. 1, pages 41–52, 1994.

[Ribeiro 03] F.P. Ribeiro. *ROADEF Challenge 2003-System Description.* In 5ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'2003) – Challenge ROADEF'2003, Avignon, France, 26-28 février 2003.

[Richard 97] P. Richard. *Contribution des réseaux de Petri à l'étude de problèmes de recherche opérationnelle.* Thèse de doctorat, Université François Rabelais, Tours, 1997.

[Ridder *et al.* 01] J.J. De Ridder, P. Bacquet, J.M. Pechinot & N. Douchin. *Mars Network Performance Evaluation Tools.* 19^{th} AIAA International Communications Satellite Systems Conference, april 17-20 2001.

[Russel & Norwig 95] S. Russel & P. Norwig. Artificial intelligence : A modern approach. Prentice Hall, 1995.

[Sabin & Freuder 94] D. Sabin & E. Freuder. *Contradicting conventional wisdom in constraint satisfaction.* In A. Borning, éditeur, Proc. of PPCP-94, Seattle, USA, May 1994.

[Savelsbergh 97] M. Savelsbergh. *A branch-and-price algorithm for the generalized assignment problem.* Operations Research, vol. 45, no. 6, pages 831–841, 1997.

[Schiex *et al.* 95] T. Schiex, H. Fargier & G. Verfaillie. *Valued Constraint Satisfaction Problems: hard and easy problems.* In International joint Conference in Artificial Intelligence (IJCAI 95), Montreal, Canada, August 1995.

[Schwalb & Dechter 97] E. Schwalb & R. Dechter. *Processing disjunctions in temporal constraint networks.* Artificial Intelligence, vol. 93, pages 29–61, 1997.

[Smith *et al.* 00] D.E. Smith, J. Frank & A.K. Jónsson. *Bridging the gap between Planning and Scheduling.* Knowledge Engineering Review, vol. 15, no. 1, 2000.

[Sourd 04] F. Sourd. *Earliness-tardiness scheduling with setup considerations.* to appear in Computers and operations research, 2004.

[Sourisse & Sorre 97] P. Sourisse & H. Sorre. *Skybridge : un réseau multimédia haute vitesse par satellite.* Revue de l'électricité et de l'électronique, vol. 11, pages 49–52, 1997.

[Teghem 96] J. Teghem. Programmation linéaire. Ellipses, Bruxelles, 1996.

[Torres & Lopez 00] P. Torres & P. Lopez. *On not-first/not-last conditions in disjunctive scheduling.* European Journal of Operational Research, vol. 127, no. 2, 2000.

[Van Den Akker *et al.* 00] J.M. Van Den Akker, C.A.J. Hurkens & M.W.P. Savelsbergh. *Time-indexed formulations for machine scheduling problems : column generation.* Informs Journal on Computing, vol. 12, no. 2, pages 111–125, 2000.

[Van Den Akker 94] J.M. Van Den Akker. *LP-based solution methods for single-machine scheduling problems.* PhD thesis, Technische Universiteit Eindhoven, The Netherlands, 1994.

[Vance 98] P.H. Vance. *Branch-and-price algorithms for the one-dimensional cutting stock problem.* Computational Optimization and Applications, vol. 9, no. 3, pages 211–228, 1998.

[Vanderbeck & Wolsey 96] F. Vanderbeck & L.A. Wolsey. *An exact algorithm for IP column generation.* Operations research letters, vol. 19, pages 151–159, 1996.

[Vanderbeck 00] F. Vanderbeck. *On Dantzig-Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm.* Operations Research, vol. 48, no. 1, pages 111–128, 2000.

[Vasquez & Hao 01] M. Vasquez & J.K. Hao. *A logic-Constrained knapsack formulation and a Tabu Algorithm for the daily photograph scheduling of an Earth Observation satellite.* Journal of Combinatorial Optimization and Applications, vol. 20, no. 2, pages 137–157, 2001.

[Vasquez & Hao 03] M. Vasquez & J.K. Hao. *Upper bounds for the SPOT 5 daily photograph scheduling problem.* Journal of Combinatorial Optimization, vol. 7, pages 87–103, 2003.

[Vasquez *et al.* 03] M. Vasquez, Y. Vimont & J. Vlasak. *Algorithmes approchés pour la résolution du problème de planification d'un satellite agile.* In 5ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'2003) – Challenge ROADEF'2003, Avignon, France, 26-28 février 2003.

[Verfaillie & Bensana 95] G. Verfaillie & E. Bensana. *Évaluation d'algorithmes sur le problème de programmation journalière des prises de vues du satellite d'observation SPOT5.* Rapport technique ONERA, août 1995.

[Verfaillie *et al.* 96] G. Verfaillie, M. Lemaître & T. Schiex. *Russian Doll Search for solving constraint optimization problems.* In 13th National Conference on Artificial Intelligence, pages 182–187, Portland, USA, 1996.

[Wallace & Schimpf 02] M. Wallace & J. Schimpf. *Finding the right hybrid algorithm – A combinatorial meta-problem.* Annals of Mathematics and Artificial Intelligence, vol. 34, pages 259–269, 2002.

[Wild *et al.* 03] E. Wild, G. Parmentier, P.F. Dutot & G. Da-Costa. *Challenge ROADEF'03 Gestion de la mission de satellites d'observation de la Terre.* In 5ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'2003) – Challenge ROADEF'2003, Avignon, France, 26-28 février 2003.

[Wolfe & Sorensen 00] W.J. Wolfe & S.E. Sorensen. *Three scheduling algorithms applied to the Earth observing systems domain.* Management Science, vol. 46, no. 1, pages 148–168, 2000.

[Wolsey 98] L.A. Wolsey. Integer programming. Wiley, New-York, 1998.