SPIM

Habilitation à Diriger des Recherches

UFC

école doctorale **sciences pour l'ingénieur et microtechniques**

UNIVERSITÉ DE FRANCHE-COMTÉ

Nouveaux modèles de programmation linéaires et de flots pour la résolution de problèmes d'optimisation difficiles

KARINE DESCHINKEL

SPIM

Habilitation à Diriger des Recherches

UFC

école doctorale **sciences pour l'ingénieur et microtechniques**

UNIVERSITÉ DE FRANCHE-COMTÉ

HABILITATION À DIRIGER DES RECHERCHES

de l'Université de Franche-Comté

préparée au sein de l'l'Université de Franche-Comté

Spécialité : **Informatique**

présentée par

KARINE DESCHINKEL

Nouveaux modèles de programmation linéaires et de flots pour la résolution de problèmes d'optimisation difficiles

Soutenue publiquement le XX Mois XXXX devant le Jury composé de :

LAËTITIA JOURDAN	Rapporteur	Professeur à l'Université de Lille 1
LHASSANE IDOUMGHAR	Rapporteur	Professeur à l'Université de Haute-Alsace
AMMAR OULAMARA	Rapporteur	Professeur à l'Université de Lorraine
YE-QIONG SONG	Examinateur	Professeur à l'Université de Lorraine
PIERRE-CYRILLE HÉAM	Examinateur	Professeur à l'Université de Franche-Comté
RAPHAËL COUTURIER	Examinateur	Professeur à l'Université de Franche-Comté

REMERCIEMENTS

J'exprime mes remerciements les plus sincères envers mes rapporteurs, Laetitia Jourdan, Lhassane Idoumghar, Ammar Oulamara, qui m'ont fait l'honneur de consacrer du temps à la lecture de ce manuscrit pour donner leurs conseils avisés et leurs appréciations sur mes travaux de recherche.

Mes remerciements sont également adressés aux examinateurs Ye-Qiong Song, Raphaël Couturier, et Pierre-Cyrille Héam, qui m'ont fait l'honneur de participer au jury.

Je tiens aussi à remercier l'ensemble du jury d'avoir accepté de juger mon travail.

Les travaux présentés dans ce mémoire représentent une partie des fructueuses collaborations scientifiques qui ont jalonné ma carrière. Je remercie sincèrement Sid Touati qui m'a appris beaucoup sur notre métier. J'ai toujours grand plaisir à travailler en interdisciplinarité avec d'autres chercheurs. Je les remercie tous de m'avoir exposée leur problématique et ouvert mon esprit sur de nouveaux domaines d'application.

J'adresse un remerciement collégial, aux différents collègues qui ont croisé ma route au PRiSM, et à tous les collègues du DISC et de l'IUT avec qui je partage actuellement de bons moments de détente et de travail.

Je tiens tout particulièrement à exprimer ma profonde reconnaissance à Jacques Bahi, d'avoir accepté ma venue dans l'équipe AND, et de m'avoir accordée son entière confiance, depuis mon arrivée à l'Université de Franche Comté, en me chargeant de plusieurs responsabilités.

Je remercie vivement Raphaël Couturier pour m'avoir accueillie chaleureusement dans l'équipe AND. J'apprécie fortement sa disponibilité, ses conseils, sa sympathie, et ses encouragements.

Enfin un grand merci à mon mari qui m'a toujours encourager dans mon travail d'enseignant-chercheur.

SOMMAIRE

INTRODUCTION

1

RENSEIGNEMENT GÉNÉRAUX

1.1/ ÉTAT CIVIL

Date, lieu de naissance : 27 Mai 1975 à Laxou (54)
Nationalité : française
Situation de famille : 2 enfants, nés en novembre 2003 et 2006
Adresse personnelle : 15 grande Rue 25490 BADEVEL
Téléphone personnel : 09.50.51.19.11
Adresse professionnelle : I.U.T. de Belfort – Rue Engel Gros BP 527, 90000 Belfort
Téléphone professionnel : 03.84.58.77.84
Adresse électronique : karine.deschinkel@univ-fcomte.fr
Page personnelle : http ://members.femto-st.fr/karine-deschinkel

1.2/ FORMATION

2001 **THÈSE DE DOCTORAT** en Systèmes Industriels
Régulation du trafic aérien par optimisation dynamique des prix d'utilisation du réseau
Mention : Très Honorable avec Félicitations
École Nationale Supérieure de l'Aéronautique et de l'Espace (ENSAE)-Toulouse
Thèse soutenue le 8 novembre 2001 devant le jury composé de :
J.L. FARGES, Ingénieur de recherche ONERA, Directeur de thèse
D.DELAHAYE, Chercheur LOG, CENA, Co-directeur
E. FÉRON, Professeur MIT, Rapporteur
J.B. LESORT Directeur LICIT , Rapporteur
M. MINOUX, Professeur Université Paris 6, Président
J.F. BONNANS, Directeur de recherche INRIA, Examinateur

1998 **D.E.A. MATHÉMATIQUES APPLIQUÉES**

Mention : Assez Bien

Université Blaise Pascal -Clermont-Ferrand

Optimisation, Homogénéisation d'équations aux dérivées partielles, Méthodes numériques pour la mécanique des fluides, Méthodes stochastiques en analyse d'images

1998 **DIPLÔME D'INGÉNIEUR GÉNIE MATHÉMATIQUES ET MODÉLISATION**

POLYTECH CLERMONT - CUST Institut des Sciences de l'Ingénieur – Clermont-Ferrand

1.3/ PARCOURS PROFESSIONNEL

2009- **MAÎTRE DE CONFÉRENCES** Section 27 **UFC Université de Franche-comté**

Enseignement : I.U.T. de Belfort-Montbéliard – Département informatique

Recherche : DISC (DEPARTEMENT INFORMATIQUE ET SYSTEMES COMPLEXES) de l'institut FEMTO-ST (Franche-Comté Electronique, Mécanique, Thermique et Optique - Sciences et Technologies)

Equipe AND (Algorithmique Numérique et Distribuée)

2002/09 **MAÎTRE DE CONFÉRENCES** Section 27 **UVSQ**

Enseignement : Université Versailles Saint Quentin en Yvelines (UVSQ)

Recherche : PRiSM (Laboratoire de recherche en informatique) – Equipe Opale (Optimisation et Parallélisme), renommée OCCR (Optimisation et Calcul Réparti)

2001/02 **A.T.E.R.** (Attaché Temporaire d'Enseignement et de Recherche) Section 27

Enseignement : Université Paris 1 - Panthéon Sorbonne

Recherche : CEntre de Recherche en Mathématiques, Statistique et Economie Mathématique (CERMSEM)

1998/01 **DOCTORANTE**

Office National d'Études et de Recherches Aérospatiales (O.N.E.R.A.)- Toulouse

Département de Commande des systèmes et Dynamique du Vol

2

ACTIVITÉS DE RECHERCHE

2.1/ HISTORIQUE DES ACTIVITÉS

J'ai choisi de débuter ce mémoire par une présentation des différentes étapes d'évolution de mon activité de recherche.

2.1.1/ THÈSE DE DOCTORAT(1998-2001)

J'ai réalisé ma thèse à l'Office National d'Études et de Recherches Aérospatiales (O.N.E.R.A.) de Toulouse au Département de Commande des systèmes et Dynamique du Vol. Mes travaux de thèse portent sur la tarification de l'espace aérien pour la régulation du trafic. La croissance du trafic aérien conduit à des retards importants des avions. Le problème de congestion de l'espace aérien peut être résolu en adaptant la demande à la capacité réellement disponible. La tarification se présente comme un outil possible pour inciter les compagnies aériennes à modifier leurs plans de vols. La démarche adoptée pour parvenir à une tarification adéquate des secteurs aériens comporte trois étapes principales :

- Modéliser l'influence des conditions financières sur la définition des horaires et sur le choix des routes par les compagnies aériennes. C'est le modèle Logit de choix discret que nous avons choisi pour prédire les décisions des compagnies aériennes.
- Identifier les paramètres du modèle de choix des compagnies. Nous avons formulé le problème d'identification des paramètres comme un problème de minimisation de l'erreur quadratique et nous l'avons résolu.
- Fixer les prix d'options pour que les décisions prises par les compagnies conduisent à la planification de vols désirée.

Le problème de calcul des prix de secteurs peut se ramener à un problème de minimisation quadratique entre les nombres de vols issus du modèle et les nombres de vols définis par la cible. L'algorithme de descente par le calcul du gradient, l'algorithme du recuit simulé, et un algorithme de recherche Tabou ont été retenus pour l'identification des paramètres et le calcul des prix de secteurs . Tous ces algorithmes ont été implémentés en C++ et testés sur deux exemples, un exemple académique et un exemple plus réaliste. Les résultats obtenus montrent qu'une affectation cible peut être atteinte en orientant le choix des compagnies par les prix ([Deschinkel, 1998], [Deschinkel, 1999], [Deschinkel, 2000], , [Deschinkel et al., 2000a], [Deschinkel et al., 2000b], [Deschinkel et al., 2001a], [Deschinkel et al., 2001b],

[Deschinkel et al., 2002a], [Deschinkel et al., 2002b], [Deschinkel, 2003]).

2.1.2/ POSTE D'ATER(2001-2002)

Après ma thèse, j'ai effectué une année d'ATER (Attaché temporaire d'Enseignement et de Recherche) au Centre de Recherche en Mathématiques, Statistique et Économie Mathématique (CERMSEM). J'ai porté mon intérêt sur les problèmes d'optimisation inverse car le problème de tarification des secteurs aériens en vue d'atteindre une cible peut, sous certaines conditions, être assimilé à un problème d'affectation inverse. J'ai travaillé en collaboration avec le professeur Marc Demange (maintenant Professeur à Essec Business Scool) sur ce sujet.

2.1.3/ MAÎTRE DE CONFÉRENCES UVSQ (2002-2009)

J'ai été recrutée sur un poste de Maître de Conférences en septembre 2002 à l'Université de Versailles St Quentin en Yvelines pour intégrer l'équipe de recherche AOC (Algorithmique, Optimisation et Combinatoire) du laboratoire PRiSM (Parallélisme, Réseaux, systèmes, Modélisation). Mes axes principaux de recherche dans ce laboratoire ont porté sur des :

- Problèmes de tarification dans les télécommunications,
- Problèmes d'optimisation en curiethérapie,
- Méthodes de coupes pour des problèmes d'optimisation en électricité,
- Problèmes d'optimisation de registres en compilation.

2.1.3.1/ PROBLÈMES DE TARIFICATION DANS LES TÉLÉCOMMUNICATIONS

Modèles et Méthodes de tarification Suite à ma thèse, j'ai étendu mes connaissances des problèmes de tarification au domaine des télécommunications en participant en 2003 et 2004 à une action de recherche coopérative PrixNET financée par l'INRIA. Le but de l'action est de développer, d'implémenter et de comparer des méthodes de tarification des réseaux tels que l'Internet, permettant un contrôle de la congestion et une répartition équitable des ressources.
Dans le cadre de cette action, j'ai entrepris une recherche bibliographique, pour faire une synthèse ([Deschinkel, 2004]) sur les modèles d'utilité développés dans les études portant sur la tarification de l'Internet, et voir comment on peut enrichir ces modèles pour mieux décrire le choix des utilisateurs.
En collaboration avec des membres de l'équipe AlCAAp (Algorithmique, combinatoire analytique et applications) du PRiSM à Versailles, j'ai poursuivi mon étude sur les modèles et les méthodes de tarification appliqués à l'Internet. Nous avons intégré le projet européen EURO-NGI (Network of Excellence for Next Generation Internet) en rejoignant les équipes travaillant sur le "workpackage" intitulé : "Payment and Cost Models for Next Generation Internet". Dans le cadre de ce projet, nous avons dressé un bilan sur la modélisation des comportements des usagers (modèles d'utilités) et sur les principes généraux de la tarification de l'Internet en proposant des méthodes prometteuses ([Deschinkel et al., 2004], [Barth et al., 2004a], [Barth et al., 2004b]) pour garantir une meilleure qualité de service aux usagers en fonction de leur propension à payer.

Problème de planification et tarification dans un réseau Nous nous sommes intéressés à un problème de tarification dans un réseau de transport (aérien ou routier, ou de télécommunications). Sous certaines hypothèses idéales, chaque utilisateur du réseau sélectionne la route qui minimise ses coûts de transport, il en résulte alors un flot appelé « équilibre utilisateur » car aucun utilisateur ne peut diminuer ses coûts de transport en changeant de route. Mais cet équilibre utilisateur ne correspond pas nécessairement à la meilleure utilisation du réseau. On retrouve ce résultat dans la paradoxe de Braess où le fait d'ajouter une route supplémentaire peut dans certains cas réduire la performance globale. Un gestionnaire de réseau peut envisager une autre répartition du flot sur le réseau, appelé « équilibre système ». Pour passer d'un équilibre utilisateur à un équilibre système, un prix/péage peut être imposé sur certains arcs du réseau.

Ce problème peut être formulé comme un problème de plus courts chemins inverses. Étant donné un graphe avec des coûts sur les arêtes, le problème consiste à augmenter artificiellement (en fixant des prix) le coût de certaines arêtes de manière à ce que chaque option choisie par le gestionnaire pour chaque utilisateur soit effectivement la meilleure option (du point de vue de l'utilisateur).

Des méthodes de type programmation quadratique convexe ont été proposées mais sans astreindre les coûts à augmenter. Nous avons travaillé sur ce problème, en proposant d'autres méthodes de résolution et étudié la manière de l'appliquer, sous certaines conditions, au problème de tarification de l'espace aérien, et de tarification de requêtes en télécommunication. J'ai proposé un stage de D.E.A. en juin 2004 autour de cette problématique et le travail résultant a été présenté à la conférence ROADEF en Février 2005 [Deschinkel and Oudot, 2005].

Routage de requêtes dans un réseau MPLS Ce travail se concentre sur la partie "planification de trafic" dans le réseau Internet sous une architecture MPLS. L'objectif est de distribuer le trafic (tronçon : paquets de trafic de même classe) sur les différentes routes de manière efficiente pour obtenir une meilleur utilisation des ressources et une plus forte stabilité de routage. Ici, nous nous intéressons au problème off-line. A chaque tronçon, on associe un niveau de trafic dépendant de la classe à laquelle il appartient. Un tronçon est donc défini par sa demande en bande passante, et son niveau de priorité. Le réseau est modélisé sous forme d'un graphe avec des capacités et des coûts associés aux arêtes. Nous cherchons à affecter à chaque tronçon une route en respectant les contraintes de capacités et de manière à maximiser le produit total : (demande + priorité) * coût de la route. Nous avons proposé une formulation mathématique de ce problème sous forme de programme linéaire en nombres entiers ([Deschinkel and Echabbi, 2004]. L'algorithme glouton développé pour ce problème n'est pas optimal. Nous avons recherché une méthode de résolution exacte et une heuristique qui permettent d'obtenir des solutions de meilleure qualité.

2.1.3.2/ PROBLÈMES D'OPTIMISATION EN CURIETHÉRAPIE

J'ai co-encadré avec Catherine Roucairol la thèse de doctorat de François Galéa sur cette thématique de 2003 à 2006.

Le but de ce projet était d'apporter une aide à des médecins pour construire des plans de

traitement plus efficaces et moins ”destructeurs” en curiethérapie (traitement des tumeurs cancéreuses).

Dans un premier temps, nous avons identifié les problèmes d'optimisation qui se posent en curiethérapie en montrant leur spécificité et ensuite nous avons analysé les modèles et les méthodes actuellement utilisées pour les résoudre, sans porter un jugement médical mais plus à la lueur des techniques de Recherche Opérationnelle ([Deschinkel et al., 2005c], [Deschinkel et al., 2006c]).

Nous avons réalisé cette étude en réponse à une demande de Gilbert Boisserie, radiophysicien de l'Unité de Physique du service de radiothérapie de l'hôpital de la Pitié-Salpétrière de Paris, qui souhaitait savoir si le système classique d'implantation des cathéters (connu sous le nom de Système de Paris) utilisé en curiethérapie LDR (à bas débit de dose) pouvait être adapté pour le traitement en curiethérapie HDR (haut débit de dose). Nous avons réalisé un logiciel graphique 3D qui permet de visualiser la distribution de dose sur un fantôme géométrique, et le placement des cathéters. L'utilisateur peut sélectionner un outil d'optimisation qui génère automatiquement les temps d'arrêt en fonction des contraintes de doses qui ont été spécifiées. Les positions et les temps d'arrêt sont obtenus par la résolution d'un programme linéaire continu ([Deschinkel et al., 2005a], [Deschinkel et al., 2005b], [Deschinkel et al., 2006a], [Deschinkel et al., 2006b]) avec des solveurs accessibles dans des logiciels libres (lp_solve, glpk,...). Les résultats de notre étude sur le placement des cathéters ont été publiés dans la revue APJOR [Deschinkel et al., 2008b].

2.1.3.3/ MÉTHODES DE COUPES POUR DES PROBLÈMES D'OPTIMISATION EN ÉLECTRICITÉ

De février 2005 à janvier 2009, j'ai co-encadré avec Catherine Roucairol la thèse de doctorat de Christophe Louat. Cette thèse s'est effectuée dans le cadre d'un contrat CIFRE établi avec RTE (Réseau Transport Électrique).

Cette thèse a pour but d'étudier l'impact des méthodes de coupes efficaces pour des problèmes d'électricité de grande taille afin de les intégrer dans un Branch and Bound (algorithme de séparation-évaluation). Un premier travail consiste à constituer un état de l'art des méthodes de coupes générales (Gomory, MIR, Lift and Project) qui peuvent être intéressantes pour des problèmes avec des variables binaires. Il s'agit ensuite de développer et tester ces méthodes de coupes avec le solveur Xpress Mosel, et d'envisager, après leur analyse, leur intégration au solveur de programmation linéaire mixte développé par RTE.

Une hiérarchisation de ces méthodes et les premières phases d'implémentation et d'analyse ont été présentées à la 3ème édition des Journées Polyèdres et Optimisation Combinatoire en juin 2006 et à ECCO 2007 [Louat et al., 2007].

Ce travail nous a conduit à développer nos propres outils en logiciels libres pour pouvoir gérer l'intégration des différentes méthodes de coupes et leurs paramétrages dans les solveurs commerciaux. *Glop* est une librairie qui permet de créer, manipuler, résoudre des programmes linéaires (LP) ainsi que des programmes mixtes entiers (MIP).

Nous avons choisi d'examiner différentes stratégies de générations de coupes en les testant sur des problèmes MIP 0-1 de la librairie Linlib ([Deschinkel et al., 2008a]).

2.1.3.4/ PROBLÈMES D'OPTIMISATION DE REGISTRES EN COMPILATION

J'ai travaillé sur ces problèmes avec Sid Touati de l'équipe ARPA (Architecture et Parallélisme) du laboratoire PRiSM, désormais professeur à l'Université de Nice Sophia-Antipolis. Ce travail a été en partie financé par le projet ANR MOPUCE. J'ai mis à profit mes compétences techniques de recherche opérationnelle au service d'un domaine de haute technologie, à savoir l'utilisation efficace des nouveaux processeurs à parallélisme d'instructions embarqués.

Le problème d'ordonnancement périodique de tâches (instructions) cycliques avec minimisation du nombre de registres, dans sa forme la plus générale, peut se formuler sous forme d'un programme linaire en nombre entiers où les variables de décisions sont les dates de début de chacune des tâches, des variables binaires indiquant une réutilisation ou non de registres entre deux tâches (non nécessairement distinctes), et les distances de réutilisation. Le programme linéaire ainsi formé présente une structure très forte permettant d'envisager l'application de techniques de décomposition qui, couplées avec des solveurs performants, pourraient rendre possible la résolution exacte de problèmes d'instances réelles. Nous pouvons nous tourner également vers la mise en œuvre de méthodes approchées permettant d'obtenir des solutions de bonne qualité en un temps raisonnable. Nous avons, par exemple, exploité le fait que le problème en question fait apparaître un problème sous-jacent d'affectation pour lequel nous connaissons un algorithme en temps polynomial (méthode Hongroise) pour proposer une heuristique appropriée. Les différentes contributions (modélisation du problème, méthode de résolution par décomposition, modélisation sous forme d'un problème de flot de coût minimum, registres de types multiples) sur ce problème ont fait l'objet de présentations dans les conférences ([Deschinkel and Touati, 2008b], [Deschinkel and Touati, 2008a], [Deschinkel and Touati, 2009], [Deschinkel and Touati, 2010]) et de publications en revue ([Deschinkel et al., 2011], [Touati et al., 2011], [Touati et al., 2013]).

2.1.4/ MAÎTRE DE CONFÉRENCES UFC (2009-)

En septembre 2009, pour des raisons personnelles, j'ai souhaité quitter la région parisienne et j'ai obtenu un poste (par une procédure d'échanges croisés) à l'Université de Franche-comté dans l'équipe AND (Analyse Numérique et Distribuée) du département DISC (Département Informatique des Systèmes Complexes) du laboratoire FEMTO-ST.

Au cours de ma première année sur Belfort, j'ai finalisé mes travaux de recherche entrepris avec Sid Touati, sur le problème d'ordonnancement périodique de tâches cycliques avec minimisation du nombre de registres.

Depuis, j'apporte mes connaissances et compétences en recherche opérationnelle dans l'équipe AND. Je collabore également avec d'autres chercheurs localisés sur Belfort, en apportant mon expertise en théorie des graphes et optimisation : des chercheurs de l'UTBM en conception mécanique et des chercheurs du département ENERGIE de FEMTO-ST. Mon travail de recherche vise à fournir des modèles originaux et des algorithmes de résolution pour des problèmes liés aux thématiques suivantes :

- Problèmes d'optimisation dans les réseaux de capteurs (équipe AND, département DISC, FEMTO-ST, UFC)
- Problème d'optimisation en conception mécanique (équipe INCIS, laboratoire IRTES-M3M, UTBM)

— Problèmes d'optimisation dans le domaine de l'énergie (équipe THERMIE et équipe SHARPAC, département ENERGIE, FEMTO-ST, UFC)

2.1.4.1/ PROBLÈMES D'OPTIMISATION DANS LES RÉSEAUX DE CAPTEURS

Dans l'équipe AND, j'ai réorienté mes travaux de recherche vers la thématique des réseaux de capteurs, en proposant notamment deux algorithmes centralisés d'optimisation de durée de vie du réseau ([Deschinkel, 2012], [Deschinkel and Hakem, 2013]). De septembre 2012 à décembre 2015, j'ai co-encadré, en thèse, Ali Khadum Idrees sur cette problématique. Nous avons étudié des protocoles d'optimisation distribués avec l'objectif de prolonger la durée de vie opérationnelle du réseau [Idrees et al., 2015a]. Pour résoudre le problème, nous avons proposé de nouvelles approches articulées en deux phases. Dans un premier temps, la région à surveiller est divisée en petites sous-régions en utilisant le concept de diviser pour mieux régner. Ensuite, l'un de nos protocoles d'optimisation distribués est exécuté par chaque nœud capteur dans chaque sous-région, afin d'optimiser la couverture et la durée de vie du réseau. Nous proposons trois protocoles distribués [Idrees et al., 2014] qui combinent, chacun, deux techniques efficaces : l'élection d'un nœud leader dans chaque sous-région, suivie par la mise en oeuvre par celui-ci d'un processus d'ordonnancement d'activité des nœuds capteurs de sa sous-région. Cet ordonnancement est porté par la formulation et la résolution de programmes linéaires. Pour les deux premiers protocoles, il s'agit de minimiser simultanément la non couverture ou la sur-couverture d'un ensemble de points particuliers [Idrees et al., 2015b]. Pour le troisième protocole, le nouveau modèle repose sur la couverture du périmètre de chacun des capteurs [Idrees et al., 2016]. Nous avons réalisé plusieurs simulations en utilisant le simulateur à évènements discrets OMNeT++ pour valider l'efficacité de nos protocoles proposés. Nous avons pris en considération les caractéristiques d'un capteur Medusa II pour la consommation d'énergie et le temps de calcul. En comparaison avec deux autres méthodes existantes, nos protocoles ont la capacité d'augmenter la durée de vie du réseau de capteurs et d'améliorer les performances de couverture.
Prochainement, j'encadrerai un nouveau doctorant sur cette thématique. Nous proposerons de nouveaux modèles de maximisation de durée de vie intégrant les notions de couverture partielle, et d'hétérogénéité (en énergie, en unités de captage) des capteurs.

2.1.4.2/ PROBLÈME D'OPTIMISATION EN CONCEPTION MÉCANIQUE

J'ai tissé des liens de recherche avec les membres de l'équipe INCIS du laboratoire IRTES-M3M de l'UTBM. J'ai co-encadré une thèse CIFRE [Robert, 2012](thèse d'Aurélie Robert soutenue en 2012), dont les principaux verrous scientifiques concernent l'adaptation, le déploiement et puis la validation des concepts, méthodes et outils développés au laboratoire IRTES-M3M et dédiés à la conception Hautement Productive, à l'approche de "Design For Manufacturing and Assembly" DFMA de couples produits-process modulaires. Mon travail a consisté à proposer des méthodes fondées sur la théorie des graphes pour la conception de produits modulaires et pour la génération des gammes d'assemblage associées. Nous utilisons la même modélisation que celle employée dans l'algorithme ASDA (Assembly Sequences Definition Algorithm) proposé par Demoly [Demoly et al., 2011] en représentant les composants élémentaires

d'un produit à assembler par des nœuds et les relations de contact ou de précédence entre 2 composants par des arcs. Le graphe orienté ainsi obtenu est simplifié par application d'un algorithme de réduction transitive, puis coloré en fonction de l'appartenance des composants à un module, et pour finir, contracté (regroupement de nœuds) en fonction des informations issues de la matrice de contact. La génération de séquences d'assemblage ordonnées et admissibles est réalisée par l'application d'un algorithme de tri topologique, le graphe étant orienté et acyclique. Afin de rendre opérationnelle cette méthodologie, un outil logiciel appelé ORASSE Produit a été développé. Il permet de guider pas à pas l'utilisateur dans certaines étapes complexes et difficiles à mettre en œuvre manuellement. Et il permet surtout des gains de temps hautement significatifs (jusqu'à 60%) pour l'architecte produit ([Robert et al., 2011c], [Robert et al., 2011a],[Robert et al., 2011b], [Deffrenne et al., 2013], [Bettwy et al., 2014]).

2.1.4.3/ PROBLÈMES D'OPTIMISATION DANS LE DOMAINE DE L'ÉNERGIE

L'intégration du département DISC au sein de l'Institut FEMTO-ST en 2012 a favorisé le travail d'interdisciplinarité. Celui-ci s'est traduit pour ma part, par la collaboration avec des membres du département ENERGIE sur :

- Le contrôle actif en mécanique des fluides ;
- Le dimensionnement et la gestion d'énergie d'un système hybride électrique

Contrôle actif en mécanique des fluides Ce travail a été réalisé en collaboration avec l'équipe THERMIE (*TH*ermique, *E*coulements, instru*M*entat*I*on, *E*nergie) du département ENERGIE de l'Institut FEMTO-ST. L'objectif visé, à terme, est le contrôle actif d'écoulement aérodynamique. Plus précisément, il s'agit d'aboutir à un dispositif composé de micro-actionneurs (émettant ou non de l'air sous pression) et de micro-capteurs (mesurant la pression locale), qui permet de modifier automatiquement l'écoulement d'air au niveau d'une automobile, afin de réduire sa résistance à l'air. Pour cela, à une vitesse de déplacement donnée, il faut pouvoir caractériser un écoulement correspondant à une configuration des micro-actionneurs (ce sont eux qui vont modifier l'écoulement). Or le calcul de l'écoulement avec un logiciel dédié à la mécanique des fluides numérique (CFD pour Computationnal Fluid Dynamics) tel Fluent prend beaucoup trop de temps pour espérer être utilisé en temps réel.

Dans un premier temps, l'étude a porté sur la pertinence de la substitution du calcul CFD par un approximateur universel, à savoir un Réseau de Neurones Artificiels. Ce choix tirait partie de l'expertise développée par l'équipe AND sur les réseaux de neurones notamment dans le contexte de la dosimétrie en radiothérapie. L'étude portait sur un écoulement à l'aval d'une marche car cette géométrie simple facilite les calculs CFD et correspond grossièrement à l'arrière d'une voiture. Nous avons ensuite proposé et validé par simulation un algorithme glouton d'optimisation de la vitesse des jets des actionneurs de manière à maximiser la force de pression exercée à l'arrière du véhicule ([Couchot et al., 2012], [Couchot et al., 2013]).

Dimensionnement et gestion d'énergie d'un système hybride électrique Ce travail de recherche est le fruit d'une collaboration avec l'équipe SHARPAC (Systèmes électriques Hybrides, ActionneuRs électriques, systèmes Piles A Combustible) engagée depuis septembre 2013 et qui se concrétise à travers l'obtention en 2014 d'un projet Région nommé GEOSEFA (Gestion d'énergie des systèmes hybrides électriques optimisée et fiable) et l'encadrement mutuel d'un doctorant Pierre Saenger (contrat doctoral Région depuis septembre 2014).

La thèse concerne le déploiement de stratégies de gestion d'énergie sur une maquette test, composée d'une alimentation, qui émulera une source permanente de puissance (du type alternateur couplé à une turbine ou un système pile à combustible), d'un pack d'accumulateurs et d'un pack de super-condensateurs. Il s'agit de dimensionner les stratégies de gestion d'énergie et d'optimiser la gestion de celle-ci à travers des algorithmes déployés sur les GPUs. Cette association de sources est générique, on peut la retrouver dans diverses applications : automobile, véhicules lourds, aéronautique ou bien stationnaires (en particulier dans le cas de sites îlotés, alimentés par des sources d'énergie renouvelables). Le cadre applicatif privilégié est celui d'une application aéronautique car elle rassemble de nombreuses contraintes en termes de dynamiques de charge, de température ambiante, de qualité de l'énergie. L'objectif de la thèse est de développer des algorithmes d'optimisation incluant le dimensionnement des éléments de stockage et les paramètres de la stratégie de gestion d'énergie répartissant la demande de puissance sur les différentes sources.

Dans un aéronef plus électrique, les éléments de stockage sont multiples et sont connectés (directement ou via un convertisseur) aux bus. Le premier problème étudié dans cette thèse concerne le dimensionnement. Il s'agit de déterminer, pour chaque modèle d'architecture électrique, le nombre et le type de cellules unitaires (accumulateurs et super-condensateurs) parmi une base de données, qui permettent de répondre aux besoins énergétiques (énergie et puissance) et qui constituent des systèmes dont la masse est minimisée. Le problème de dimensionnement est formalisé sous forme d'un problème de minimisation d'une fonction masse (somme des masses du pack accumulateurs et du pack super-condensateurs, et masses des convertisseurs additionnels) dépendante de plusieurs variables (résistance, capacité, décharge, fréquence de coupure, température,...). Un algorithme de recuit simulé est utilisé pour résoudre le problème. La formalisation du problème et sa résolution a fait l'objet d'une publication à la conférence VPPC en 2015 ([Saenger et al., 2015]) et dans la revue IEEE Transactions on Vehicular Technology en 2016 ([Saenger et al., 2016]).

2.2/ RESPONSABILITÉS SCIENTIFIQUES

2.2.1/ ORGANISATION D'ÉVÉNEMENTS SCIENTIFIQUES

Membre du Comité d'organisation des Journées Franciliennes de Recherche Opérationnelle (2004-2006)
De juin 2004 à septembre 2006, j'ai fait partie du comité d'organisation des Journées Franciliennes de Recherche Opérationnelle. Ces journées thématiques ont un double objectif, celui de présenter un tutoriel sur le sujet abordé ainsi que des applications industrielles ou scientifiques en rapport direct avec le sujet. Cette expérience est particulièrement enrichissante, car elle implique un réel travail d'organisation (site web,

annonces) mais aussi une réflexion et un recul sur mon domaine de recherche, ainsi que de nombreuses prises de contacts avec des spécialistes du domaine. J'ai participé à l'organisation de 4 journées, autour des thèmes suivants : Emploi du temps et planning, Recherche opérationnelle et Théorie des Jeux, Approches polyédrales, Problème de sac à dos en variables 0-1. Ces journées remportent un franc succès auprès de la communauté RO d'Île de France, le nombre de participants est généralement compris entre 40 et 90.

Organisatrice d'une session à ROADEF (2003)

J'ai organisé une session "Optimisation et Tarification" en Février 2003.

Organisatrice d'une journée Tarification et Optimisation (2003)

Cette journée s'est déroulée au PRiSM en septembre 2003. Cette journée réunit chercheurs, ingénieurs, industriels concernés par le problème de tarification dans différents domaines. J'ai créé une page web pour promouvoir cette journée, j'ai diffusé l'annonce aux personnes susceptibles d'être intéressées et j'ai organisé le programme de la journée. A cette occasion, j'ai invité Mr P. Reichl du centre de recherche en télécommunications de Vienne (The Telecommunications Research Center Vienna, ftw.).

2.2.2/ EXPERTISES SCIENTIFIQUES

Relecture d'articles

- pour des articles soumis aux Journées Doctorales en Informatique et Réseaux (JDIR), aux conférences ROADEF
- pour des articles des journaux *EJOR* (European Journal of Operational Research), *JCO* (Journal of Combinatorial Optimization), *RAIRO- Operations Research*, *Sensors.*

Expertise de projets

- Experte pour une évaluation d'un projet RCE-MITACS (octobre 2008) (Réseau de centres d'excellence - Les mathématiques des technologies de l'information et des systèmes complexes) du Canada.
- Experte pour une évaluation d'un projet du programme FQRNT (octobre 2010)
  Programme Établissement de nouveaux chercheurs du Fonds québécois de la recherche sur la nature et les technologies visant une aide au démarrage de la carrière des nouveaux professeurs des universités québécoises)
  Projet de 7 pages avec budget et CV à expertiser *Simplexe en nombre entiers pour les problèmes de partitionnement*

2.3/ ENCADREMENTS

2.3.1/ ENCADREMENT DE THÈSES

- P. Saenger, thèse débutée en septembre 2014 *Stratégies de gestion d'énergie dans un aéronef* (25% encadrement)
- A. K. Idrees, thèse soutenue en décembre 2015 *Optimisation de la durée de vie du réseau de capteurs* (40% encadrement) -
  http ://www.theses.fr/s111697
  Enseignant chercheur à l'Université de Babylon (Iraq)

— A. Robert, thèse soutenue en novembre 2012 *Vers une méthodologie de structuration de la dynamique des interactions au sein du modèle Multi-Domaines et Multi-Vues. Application à la conception de familles de produits modulaires* (50% encadrement – encadrement non officiel)
http ://www.theses.fr/2012BELF0189
Conseillère Industrie de la Chambre du Commerce et de l'Industrie de la Région de Mulhouse
— F. Galea, thèse soutenue en juin 2006 *Curiethérapie : un logiciel 3D d'aide au planning de traitement*, bourse MNERT, (70% encadrement)
http ://www.theses.fr/2006VERS0040
Ingénieur-chercheur CEA Saclay
— C. Louat, thèse soutenue en Janvier 2009 *Coupes pour la PLNE mixte*, thèse CIFRE RTE (Réseau Transport Electrique).(50% encadrement)
Ingénieur consultant aéronautique Toulouse

2.3.2/ Encadrements de stages de Master

2.3.2.1/ Encadrements de stages de Master 1

— Sujet 2003 : *Problème de tarification de services hôteliers*
— Sujet 2005 : *Problème de tarification de l'espace aérien*
— Sujet 2006 (1) : *Problème d'attribution de chantiers*
— Sujet 2006 (2) : *Tarification bi-niveau en hôtellerie*
— Sujet 2008 : *Problèmes d'optimisation combinatoire inverses*

2.3.2.2/ Encadrement de stages de Master 2

— Sujet 2004 : *Problème de planification et de tarification dans un réseau de télécommunications*
— Sujet 2005 : *Problème de routage de requêtes dans un réseau MPLS*
— Sujet 2010 : *Contrôle actif d'écoulements aérodynamiques par réseaux de neurones*
— Sujet 2016 : *Application de la théorie des graphes à l'optimisation centralisée de la durée de vie des réseaux de capteurs*

2.4/ Publications

Mes publications majeures portent sur :
— le développement d'une heuristique efficace pour l'ordonnancement périodique de tâches sous contrainte de stockage [Deschinkel et al., 2011],
— le développement d'un protocole d'optimisation distribué ayant pour objectif l'allongement de la durée de vie du réseau et reposant sur la couverture du périmètre des capteurs [Idrees et al., 2016],
— le dimensionnement d'une plate-forme aéronautique utilisant plusieurs sources d'énergie (packs de batteries et de condensateurs) [Saenger et al., 2016]

Le tableau 2.1 synthétise le nombre et le type de publications.

	Journaux internationaux	Conférences internationales	Conférences nationales	Autres travaux
Pendant la thèse à l'ONERA (1998-2001)	2	3	1	4
Dans l'équipe OPALE à l'UVSQ (2002-2009)	4	11	4	3
Dans l'équipe AND (DISC, FEMTO-ST) à l'UFC (depuis 2009)	5 [Deschinkel, 2012] [Couchot et al., 2013] [Idrees et al., 2015b] [Idrees et al., 2016] [Saenger et al., 2016]	9	2	
TOTAL	11	23	7	7

TABLE 2.1 – Nombre et types de publications

3

ACTIVITÉS D'ENSEIGNEMENT ET TÂCHES COLLECTIVES

3.1/ ACTIVITÉS D'ENSEIGNEMENT

3.1.1/ A L'UNIVERSITÉ DE VERSAILLES

J'ai eu la responsabilité des enseignements suivants à l'UVSQ et à l'ISTY (Institut des Sciences et Techniques des Yvelines, école d'ingénieurs au sein de l'UVSQ) :

- Cours d'Algorithmique, ISTY 1ère année
- Cours de Recherche Opérationnelle, ISTY 2ème année
- Cours de Fondements de l'informatique 2, Licence Informatique 1ère année

Cette responsabilité comprenait l'animation de l'équipe pédagogique, le recrutement des chargés de TD pour les enseignements en vacations, ainsi que la coordination des groupes de travaux dirigés. Pour réaliser ce travail, j'ai utilisé la plate-forme collaborative *e-campus* (http://www.e-campus.uvsq.fr/) de l'UVSQ sur laquelle je mettais en ligne une partie de mes cours, des énoncés de projets, de tds et d'examens.

3.1.2/ A L'IUT DE BELFORT

Au sein du département Informatique de l'IUT, je suis responsables du module *Techniques d'Optimisation* (2ème année DUT) et du module *PPP*. Le module *Techniques d'optimisation* est une initiation au domaine de la recherche opérationnelle, les bases de la programmation linéaire et de la théorie des graphes sont enseignées. L' objectif du module *PPP* (Projet professionnel personnalisé) est de faire découvrir aux élèves, les différents parcours pour atteindre un objectif professionnel, de faire connaître les métiers de l'informatique, et de donner une idée des secteurs d'activité dans lesquels ils peuvent exercer, pour les aider ainsi à construire leurs projets. A ce titre, j'organise une dizaine de rencontres annuelles entre les étudiants du DUT Informatique et des acteurs du monde informatique (ex : anciens étudiants du DUT, représentant d'écoles d'ingénieurs en informatique, direction des services informatiques des finances publiques,...).

3.1.3/ A L'UTBM

J'enseigne en anglais un module sur la *Théorie des graphes* aux élèves ingénieur de l'Université Technologique de Belfort-Montbéliard (UTBM). Ce module présente un ensemble de définitions, de théorèmes et d'algorithmes de graphes (plus court chemin, flot maximal, coloration, arbre couvrant,...).

3.1.4/ RÉSUMÉ

Le tableau 3.1 synthétise mes activités d'enseignement, classées par thèmes.

3.2/ TÂCHES COLLECTIVES

3.2.1/ RESPONSABILITÉS PÉDAGOGIQUES

3.2.1.1/ RESPONSABLE DE LA MENTION INFORMATIQUE EN LICENCE (3 ANS : 2004-2007)

J'ai été responsable de la Mention Informatique en Licence (les 3 premières années dans le parcours LMD) appuyé par des responsables pour chaque année. J'ai contribué à la mise en place du système LMD, à l'organisation des parcours informatique et mathématiques-informatique de la licence mention informatique. Je veillais à ce que l'offre des unités d'enseignement proposées en Informatique soit cohérente et fournisse aux étudiants des connaissances de base solides en informatique (algorithmique, programmation, architecture des ordinateurs). En tant que responsable de la mention, j'assurais aussi le suivi pédagogique des étudiants en les guidant dans leur choix de parcours et d'unités d'enseignements obligatoires ou optionnelles. J'ai participé activement au recrutement sur dossier des étudiants pour la troisième année de licence. J'ai joué le rôle d'interface entre les enseignants du département d'informatique et les différents services administratifs qui gèrent les inscriptions, les emplois du temps, l'organisation des examens et des jurys.

3.2.1.2/ RESPONSABLE DES STAGES À L'ISTY (3 ANS : 2003-2006)

J'ai été responsable des stages à l'ISTY. Cette responsabilité implique la gestion d'un site web des stages, la coordination du tutorat des stages et l'organisation des soutenances de stages. J'étais l'interlocutrice privilégiée pour les entreprises qui proposent des stages à l'école, en diffusant leurs offres aux étudiants, en répondant à leur interrogations sur les modalités du stage et les exigences de l'école. J'ai été moi-même tutrice de stages et je le suis encore en DUT Informatique. Cette tâche consiste à jouer le rôle d'intermédiaire entre l'école et l'entreprise, à veiller au bon déroulement du stage, à rendre visite à l'étudiant sur son lieu de stage, à lire le rapport de l'étudiant et à participer à sa soutenance.

3.2.1.3/ CHEF DU DÉPARTEMENT INFORMATIQUE DE L'ISTY (2 ANS : 2007-2009)

En septembre 2007, j'ai été élue chef du département informatique de l'école d'ingénieurs ISTY de l'UVSQ (une soixantaine d'étudiants par promotion).

J'ai obtenu ce poste dans une période où l'ISTY devait relever un grand nombre de défis pour obtenir sa réhabilitation par la Commission des Titres d'ingénieurs (CTI, avril 2009).

J'ai eu la responsabilité d'animation et du suivi des études des 3 années du cycle d'ingénieur de la spécialité informatique, en collaboration avec les 3 responsables d'années. En plus des tâches administratives régulières (jurys, emplois du temps, conseils pédagogiques, suivi des heures d'enseignement du département), j'ai travaillé au renforcement du dispositif de classe préparatoire intégrée et à l'élaboration d'un partenariat avec les IUTs de l'UVSQ pour accroître nos effectifs. J'ai participé à des salons pour faire connaître l'école.

Je me suis efforcée aussi de tisser des passerelles entre la formation d'ingénieur et celles des autres Masters d'informatique de l'université pour diversifier l'offre.

Pour répondre à la demande de la CTI, j'ai élaboré un système d'évaluation des enseignements qui comporte des questionnaires accessibles sur le web et dont les résultats peuvent être facilement exploités sous Excel.

J'ai été également représentante du personnel enseignant au Comité d'administration de l'ISTY (2004-2008).

3.2.1.4/ JURY AUX ÉPREUVES TIPE (4 ANS, DEPUIS 2013)

Je suis jury aux épreuves orales annuelles d'évaluation des TIPE (travaux d'initiatives personnelle encadrés). A ce titre, je rédige un dossier scientifique (un article d'une quinzaine de pages et un éventail de questions/réponses) sur une problématique en mathématique/informatique. Ce dossier est préparé pendant 2h15 par le candidat qui le présente au jury pendant 10 minutes. Le jury interroge le candidat en se référant à des questions qui accompagnent le dossier. Le candidat présente ensuite au jury son travail d'initiative personnelle encadré sur la thématique scientifique de son choix. Cette épreuve est une épreuve commune aux Concours Centrale-Supélec, Concours Communs Polytechniques, et Concours Commun Mines-Ponts.

3.2.2/ RESPONSABILITÉS ADMINISTRATIVES

3.2.2.1/ MEMBRE DE COMITÉS DE SÉLECTION SECTION 27

2003- 2007 : Commission de Spécialistes MCF de l'Université de Versailles Saint Quentin en Yvelines.

2011 : Comité de Sélection MCF Université de Franche-Comté

2015 : Commission de recrutement ATER Université de Franche-Comté

2016 : Comité de Sélection MCF Université de Franche-Comté, Comité de Sélection MCF Université Haute Alsace

3.2.2.2/ MEMBRE D'INSTANCES UNIVERSITAIRES

2012-2016 (4 ans) : membre élue au **Conseil d'Institut** de l'IUT Belfort Montbéliard

2012-2016 (4 ans) : membre élue de la **Commission Recherche** du Conseil Académique (anciennement Conseil scientifique) de l'Université de Franche-Comté

2013-2016 (3 ans) : membre nommée du **Conseil Académique Restreint** de l'Université de Franche-Comté

2015-2016 (1 an) : membre nommée du **Conseil Académique provisoire** de la **COMUE** (Communauté d'universités et d' établissements "Université Bourgogne Franche-Comté")

3.2.2.3/ MEMBRE D'INSTANCE NATIONALE

2016-2020 (4 ans) : membre élue à la **Commission Nationale des Universités (CNU)** en section 27

3.3/ RÉSUMÉ DE CARRIÈRE

La frise 3.2 résume ma carrière en enseignement et en recherche.

Période	Établissement / UFR	Nom du cours / TD / TP	Statut	Type de public	Volume horaire Annuel
		Algorithmique – Programmation			
2009-2010	I.U.T. Belfort	TD/TP **Algorithmique + Projet**	MCF	30 élèves	40h
2002-2009	ISTY	Cours **Algorithmique + Projet**	MCF	50 Élèves Ingénieur 1ère année	60 h
		Cours **Algorithmique**	MCF	30 Élèves Prépa intégrée 1ère année	18h
		TD **Systèmes (Programmation C)**	MCF	50 Élèves Ingénieur 1ère année	36h
		Cours/TD Mise à niveau **Algorithmique/C**	MCF	15 Élèves Ingénieur 1ère année	20h
2004-2009	UVSQ	Cours **Fondements de l'Informatique 2**	MCF	130 étudiants en L1	18h
2001-2002	Université Paris 1	TD **Algorithmique** et **Programmation** (Pascal)	ATER	25 étudiants DEUG MASS (2ème année)	54 h
		Cours/TD **Méthodologie Informatique** : Algorithmique	ATER	25 étudiants DEUG MASS (1ère année)	54 h
1999-2001	ENSAE	Projet TrEx (Travaux expérimentaux) **Programmation en C++** de jeu à 2 joueurs	Vacataire	2 groupes de 6 étudiants	60 h
		Bases de Données			
2012-2016	I.U.T. Belfort	TD **Bases de données**	MCF	30 élèves	40h
2009-2012	I.U.T. Belfort	TD/TP Analyse et conception des **systèmes d'information (ACSI)**	MCF	30 élèves	90 h
2009-2012	I.U.T. Belfort	TD/TP **Bases de données** + Projet	MCF	30 élèves	80 h
2001	Université Paris 1	TP **Bases de données**	ATER	30 étudiants Maîtrise Sciences et Gestion	54 h
		Recherche Opérationnelle			
2010-2016	I.U.T. Belfort	Cours TD/TP T**echniques d'Optimisation**	MCF	60 élèves	60 h
2011-2016	UTBM	Cours TD/TP**Graph Theory** (enseignement en anglais)	MCF	32 élèves en génie Informatique	60 h
2014-2016	UTBM	Cours TD/TP**Graph Theory** appliqué à la génération de séquences d'assemblage en conception mécanique (enseignement en anglais)		60 élèves en génie Mécanique et Conception	20h
2002-2009	ISTY	Cours / TD **Recherche Opérationnelle** dont **Théorie des graphes**	MCF	50 Élèves Ingénieur 2ème année	54 h
2002-2009	UVSQ	TD **Recherche Opérationnelle**	MCF	30 étudiants Master1 M1 info.	36 h
2002-2009	UVSQ	Cours **Programmation linéaire avancée Méthode exacte et méthaheuristiques**	MCF	10 étudiants Master2 Math-Info	12 h
2005-2009	UVSQ	Cours d'**Introduction à la Recherche Opérationnelle**	MCF	50 étudiants de L1 avec spécialité info.	3 h
2003-2009	ENSTA	Cours de **Métaheuristiques avec population** (Algorithmes génétiques, Colonies de fourmis, Scatter Search)	MCF	10 élèves de 3ème année Filière Optimisation et RO	4 h
2001- 2002	Université Paris 1 UFR Maths/info	TD Informatique et Algorithme 2 : **Théorie des graphes**	ATER	25 étudiants Maîtrise MASS	24 h
1999-2001	ENSEEIHT	Cours d'**Optimisation non linéaire**	Vacataire	25 Élèves Ingénieur 2ème année	8 h

FIGURE 3.1 – Résumé de mes enseignements

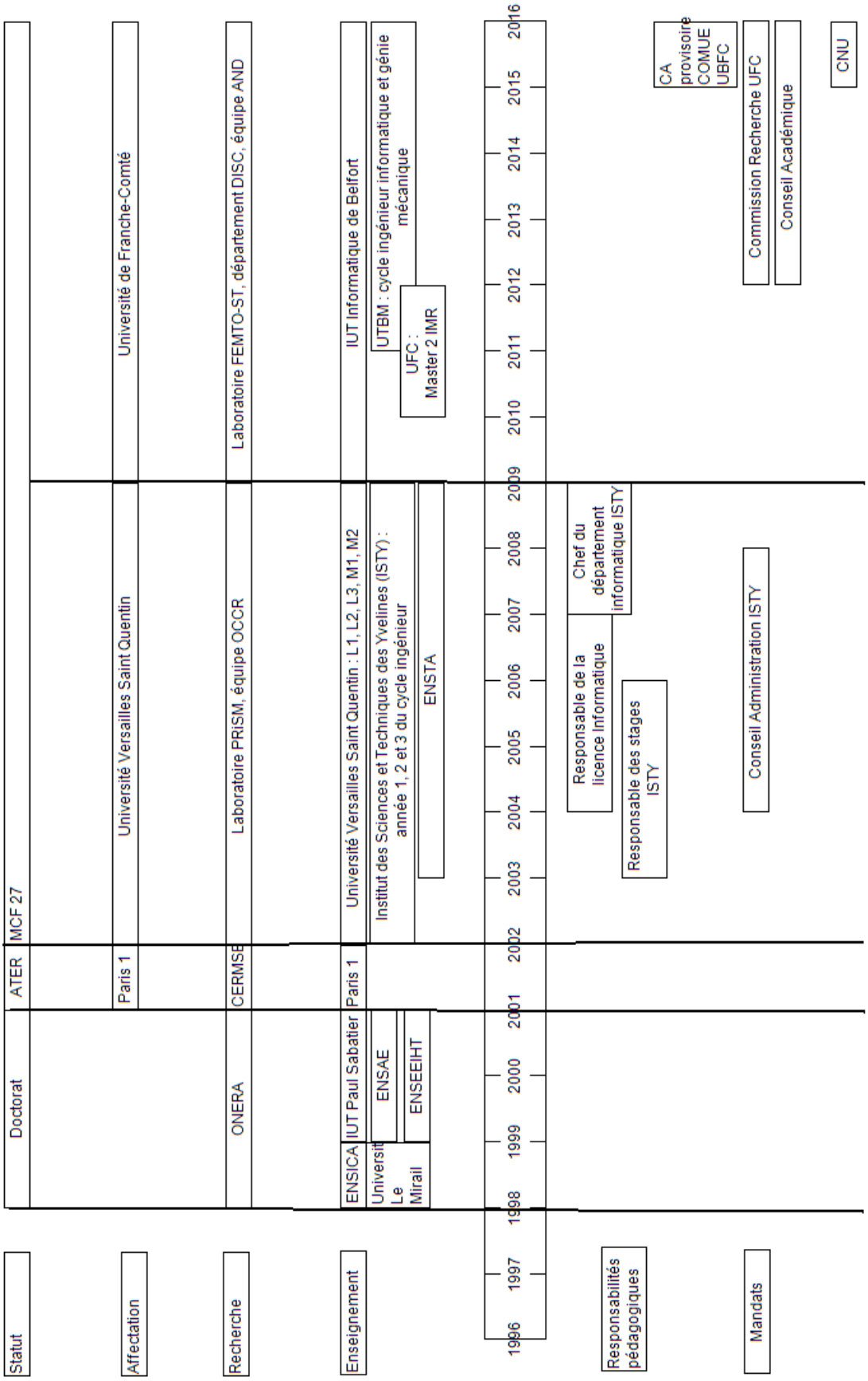


FIGURE 3.2 – Résumé de mon parcours professionnel

4

MOTIVATIONS ET ORGANISATION DU MÉMOIRE

4.1/ MOTIVATIONS

La recherche opérationnelle (RO) est une discipline au confluent de l'informatique, des mathématiques appliquées, de l'économie et du génie industriel. Elle permet de fournir des bases scientifiques à la prise de décisions, habituellement dans un but de contrôle ou d'optimisation (améliorer l'efficacité, diminuer les coûts, etc.). Les applications de la recherche opérationnelle sont nombreuses et se retrouvent par exemple dans le domaine du transport, de la finance ou de la santé. La RO s'appuie sur de nombreuses méthodes issues des mathématiques appliquées et sur les progrès en informatique pour essayer de résoudre en un temps de calcul raisonnable des problèmes NP-difficiles. Beaucoup de ces méthodes sont génériques, mais dans la plupart des cas, un travail conséquent de modélisation et de décomposition du problème, est nécessaire pour mettre en oeuvre de manière performante une telle méthode sur un problème donné. Parmi les outils de modélisation à disposition des experts en recherche opérationnelle, on trouve entre autres la programmation linéaire et la théorie des graphes. La programmation linéaire, en variables réelles ou entières, est fortement liée à l'histoire de la recherche opérationnelle. En effet, l'algorithme du simplexe créé en 1947 pour la résolution efficace de programmes linéaires est à l'origine du développement de la RO. La programmation linéaire consiste à minimiser (ou maximiser) une fonction linéaire sous des contraintes également linéaires, ce qui, en pratique, permet de modéliser un grand nombre de situations. La théorie des graphes offre également des possibilités de modélisation très riches (plus courts chemins, flux de transports, ordonnancement de tâches). De nombreux problèmes d'échange d'informations (réseaux d'entreprises, internet), de transfert de marchandises peuvent se modéliser comme un problème de flot dans un graphe.

Mes travaux en recherche opérationnelle sont principalement tournés vers la modélisation de problèmes d'optimisation difficiles dans des domaines d'applications variés (aérien, télécommunication, médecine, compilation,...).

Dans ce mémoire, nous utilisons le terme générique *problème d'optimisation difficile* pour désigner indistinctement :

- un problème d'optimisation pour lequel les difficultés liées à sa modélisation sont nombreuses : nombre de paramètres élevé, présence simultanée de plusieurs objectifs parfois contradictoires, évaluation complexe de la fonction objectif,...
- un problème d'optimisation pour lequel on ne trouve pas d'algorithme en temps

polynomial pour le résoudre. Ce type de problème d'optimisation est dit NP-complet [1].

Je m'intéresse aussi bien au développement de résultats théoriques qu'à la résolution de problèmes réels dans le cadre de contrats industriels ou institutionnels (ARC PrixNet, Hopital Pitié Salpétrière, bourse CIFRE RTE). Mes travaux relèvent de différentes approches : synthèse, étude de complexité, théorie des graphes, optimisation inverse, programmation à deux niveaux, résolution par méthodes exactes (simplexe, branch and bound, méthodes de coupes, relaxation lagrangienne, décomposition,...) ou par algorithmes approchés (recuit simulé, tabou, algorithme génétique,...).

Mes travaux de recherche m'ont également amenée à utiliser des solveurs de programmation linéaires commerciaux ou libres (CPLEX, XPRESS, GLPK, LPSOLVE, COIN OSI, BOBPP) et un simulateur de réseaux de capteurs (OMNET++). J'ai aussi contribué au développement et la réalisation de nouveaux logiciels : Isodose 3D pour la curiethérapie, GLOP pour l'intégration de méthodes de coupes dans les solveurs de programmation linéaire, et ORASSE pour la conception de produit en mécanique.

L'ensemble de mes travaux témoigne de ma capacité d'ouverture vers les autres disciplines pour dialoguer avec des acteurs d'horizons scientifiques variés (télécommunications, médecine, électricité, énergie, compilation,...) pour comprendre leurs problématiques, créer des modèles mathématiques et informatiques appropriés et proposer des méthodes de résolution efficaces. En guise d'illustration, j'ai fait le choix de centrer ce mémoire sur trois problèmes significatifs dans ma carrière d'enseignant-chercheur :

- Le problème d'optimisation de la dose en curiethérapie,
- Le problème d'optimisation de registres en compilation,
- Le problème de maximisation de la durée de vie d'un réseau de capteurs.

Le premier problème traité touche au domaine de la santé et m'a permis d'appréhender toutes les phases de développement d'une solution en recherche opérationnelle, en passant par une première étape de modélisation, où il faut s'imprégner d'un problème concret dans une discipline nouvelle par rapport à sa formation, pour finir par une étape de réalisation logicielle. Le second problème m'a donné l'occasion d'aborder une nouvelle thématique en informatique, celle du processus de compilation en présence de parallélisme d'instructions. La difficulté a été d'appréhender un cadre théorique complexe, mais cet effort a été récompensé par l'obtention d'excellents résultats grâce à une heuristique appropriée. Enfin, en m'intéressant au problème de couverture et de durée de vie dans un réseau de capteurs, j'ai rejoint un domaine de recherche de mon équipe actuelle, et j'ai réussi à proposer de nouveaux modèles et de nouvelles heuristiques pour le traitement de ce problème. Ces trois problèmes ont en commun d'être abordés sous l'angle de la modélisation par programmation linéaire et/ou par la théorie des graphes.

4.2/ PLAN DU MÉMOIRE

Ce mémoire est organisé en 4 parties.

1. On utilise ici et dans le reste du mémoire un abus de langage car on devrait utiliser le terme NP-difficile pour un problème d'optimisation et NP-complet pour le problème de décision associé.

Partie I : Modélisation de problèmes d'optimisation difficiles

Cette partie est consacrée à la présentation des trois problèmes précédemment cités. Cette partie se décline en trois chapitres (un chapitre par problème, chapitres 5,6 et 7), chacun structuré de la même manière : une présentation de la problématique, un état de l'art, et pour finir une description détaillée de nos propres modèles. Tous nos modèles sont des modèles de programmation linéaire : un programme linéaire à variables continues (PL) en curiethérapie, un programme linéaire à variables entières (IP) en compilation, et des programmes linéaires mixtes (MIP) ou (IP) pour la maximisation de la durée de vie du réseau de capteurs.

Partie II : Résolution par décomposition du problème

La seconde partie présente nos heuristiques de résolution fondées sur un même principe, celui de décomposer le problème en sous-problèmes présentant des structures remarquables pour lesquelles des algorithmes en temps polynomial s'avèrent efficaces. Nous décrivons les approches conçues pour résoudre les problèmes, ainsi que les résultats expérimentaux obtenus.

Le chapitre 8 présente l'heuristique SIRALINA dédiée à la minimisation du nombre de registres. Elle se déroule en deux étapes : résolution d'un problème d'ordonnancement puis d'un problème d'affectation linéaire.

Les chapitres 9 et 10 se concentrent sur la résolution du problème de maximisation de la durée de vie d'un réseau de capteurs. Nous avons proposé deux méthodes centralisées de planification des périodes de réveil des capteurs. Nous avons développé une heuristique de construction en parallèle de différents ensembles couvrants disjoints (chapitre 9). La répartition des différents capteurs dans les ensembles est guidée par la résolution d'un problème d'affectation linéaire.

Dans le cas d'ensembles couvrants non disjoints, le nombre de combinaisons possibles augmente de manière exponentielle avec le nombre de capteurs et de cibles. Pour faire face, nous proposons une méthode de génération de colonnes (chapitre 10) qui nécessite la résolution d'un programme auxiliaire en nombre entiers. Ce problème auxiliaire est résolu, soit de manière exacte avec un algorithme type Branch and Bound, soit avec une heuristique gloutonne.

Partie III : Modélisation sous forme de problèmes de flots

Dans la troisième partie de ce mémoire, nous introduisons des nouvelles modélisations à base de problèmes de flots dans un graphe.

Dans le premier chapitre de cette partie (chapitre 11), nous montrons qu'il est possible de ramener la résolution d'un des sous-problèmes de SIRALINA à celle d'un problème de flot à coût minimum pour lequel plusieurs algorithmes polynomiaux sont connus, ce qui permet ainsi de s'affranchir de l'utilisation d'un solveur de programmes linéaires.
Dans le chapitre suivant (chapitre 12), nous détaillons un modèle de graphe associé à un problème de flot maximal qui fait office de référence pour le problème de génération d'ensembles couvrants disjoints. Nous nous intéressons à une modélisation limitant

simultanément la sur-couverture et la sous-couverture pour ce même problème. Nous montrons qu'il est possible d'interpréter ce modèle comme celui d'un problème de flot à coût minimum dans le graphe précédemment construit.

Une conclusion et des perspectives sont données en dernière partie.

I

MODÉLISATION DE PROBLÈMES D'OPTIMISATION DIFFICILES

5

LE PROBLÈME D'OPTIMISATION DE LA DOSE EN CURIETHÉRAPIE

5.1/ PROBLÉMATIQUE

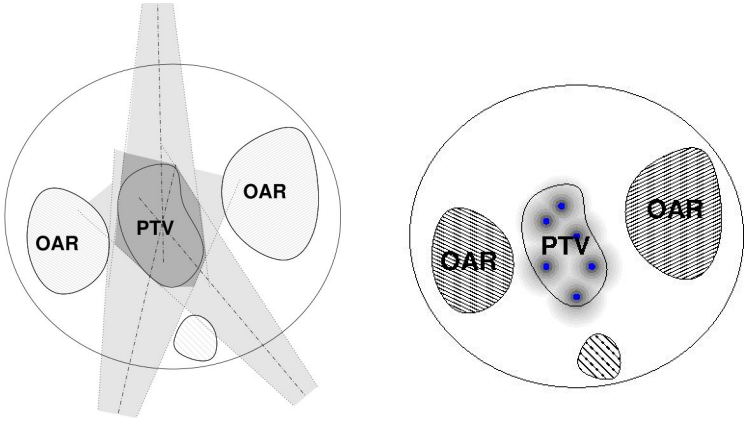
5.1.1/ PRÉSENTATION DE LA CURIETHÉRAPIE

La radiothérapie est une des techniques les plus efficaces de lutte contre le cancer en particulier pour les tumeurs malignes localisées. A la différence de la radiothérapie externe où la tumeur est irradiée par la conjonction de faisceaux d'électrons (figure 5.1(a)), la curiethérapie (appelée également brachythérapie) est un type de radiation qualifiée d'interne. Les sources radioactives sont en effet introduites par voie opératoire à l'intérieur du corps du patient (figure 5.1(b)). Cette technique de traitement s'est développée ces dernières années grâce aux avancées technologiques en imagerie, et aux outils et équipements informatiques mis à disposition des radio-physiciens et des médecins. Il existe en fait deux types de curiethérapie. La curiethérapie à bas débit de dose dite LDR pour Low Dose Radiation (débit de dose de référence compris entre 0.4 et 2 Gray(Gy)/h) [1] et la curiethérapie à haut débit de dose dite HDR (débit de dose de référence supérieure à 12 Gy/h). Les objectifs de ces deux types de curiethérapie sont les mêmes, c'est-à-dire détruire la tumeur par irradiation en délivrant une dose suffisante pour détruire les tissus tumoraux sans surdosage susceptible d'entraîner des effets secondaires et tout en protégeant les organes à risque (comme le cerveau, la vessie, etc.) et les tissus sains.

La curiethérapie à bas débit de dose dite LDR pour Low Dose Radiation est très utilisée pour les cancers de la prostate. Dans le cas d'une curiethérapie par implant pour une tumeur prostatique, une source de faible énergie est insérée dans le corps humain à travers une aiguille et ensuite l'aiguille est retirée rapidement, laissant la source sous forme d'un 'grain' dans le patient. Les sources ont dans ce cas une durée de vie limitée : après peu de mois, il n'y a plus de radioactivité. Il s'agit donc 'd'implants permanents pour cancers prostatiques'.

La curiethérapie HDR consiste à introduire temporairement dans le corps du patient par voie opératoire une source radioactive. La source se déplace à travers un cathéter (tube étroit) et s'arrête à différents intervalles. Elle est en fait contrôlée par une machine

1. la dose absorbée par l'organe se mesure en Gray(Gy), telle que 1 Gy correspond à une énergie absorbée de 1 joule dans une masse de matière de 1kg



(a) La radiothérapie externe (b) La curiethérapie

FIGURE 5.1 – Les deux sortes de thérapies par rayonnements ionisants : en radiothérapie externe, le PTV est irradié par des faisceaux de rayonnements provenant d'une source placée à l'extérieur du patient. En curiethérapie, les rayonnements sont émis par des sources radioactives placées à l'intérieur du patient.

programmée qui s'appelle un *remote afterloader*, et qui pousse la source radioactive à haut débit (Iridium Ir192) dans chaque cathéter, en s'arrêtant à des positions calculées dites positions d'arrêt espacées par exemple de 2,5 ou 5 millimètres et pour un temps prescrit. La distribution de dose peut donc être modulée grâce aux nombreuses positions d'arrêt en y laissant plus ou moins longtemps la source. Les doses finales reçues peuvent être évaluées avant de commencer tout traitement. Un autre avantage est qu'une fois la curiethérapie terminée le patient n'est plus radioactif. Le traitement consiste alors en un petit nombre de séances (3 ou 10) de quelques minutes (10/15), une fois par semaine.

Nous détaillons ci-dessous les différentes phases du traitement en curiethérapie HDR pour bien montrer où se posent les problèmes d'optimisation.

5.1.2/ PHASES DE TRAITEMENT

PLACEMENT DES CATHÉTERS DANS LE CORPS DU PATIENT

Le médecin, après avoir localisé par imagerie 3D la tumeur et défini ce qu'on appelle le volume cible PTV (Planning Target Volume), choisit le type d'implantation des cathéters appropriée pour le type de tumeur à soigner : dans une cavité (cancer gynécologique ou buccaux) ; dans un tube (bronches, œsophage) ou à travers la tumeur (sein, prostate,...).

SIMULATION

Le radiophysicien recalcule l'exacte position des cathéters après leur implantation, localise la tumeur et les organes à risque adjacents OAR (Organs At Risk) par imagerie 3D (scanographie, rayons X, ultrasons, tomographie, imagerie par résonance magnétique,...).

DOSIMÉTRIE

Les doses à prescrire sont alors définies. Une prescription peut par exemple donner la dose minimale pour couvrir le volume cible PTV, les doses maximales à ne pas dépasser pour les organes à risque OARs. Les positions d'arrêt ainsi que le temps d'arrêt associés doivent alors être calculés. C'est cette partie qui est automatisée dans les logiciels de plan de traitement.

TRAITEMENT

Dès que le traitement proposé est approuvé, les données sont transférées à la machine d'émission de source qui applique le traitement sur le patient. Après le traitement, les applicateurs sont retirés du corps du patient.
Un plan de traitement consiste à trouver le nombre et la position des cathéters ainsi que les positions et les temps d'arrêt de la source radioactive dans chacun d'eux. Le meilleur plan de traitement est celui qui produit une distribution de dose remplissant au mieux les critères cliniques prescrits qui varient suivant le type de tumeur traitée. Nous verrons qu'il n'existe pas de critère consensuel pour évaluer un plan de traitement mais que l'optimisation peut aider à construire un plan de traitement en intervenant sur différentes phases du traitement : l'implantation des cathéters et la détermination des positions d'arrêt dans un premier temps, puis le calcul des temps d'arrêt. A ces deux problèmes d'optimisation liés entre eux s'ajoute un problème annexe : le meilleur choix de points de référence pour représenter une distribution de dose. Ce problème ne sera pas étudié ici.

5.1.3/ CALCUL D'UNE DISTRIBUTION DE DOSE

Nous expliquons comment est calculée une distribution de dose avant de décrire comment elle peut être évaluée et optimisée.

POINTS DE RÉFÉRENCE

Pour calculer l'irradiation des tissus du patient pour un plan de traitement donné, un certain nombre de points de référence vont être choisis sur les organes en cause. Ils peuvent être distribués sur une grille de points régulièrement espacés, ou choisis au hasard avec une loi uniforme, ou encore suivre une certaine loi de distribution. Le nombre de points de référence à l'intérieur ou sur le pourtour d'un organe (PTV ou OAR) a son importance et va jouer sur la fiabilité des résultats obtenus. Un grand nombre de points permet de mieux rendre compte de la dose distribuée dans l'organe mais va allonger le temps de calcul de la distribution de dose. Il y a donc là un problème d'optimisation annexe puisqu'il faudrait calculer le nombre minimal de points pour rendre compte de la distribution de dose et donc faire appel à des méthodes d'analyse statistique. Ce problème a été évoqué mais rarement traité dans la littérature.

Lessard *et al.* [Lessard and Pouliot, 2001] génèrent uniformément des points sur le pourtour, et à l'intérieur des organes à risque, et du PTV. Lahanas *et al.* [Lahanas et al., 2003] le font également à l'intérieur des OARs, du PTV, et des tissus normaux et sur la surface triangulée du PTV. Dans [Karouzakis et al., 2002] Karouzakis *et al.* proposent deux variantes d'une méthode d'échantillonage stratifié pour réduire le nombre de points calculés sans perte de précision pour l'optimisation de la dose.

Les points de référence ne servent pas uniquement lors de l'optimisation de la dose, mais également pour évaluer la dose une fois l'optimisation effectuée. Ces deux opérations sont très différentes du point de vue des traitements effectués : alors que dans le premier cas des méthodes d'optimisation complexes sont exécutées, dans le second cas n'est effectué que le calcul séquentiel d'une grande quantité de valeurs. Pour un même nombre de points, l'optimisation des temps d'arrêt est une opération beaucoup plus longue (en termes de temps d'exécution) que la dosimétrie *a posteriori*. Il est donc possible, et même souhaitable, d'utiliser un ensemble plus grand de points de référence dans ce second cas. Cela permet de mesurer la qualité d'une optimisation avec beaucoup plus de précision, et éventuellement, de valider ou non le choix des points utilisés pour l'optimisation.

CALCUL DE LA DISTRIBUTION DE DOSE

Le débit de dose reçu en point de calcul i à partir d'une source ponctuelle situé en un point j est donné par la formule suivante [Gerbaulet et al., 2002] :

$$d(i,j) = S_k \Lambda \phi_{an} g(dist(i,j))/dist(i,j)^2,$$

où S_k est l'intensité (kerma dans l'air) de la source, Λ est la constante de dose de la source, ϕ_{an} est le facteur d'anisotropie, $dist(i,j)$ est la distance euclidienne entre les points i et j, $g(dist(i,j))$ est la fonction de dose radiale. Cette fonction, dépendante de la source utilisée et fournie par le fabricant de la source, peut être en général approximée par une fraction de 2 polynômes. La dose reçue par un point i décroît comme l'inverse du carré de sa distance à la source. Cette formule montre évidemment que plus on est près de la source, plus la dose de radiation reçue augmente. Le débit de dose exprime la dose reçue par unité de temps. Pour un point i irradié par plusieurs sources ou de façon équivalente, par une source se déplaçant sur ensemble J de positions avec un temps d'arrêt t_j en chaque position, comme en curiethérapie HDR, la dose reçue est :

$$d(i) = \sum_{j \in J} d(i,j).t_j.$$

En curiethérapie LDR, lorsque toutes les sources sont implantées de façon définitive (par exemple, traitement de la prostate par des grains d'iode 125), la dose reçue en un point n'est plus une somme de produits "contribution de débit de dose $\times$ temps" mais une somme des contributions de dose, en considérant la présence ou non d'une source en chaque position j. La variable x_j est une variable qui vaut 1 si une source est en placée en j, et 0 sinon.

$$d(i) = \sum_{j \in J} d(i,j).x_j.$$

ÉVALUATION QUALITATIVE D'UNE DISTRIBUTION DE DOSE

Être capable de calculer la dose en un point quelconque du volume considéré n'est pas suffisant du point de vue des médecins, car cela entraîne la génération d'une quantité de données bien trop grande pour pouvoir être évaluée de manière judicieuse. Des outils

graphiques (Histogrammes dose-volume HDV) et numériques (indices d'homogénéité, COIN, *etc.*) ont été introduits, permettant aux médecins d'obtenir rapidement un aperçu de la qualité d'une simulation de traitement, et par là-même de décider de l'application de tel ou tel traitement proposé.

Au sein de la communauté médicale, il n'y a pas de critères unanimes pour rendre compte qualitativement de la distribution d'une dose générée par un plan de traitement spécifique. Globalement, un bon plan de traitement doit permettre d'obtenir une distribution de dose qui recouvre totalement ou presque le volume cible avec une dose au moins égale à la dose prescrite, tout en minimisant la dose reçue dans l'espace environnant, en particulier dans les éventuels organes à risque.

Les critères permettant de juger cette distribution de dose vont concerner les quantités de doses délivrées (respect des prescriptions, enveloppes isodose) et leur répartition à travers les différents organes (histogramme dose-volume, paramètres d'uniformité). Pour plus de détails, le lecteur peut se référer au chapitre 2 de la thèse de F. Galéa [Galea, 2006]. Ces critères sont rarement utilisés dans les modèles d'optimisation de la dose, ils sont utilisés à posteriori pour évaluer la qualité de la distribution de la dose.

5.2/ ÉTAT DE L'ART

S'il existe de nombreux articles dans des revues de Recherche Opérationnelle concernant l'optimisation d'un traitement par radiation externe, le problème du traitement par curiethérapie est plutôt abordé dans des revues de Radiophysique ou de Médecine. Cette partie fait une synthèse des modèles et méthodes existantes sur ce problème avant nos travaux (c'est-à-dire jusqu'en 2005). Ces méthodes s'appuient sur des techniques d'imagerie médicale (scanner, IRM,...) qui permettent de situer avec précision la forme de la tumeur à traiter. Les images sont des vues en coupe, parallèles, entre elles du volume à traiter. Le problème principal consiste à déterminer la façon de placer les sources actives à l'intérieur du volume à traiter (PTV), de manière à avoir une distribution de dose correcte en termes de conformation (enveloppement de la tumeur) et d'homogénéité. Nous allons effectuer une synthèse sur les différents modèles et méthodes de résolution employés en curiethérapie LDR prostatique, et en curiethérapie HDR.

D'une manière générale, la question posée en curiethérapie LDR prostatique est : "étant donnée une position possible de source, doit-on y placer une source ou non ?". En curiethérapie HDR, il s'agira plutôt de déterminer, pour chacune des positions possibles de la source, combien de temps la source devra rester sur cette position. Dans le premier cas, les problèmes évoqués sont donc des problèmes à variables binaires ; A chaque position possible de la source est associée une variable binaire x_j, dont la valeur est mise à 1 si une source doit être placée sur la position j, et 0 dans le cas contraire. Dans le second cas, en curiethérapie HDR, les variables sont continues et correspondent au temps d'arrêt t_j (éventuellement nul) de la source en une position j. Pour tous les cas, les principes d'optimisation proposés dans la littérature reposent sur des considérations proches : il s'agit de spécifier pour chaque point de l'espace un intervalle de dose à respecter et de définir des pénalités en cas de dépassement de l'une ou l'autre des bornes.

Plusieurs objectifs se retrouvent en conflit pour irradier convenablement la tumeur tout en protégeant les organes à risques. La présence de plusieurs objectifs en conflit donne lieu

à deux approches distinctes. La première consiste à regrouper tous les objectifs dans une seule fonction en pénalisant avec des poids plus ou moins forts pour privilégier l'un ou l'autre des critères et fait ressortir une seule solution. La seconde approche fait appel à des techniques d'optimisation multiobjectifs qui fournissent un ensemble de solutions traduisant les possibilités de compromis possibles. Dans le premier cas, l'expertise du planificateur intervient en amont de l'optimisation pour fixer les différents poids. Dans le second cas, le planificateur doit analyser toutes les solutions proposées après optimisation pour faire le choix d'une solution appropriée. Cette tâche peut devenir rapidement complexe quand le nombre d'objectifs est élevé, d'où l'intérêt de proposer, en complément, des outils graphiques de visualisation de solutions, comme ceux développés par Hamacher [Hamacher et al., 2005] pour faciliter le travail.

En ce qui concerne les méthodes d'optimisation utilisées, la méthode du Simplexe (et/ou Branch and Bound pour des variables entières) est clairement adaptée pour la résolution du problème, quand le modèle associé est linéaire, car contrairement au recuit simulé, elle fournit la solution optimale exacte, et de plus, les solveurs de programmation linéaire sont suffisamment puissants pour traiter des problèmes de taille semblable à ceux rencontrés en curiethérapie. Pour des objectifs particuliers avec une expression analytique explicite comme la minimisation des variances de dose, et convexe, la méthode de descente du gradient est bien appropriée. Pour le reste, il faut se tourner vers des méthodes de résolution approchée. Les algorithmes génétiques ont été étudiés et largement utilisés dans ce cadre, ils répondent bien à l'approche multiobjectifs et donnent de bons résultats. On peut ajouter que l'ensemble des méthodes implémentées offre des temps de calculs raisonnables pour être intégrées, comme c'est déjà parfois le cas, dans des logiciels fournissant des plans de traitement.

Nous donnons ici les principaux modèles proposés en curiethérapie LDR et en curiethérapie HDR reflétant les différentes approches précédemment décrites.

5.2.1/ MINIMISATION DU NOMBRE DE VIOLATIONS

Il s'agit simplement de minimiser le nombre de violations, c'est-à-dire le nombre de fois où la dose calculée en un point de référence sera en dehors de l'intervalle de dose prescrit par le physicien.

En curiethérapie LDR, Gallagher et Lee [Gallagher and Lee, 1997] [Lee et al., 1999] proposent le programme linéaire suivant :

$$\begin{cases} \max & \sum\limits_{i\in P} v_i + w_i & \\ \text{sous :} & \sum\limits_{j\in J} m_{i,j}.x_j - \underline{D}_i \geq -N_i(1-v_i) & \forall i \in P \\ & \sum\limits_{j\in J} m_{i,j}.x_j - \overline{D}_i \leq M_i(1-w_i) & \forall i \in P \\ & v_i, w_i \in \{0,1\} & \forall i \in P \\ & x_j \in \{0,1\} & \forall j \in J \end{cases}$$

Pour chaque point de référence i (P est l'ensemble des points de référence), les valeurs des paramètres N_i et M_i sont positives et choisies pour être suffisamment grandes pour que les contraintes soient toujours réalisables quand les valeurs v_i ou w_i correspondantes valent 0 (cas où la contrainte de dose n'est pas respectée pour i). $\underline{D}_i$ et $\overline{D}_i$ sont les bornes respectivement inférieure et supérieure de l'intervalle de dose pour le point i. La fonction

objectif consiste à maximiser le nombre de fois (le nombre de points de référence) où la dose reçue est dans l'intervalle de dose prescrit.

En curiethérapie HDR, Lahanas *et al.* [Lahanas et al., 2003] proposent un ensemble d'objectifs concernant le recouvrement complet du volume cible par une dose spécifiée et une protection des organes à risque adjacents. Les fonctions objectif sont les suivantes :

$$f_L^{PTV} = \frac{1}{N_{PTV}} \sum_{i=1}^{N_{PTV}} \theta[D_L^{PTV} - d_i],$$

$$f_H^{PTV} = \frac{1}{N_{PTV}} \sum_{i=1}^{N_{PTV}} \theta[d_i - D_H^{PTV}],$$

$$f_{OAR}^{j} = \frac{1}{N_{OAR}^{j}} \sum_{i=1}^{N_{OAR}^{j}} \theta[d_i - D_{crit}^{j}],$$

avec $\theta[u] = 1$ si $u > 0,$ et 0 sinon.

D_L^{PTV} est la dose prescrite, ou la dose limite inférieure et D_H^{PTV} est la dose limite supérieure pour le volume cible. N_{PTV} et N_{OAR}^{j} représentent les nombres de points de référence pris dans le PTV et dans chaque OAR j. Lahanas *et al.* [Lahanas et al., 2003] traitent la résolution du problème par une approche multiobjectifs. Ils proposent un algorithme génétique, les calculs sont effectués sur une machine 933 MHz Intel III pour une implantation avec 125 positions d'arrêt et 2500 points de référence (500 répartis aléatoirement dans le PTV et les OARs, et 2000 points dans les tissus sains adjacents). Ils obtiennent 1000 à 2000 solutions de l'ensemble Pareto-optimal en moins de cinq minutes. Le planificateur du traitement fait face alors un choix difficile parmi toutes les solutions qui lui sont offertes.

5.2.2/ MINIMISATION LINÉAIRE DES ÉCARTS DE DOSE

Ce problème proposé également par Lee *et al.* [Lee and Zaider, 2003] en curiethérapie LDR consiste à minimiser une somme pondérée de déviations par rapport à l'intervalle de dose. Pour chaque point $i \in P$, deux variables continues positives y_i et z_i sont introduites, représentant les écarts par rapport aux bornes respectivement inférieure et supérieure de l'intervalle de dose. Des paramètres α_i et β_i sont en outre utilisés, pour influer sur l'importance du respect de l'une ou l'autre des bornes, pour un point ou un groupe de points donnés.

Le modèle est le suivant :

$$\begin{cases} \min & \sum_{i \in P} \alpha_i y_i + \beta_i z_i & \\ \text{sous :} & \sum_{j \in J} m_{i,j}.x_j + y_i \geq \underline{D}_i & \forall i \in P \\ & \sum_{j \in J} m_{i,j}.x_j - z_i \leq \overline{D}_i & \forall i \in P \\ & y_i, z_i \geq 0 & \forall i \in P \\ & x_j \in \{0,1\} & \forall j \in J \end{cases}$$

Ce modèle permet de s'assurer que l'ensemble des écarts de dose ne soit pas trop élevé. La résolution du programme linéaire mixte est réalisée de manière exacte avec

une méthode Branch and Bound et permet d'obtenir des solutions optimales en un temps de calcul raisonnable.

Un modèle de pénalisation des écarts de dose a été proposé par Lessard *et al.* [Lessard and Pouliot, 2001] pour le traitement par curiethérapie HDR du cancer de la prostate. Les fonctions de pénalités sont les suivantes :

$$f_L^o = \frac{1}{N_o} \sum_{i=1}^{N_o} \theta[D_L^o - d_i].(D_L^o - d_i),$$

$$f_H^o = \frac{1}{N_o} \sum_{i=1}^{N_o} \theta[d_i - D_H^o].(d_i - D_H^o),$$

avec $\theta[u] = 1$ si $u > 0,$ et 0 sinon.
N_o est le nombre de points de référence qui appartiennent au volume o, et D_L^o, D_H^o les limites de dose respectivement inférieures et supérieures pour la structure o. Ces fonctions mesurent la manière dont les critères cliniques spécifiés par le médecin sont respectés, notamment le respect de la dose prescrite sur le volume cible, le non dépassement d'une dose maximale pour l'urètre et une irradiation très faible des tissus sains autour de la prostate. Les auteurs [Lessard and Pouliot, 2001] ont préféré se tourner vers une méthode de recuit simulé pour obtenir des plans de traitement efficaces. L'utilisation d'un recuit simulé, plutôt qu'une résolution exacte ne nous semble pas suffisamment justifiée. D'une part, la méthode du Branch and Bound, contrairement au recuit simulé, fournit la solution optimale pour le modèle considéré. Et d'autre part, les solveurs disponibles actuellement en programmation linéaire sont suffisamment performants pour traiter rapidement des problèmes de grosse dimension semblables à ceux rencontrés en curiethérapie HDR. De plus, certains de ces solveurs sont accessibles dans des logiciels libres.

5.2.3/ MINIMISATION QUADRATIQUE DES ÉCARTS DE DOSE

Dans ce type de modèle, l'idée est de pénaliser, encore plus, les fortes déviations pour essayer d'obtenir une distribution de dose plus homogène dans chaque organe. En curiethérapie HDR, Lahanas *et al.* [Lahanas et al., 2003] utilisent la fonction f_{NT} comme la moyenne des carrés des doses dans les tissus sains autour de la tumeur (N_{NT} est le nombre de points de référence associé) :

$$f_{NT} = \frac{1}{N_{NT}} \sum_{i=1}^{N_{NT}} d_i^2.$$

Milickovic *et al.* [Milickovic et al., 2002] proposent un modèle quadratique qui se concentre sur la distribution de dose dans le PTV et à sa surface par la minimisation des variances :

$$f_S = \frac{1}{N_S} \sum_{i=1}^{N_S} \frac{(d_i^S - m_S)^2}{m_S^2},$$

$$f_V = \frac{1}{N_V} \sum_{i=1}^{N_V} \frac{(d_i^V - m_V)^2}{m_V^2},$$

où m_S et m_V sont les moyennes respectives des valeurs de dose à la surface du PTV et à l'intérieur. N_S, N_V correspondent au nombre de points de référence associés, et d_i^V, d_i^S correspondent aux valeurs de doses calculées au point i, situés à l'intérieur du PTV ou à sa surface. Les algorithmes de méthode de descente du gradient sont efficaces et donnent les solutions optimales lorsqu'il s'agit de minimiser les variances de dose pour le PTV car les fonctions objectif sont convexes dans ce cas. Milickovic *et al.* [Milickovic et al., 2002] utilisent l'algorithme de Broyden-Fletcher-Goldberg-Shanno (BFGS) pour minimiser les variances de dose.

Si l'automatisation du calcul des temps d'arrêt pour optimiser la distribution de dose est affaire courante, le placement optimal des cathéters reste empirique en s'appuyant sur des systèmes classiques de dosimétrie, développés à l'origine pour la curiethérapie LDR. Il parait donc intéressant de voir comment les principales règles de ces systèmes peuvent être adaptées pour tenir compte des possibilités offertes par la curiethérapie HDR et proposer des meilleurs schémas d'implantation.

5.3/ MODÉLISATION

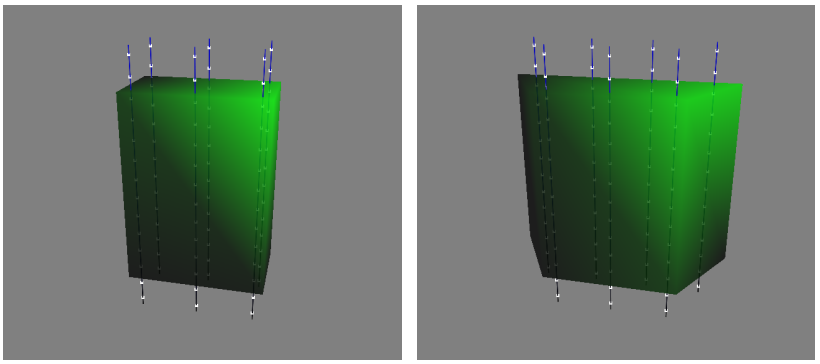
Nous nous sommes inspirés du modèle proposé par Lee *et al.* [Lee and Zaider, 2003] en curiethérapie LDR pour construire un modèle adapté à la curiethérapie HDR. Comme dans le Système de Paris [2], nous cherchons à traiter des volumes géométriques simples appelés "fantômes". Ces types de volumes sont le parallélépipède rectangle, et le prisme à base trapézoïdale isocèle, définis dans les cas respectivement d'un plan de vecteurs ou de deux plans de vecteurs disposés "en carrés", et de deux plans de vecteurs disposés "en triangles" (figure 5.2). Ces volumes géométriques simples (fantômes) sont construits très facilement par le médecin à partir des dimensions réelles de la lésion obtenues par images radiologiques ou par palpation au moment de l'examen clinique. Il lui est en effet facile de mesurer la plus grande longueur, la plus grande largeur, et la plus grande hauteur de la lésion à partir desquelles il obtient le plus petit volume (parallélépipède rectangle et prisme à base trapézoïdale) englobant parfaitement la lésion.
Notre modèle d'optimisation de la dose est le suivant :

$$\begin{cases} \min & \sum_{i \in P} \alpha_i u_i + \beta_i v_i & \\ \text{sous :} & \sum_{j \in J} d(i,j).t_j + u_i \geq \underline{D}_i & \forall i \in P \\ & \sum_{j \in J} d(i,j).t_j - v_i \leq \overline{D}_i & \forall i \in P \\ & \sum_{j \in J} d(h,j).t_j \leq 2 & \forall h \in H \\ & u_i, v_i \geq 0 & \forall i \in P \\ & t_j \geq 0 & \forall j \in J \end{cases}$$

Nous rappelons les données du problème :

- α_i, β_i : coefficients d'importance des dépassements de bornes pour le point de référence i (P ensemble des points de référence) ;
- $\underline{D}_i$, $\overline{D}_i$: bornes inférieure et supérieure de dose sur le point i ;
- $d(i,j)$: contribution de débit dose d'une source placée à la position d'arrêt j (J ensemble des points d'arrêt possibles) sur le point i.

2. Règles d'implantation de fils d'iridium 192 en curiethérapie LDR



(a) Disposition “en carrés” (b) Disposition “en triangles”

FIGURE 5.2 – Les types de volumes cible possibles, et les dispositions de vecteurs associées

— H : ensemble des points de surdosage

Les variables sont les suivantes :

— u_i, v_i : dépassements de borne inférieure et supérieure au point de référence i ;
— t_j : temps d'arrêt sur la position j .

Nous apportons quelques précisions sur le modèle en détaillant le choix des points de référence, la construction de la fonction objectif et l'écriture des contraintes.

5.3.1/ CHOIX DES POINTS DE RÉFÉRENCE

Notre choix de modélisation consiste à résoudre le problème de trouver la meilleure conformation possible de l'isodose à 100% de la dose prescrite sur le volume cible. Cela revient à chercher une distribution de dose pour laquelle :

— la dose reçue en tout point à l'intérieur et en surface du volume cible est supérieure ou égale à la dose prescrite ;
— la dose reçue à l'intérieur et en surface de chaque organe à risque est inférieure à un seuil fixé pour chaque organe ;
— la dose reçue en tout point du volume considéré et extérieur à la fois au volume cible et aux organes à risque est la plus basse possible.

Comme nous l'avons déjà évoqué, il nous est impossible de calculer la dose reçue en tout point des différents volumes. Un échantillonnage de points jugés représentatifs est donc effectué.

Pour une instance donnée, nous définissons l'ensemble P comme l'ensemble des points utilisés pour l'optimisation. A chaque point $i \in P$ est attribué un intervalle de dose, délimité par les bornes $\underline{D_i}$ et $\overline{D_i}$. Ces bornes dépendent du type de volume à l'intérieur duquel se trouve le point, et représentent les limites de dose inférieure et supérieure à ne pas dépasser, elles sont exprimées en pourcentage de la dose requise.

Par exemple, pour un point i du volume cible, la valeur $\underline{D_i}$ est égale à 1 (pour 100%). La limite supérieure de l'intervalle $\overline{D_i}$ n'ayant pas d'importance, elle sera arbitrairement fixée à $+\infty$.

Si, en revanche, le point i fait partie d'un organe à risque, la valeur $\underline{D}_i$ pour ce point est 0, et $\overline{D}_i$ représentera un pourcentage de la dose requise (par exemple 0.5 pour 50%). Pour un autre organe à risque, ce pourcentage peut être fixé à une valeur différente.

Il est important de constater que les points en surface du volume cible revêtent un intérêt particulier : si l'objectif est de s'assurer que la dose, en tout point du volume, ne dépasse pas un certain seuil, il est aisé de constater que la dose dans les points situés à l'intérieur du volume recevront généralement une dose supérieure à celle reçue par les points situés en surface du volume cible. En effet, étant donné le fait que la dose est transmise à partir de positions situées à l'intérieur du volume, et que la dose reçue en un point diminue lorsque le point s'éloigne des positions de source, la dose reçue est donc plus faible sur les points les plus éloignés des positions, c'est-à-dire en surface du volume. Ainsi, optimiser la distribution de dose dans le volume cible revient à optimiser la dose reçue sur la surface de ce volume. Il est donc dans notre intérêt de chercher à optimiser la dose sur un échantillonnage de points de dose dont une grande partie est située en surface du volume cible.

Enfin, pour un point situé dans le tissu normal, sachant qu'il n'y a pas de limite de dose explicitement formulée, ni inférieure, ni supérieure, une possibilité est de fixer un intervalle de dose non contraint (dont les bornes sont 0 et $+\infty$) pour ce point. Cela peut s'avérer judicieux lors de la présence d'organes à risque : le problème à résoudre sera de définir un compromis entre le traitement convenable de la lésion et le risque lié à la surexposition des structures à risque, sans tenir aucunement compte du dosage dans le tissu normal.

Il est également possible de fixer une borne supérieure de dose à un pourcentage relativement faible, voire proche de zéro, en fixant un faible facteur d'importance au dépassement des bornes de dose sur le point considéré, comme nous allons le décrire au paragraphe suivant.

5.3.2/ FONCTION OBJECTIF

L'évaluation de la qualité d'un traitement varie en fonction des choix effectués par le médecin. Il n'existe donc pas une manière unique d'évaluer la qualité finale du traitement.

Dans notre modèle, la fonction utilisée pour cette évaluation est une somme pondérée de pénalités obtenues pour des dépassements de dose. La pénalité de dépassement de dose pour un point de calcul considéré est l'écart entre la dose reçue et la valeur borne de l'intervalle acceptable de dose pour ce point. Les pénalités de dépassement de borne sont pondérées par des coefficients constants, choisis en fonction du type de structure contenant le point. Pour résoudre le problème, la fonction objectif choisie est la suivante :

$$\sum_{i \in P} \alpha_i u_i + \beta_i v_i$$

Les valeurs u_i et v_i sont des variables qu'il faut minimiser afin de minimiser l'amplitude totale des dépassements de dose. Le principe de l'optimisation consiste donc à trouver les valeurs pour les temps d'arrêt qui produisent des écarts de dose les plus petits possibles.

Les valeurs α_i et β_i sont des paramètres définis avant l'optimisation. Ce sont des facteurs d'importance associés aux dépassements de bornes pour les différentes structures à l'intérieur desquelles se trouvent les points de calcul de dose. Ces facteurs sont choisis

par le médecin, en fonction de ce qu'il cherche à privilégier : dans certains cas, le respect des contraintes de dose supérieure dans les organes à risque peut être plus important que le respect de la dose inférieure dans les points du volume cible. Dans d'autres cas, ce peut être l'inverse. Les valeurs choisies pour les α_i et β_i des différents points d'une même structure sont égales. Il est admis que l'importance des dépassements de bornes dans le modèle puisse être considérée comme proportionnelle à l'importance donnée aux dépassements réels de dose lors du traitement.

5.3.3/ CONTRAINTES DE DOSE

Les contraintes de dose en chaque point de calcul i de dose de l'ensemble P garantissent soit que la dose est bien respectée dans l'intervalle de dose défini pour i, soit en cas de dépassement de la borne, que la valeur de la variable de pénalité associée soit égale au dépassement proprement dit.

La contrainte de borne inférieure de dose pour $i \in P$ est définie si la borne inférieure de dose pour i est non nulle. Elle s'écrit sous cette forme :

$$\sum_{j \in J} d(i,j).t_j + u_i \geq \underline{D}_i$$

Cette contrainte exprime le fait que la somme de la dose reçue en i avec la pénalité de borne inférieure doit être supérieure ou égale à la borne inférieure de dose en i. Dans le cas le plus favorable, la dose reçue en i est supérieure ou égale à $\underline{D}_i$, et donc u_i prendra une valeur nulle. Dans le cas contraire, u_i prendra la valeur la plus petite possible telle que la contrainte soit respectée, on aura donc : $u_i = \underline{D}_i - \sum_{j \in J} d(i,j).t_j$. De même que précédemment, l'optimisation fournira une valeur nulle pour v_i si la contrainte de dose est respectée, et une valeur positive ($v_i = \sum_{j \in J} d(i,j).t_j - \overline{D}_i$) si la dose reçue en i est supérieure à $\overline{D}_i$.

5.3.4/ CONTRAINTES DE SURDOSAGE

Notre modèle d'optimisation inclut également des contraintes liées au respect d'une règle du Système de Paris, indiquant que le diamètre des manchons de surdosage (isodose à 200% de la dose requise) obtenus ne doit pas dépasser 1 cm.

Les manchons de surdosage sont modélisés par un ensemble de points de référence spécifiques, que nous appelons points de surdosage (ensemble H). Ces points sont placés de telle sorte qu'un surdosage sur l'un de ces points implique automatiquement la détection d'un diamètre de manchon supérieur à 1 cm (pour plus de détails, voir la thèse [Galea, 2006]).

5.3.5/ OPTIMISATION DU PLACEMENT DES CATHÉTERS

Le modèle d'optimisation décrit précédemment nous permet d'effectuer la simulation d'implantations de vecteurs tout en respectant les principes du Système de Paris. Le seul

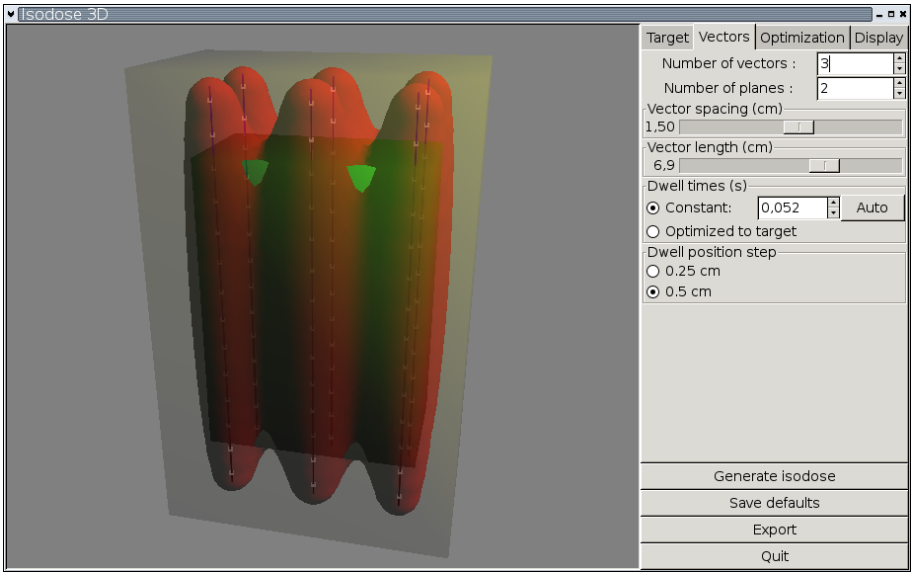


FIGURE 5.3 – Vue générale de l'interface du logiciel *Isodose 3D*

degré de liberté offert pour le placement des cathéters est l'écartement δ des vecteurs, ainsi que le choix de la disposition, en carrés ou en triangles. Le nombre de vecteurs est déterminé par la disposition et l'écartement des vecteurs (principe d'équidistance des vecteurs dans le système de Paris), ainsi que des dimensions du volume cible. La longueur des vecteurs est déterminée par les positions d'arrêt de la source pour laquelle il existe des temps d'arrêts non nuls.
Nous avons développé un logiciel graphique Isodose 3D qui permet la visualisation de la couverture d'un volume cible par la dose requise, selon différents choix et paramétrages (nombre de points de référence, valeurs des pénalités, écartement des vecteurs) du calcul des temps d'arrêt. La figure 5.3 présente une vue générale de l'interface du logiciel.

5.4/ CONCLUSION

En curiethérapie HDR, le problème d'optimisation de la dose consiste à rechercher les positions et les temps d'arrêt d'une source radioactive. Nous avons proposé un modèle original incluant des garanties quant au respect des principes fondamentaux du système de Paris pour le placement des cathéters.
Ce modèle d'optimisation nous a permis d'effectuer la simulation d'implantations respectant les principes du système de Paris afin d'en tirer des informations sur la distribution de dose et de rechercher la meilleure implantation de vecteurs possible. Nous avons mis au point un logiciel Isodose 3D qui permet de visualiser de façon synthétique la couverture d'un volume "fantôme" par la dose requise selon différents choix et paramétrages du calcul des temps d'arrêt.
La poursuite de ce travail consisterait à procéder à des simulations sur des cas réels impliquant des volumes aux formes plus complexes que les "fantômes". Et un axe de recherche pourrait consister à l'élaboration de nouvelles règles de dispositions des vecteurs, violant les principes d'équidistance et d'alignement des vecteurs dans le

système de Paris, principes qui se justifiaient à l'époque par la difficulté d'établir une méthodologie simple de calcul de la dose mais qui sont devenus désuets avec la puissance de calcul des ordinateurs.
Nous n'avons pas poursuivi de travaux sur cette thématique après 2006. Nous remarquons que depuis cette date, il y a eu d'autres propositions de modèles d'optimisation des temps d'arrêt des sources. Le modèle proposé par [Alterovitz et al., 2006] est semblable au notre. L'étude compare les solutions obtenues par recuit simulé et celles obtenues par résolution du programme linéaire et montre qu'elles sont (statistiquement) équivalentes du point de vue de critères cliniques spécifiques. Une autre étude [Holm et al., 2013] aboutit à la construction d'un modèle (PL) incluant directement des indices cliniques issus des histogrammes dose-volumes et montre que ce dernier se rapproche du modèle précédemment cité à condition de choisir convenablement les valeurs des pénalités associées aux dépassements de dose.
Concernant le placement des cathéters, des formulations semblables sous forme de (MIP) sont données dans [Karabis et al., 2009] et dans [Holm et al., 2016]. Le modèle repose toujours sur la pénalisation des écarts de dose. Il inclut le placement des cathéters en introduisant un ensemble de positions possibles pour les cathéters et en associant une variable binaire à chaque cathéter qui indique si celui-ci est utilisé ou non. Le problème est résolu soit avec un solveur de PL, soit avec des métaheuristiques [Holm et al., 2016] (Méthode Tabou, algorithme génétique, Méthode de recherche à voisinage variable). Notons que ces études se restreignent à utiliser des cathéters parallèles entre eux, mais pas forcément équidistants.

6

LE PROBLÈME DE MINIMISATION DU NOMBRE DE REGISTRES

6.1/ PROBLÉMATIQUE

Nous nous intéressons au problème d'optimisation du besoin en stockage dans les graphes de tâches périodiques. En pratique, notre problème tend à minimiser le besoin en registres dans les programmes, où les instructions d'une boucle sont représentées par un graphe de dépendances de données cyclique (GDD).

L'introduction massive de processeurs ILP (Instructions Level Parallelism) ces trois dernières décennies nous conduit à repenser la manière d'optimiser le nombre de registres de stockage dans les codes assembleur avant de commencer le processus d'ordonnancement des instructions sous contraintes de ressources. Dans ces processeurs, les instructions sont exécutées en parallèle grâce à l'existence de plusieurs petites unités de calcul (additionneurs, multiplicateurs, unités de chargement/stockage, etc.). L'exploitation de ce parallélisme à grain fin (au niveau du code assembleur) demande de revenir sur le problème classique de l'allocation de registre initialement conçu pour les processeurs séquentiels. De nos jours, l'allocation de registre n'a pas seulement pour effet de réduire au minimum les besoins de stockage, mais elle doit aussi prendre en compte le parallélisme et le temps d'exécution total du programme.

Ici, nous ne tenons pas compte des contraintes de ressources en dehors des exigences de stockage. Notre objectif est double :

- soit minimiser le nombre de registres utilisés pour une valeur de période fixe dans un problème d'ordonnancement de tâches cycliques,
- soit minimiser la valeur de la période pour un nombre fixe et borné de registres.

Notons que ce problème d'optimisation de stockage est suffisamment abstrait pour s'appliquer dans d'autres domaines de planification qui traitent conjointement du stockage et de la minimisation du temps total d'exécution pour des tâches répétitives (fabrication, transport, réseau, etc.).

6.2/ ETAT DE L'ART

Dans cette étude, nous supposons une exécution parallèle des instructions sans aucun modèle de ressources - le parallélisme étant borné par les contraintes de stockage

uniquement. Notre but est d'analyser le compromis entre le besoin en registres et le parallélisme dans un problème d'ordonnancement périodique de tâches.

Les techniques existantes d'allocation périodique de registres appliquent généralement un ordonnancement périodique d'instructions sous contraintes de ressources, tout en étant sensible aux contraintes de registres. Plusieurs travaux proposent des algorithmes d'ordonnancement d'instructions d'une boucle (sous contraintes de ressources et de période) tel que le code résultant ne crée pas plus de R variables simultanément en vie. Or, en pratique, de tels algorithmes n'engendrent pas toujours des codes rapides, car les contraintes de registres sont plus critiques que les contraintes de ressources : ceci car les accès mémoire coûtent plus de 100 unités de temps (cycles processeur) alors que les latences des instructions sont inférieures à 10 cycles. Dans cette étude, nous garantissons les contraintes des registres avant la phase d'ordonnancement d'instructions sous contraintes de ressources : nous analysons et modifions le GDD afin de garantir que tout ordonnancement d'instructions sous contraintes de ressources ne crée pas plus de R variables simultanément en vie. Notre modification du GDD essaie de ne pas altérer l'extraction du parallélisme si possible en prenant comme métrique la valeur du circuit critique du graphe.

6.3/ MODÉLISATION

6.3.1/ MODÈLE DE TÂCHES

Notre modèle est similaire à celui proposé par [Hanen and Munier, 1995]. Nous considérons un ensemble de l tâches génériques (instructions à l'intérieur d'une boucle de programme) $T_0, \ldots, T_{l-1}$. Chaque tâche T_i est exécutée n fois, où n est le nombre d'itérations de boucle. n est entier fini non borné. Chaque tâche T_i a n instances (une par boucle). La $k^{\text{ième}}$ occurrence de la tâche T_i est notée $T\langle i,k\rangle$, ce qui correspond à l'exécution de la tâche i à la $k^{\text{ième}}$ itération de la boucle, avec $0 \leq k < n$.

Les tâches (instructions) peuvent être exécutées en parallèle. Chaque tâche peut produire un résultat qui est lu (c'est-à-dire consommé/utilisé) par d'autres tâches. Les boucles considérées contiennent des dépendances entre les données qui peuvent être modélisées par un graphe $G(V,E)$ où :

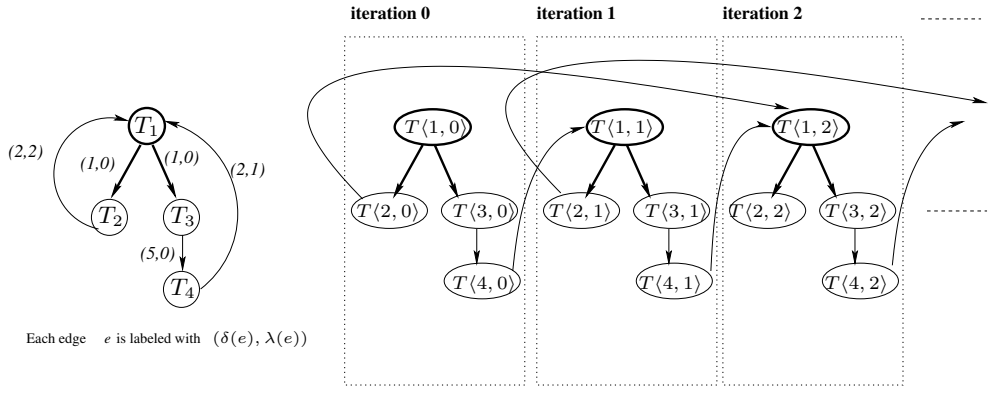
- V est l'ensemble des tâches génériques dans le corps de boucle, $V = \{T_0, \ldots, T_{l-1}\}$.
- E est l'ensemble des arcs représentant les contraintes de précédence (dépendance de flux ou autres contraintes sérialisées). Chaque arc $e = (T_i, T_j) \in E$ a une latence $\delta(e) \in \mathbb{N}$ exprimée en nombre de cycles d'horloge du processeur et une distance $\lambda(e) \in \mathbb{N}$ exprimée en nombre d'itérations de boucle. La distance $\lambda(e)$ pour l'arc $e = (T_i, T_j)$ signifie qu'il y a une dépendance entre la tâche $T\langle i,k\rangle$ et $T\langle j, k+\lambda(e)\rangle$ pour $k = 0, \ldots, n-1-\lambda(e)$.

Concernant les contraintes de stockage, on doit faire la distinction entre les tâches et les contraintes de précédence selon si elles sont liées ou non à des données stockées dans des registres :

- $V^R \subseteq V$ est l'ensemble des tâches produisant des données stockées dans des registres.
- $E^R \subseteq E$ est l'ensemble des arcs de dépendances de flux de données entre registres. Un arc $e = (T_i, T_j) \in E^R$ signifie que la tâche $T\langle i,k\rangle$ produit un résultat

stocké dans un registre puis lu (consommé) par $T\langle j, k + \lambda(e)\rangle$. L'ensemble des consommateurs (lecteurs) de la tâche générique T_i est alors défini par :
$Cons(T_i) = \{T_j \in V \mid e = (T_i, T_j) \in E^R\}$

Notons que le graphe de dépendances de tâches (DDG) est en fait un graphe multiple ; en effet il peut y avoir plus d'un arc entre deux tâches. Figure 6.1 est un exemple de graphe de dépendance (DDG) où les sommets du graphe en gras correspondent aux tâches de l'ensemble V^R produisant des données stockées dans des registres. Les arcs en gras représentent les dépendances de flux (chaque sommet final de cet arc lit/consomme une donnée produite et stockée dans un registre par le sommet initial). Les autres tâches (sommets non gras) correspondent à des instructions qui n'utilisent pas l'écriture de données dans un registre (la donnée est stockée en mémoire vive ou simplement la tâche ne produit aucune donnée). Les autres arcs (non gras) correspondent à des contraintes de précédence ou d'échanges de données différentes des dépendances de flux de données précisées précédemment. A chaque arc e dans le DDG est associé le couple $(\delta(e), \lambda(e))$.



(a) Example of a DDG with Generic Tasks (b) Loop Iterations and Instructions/Tasks Instances

FIGURE 6.1 – Exemple de graphe de dépendance avec des tâches répétitives

Dans notre modèle générique de processeur, nous supposons que l'écriture et/ou la lecture dans les registres peuvent être retardées par rapport à la date de début d'exécution de la tâche. On note $\sigma(T\langle i, k\rangle) \in \mathbb{N}$ la date d'exécution de la tâche $T\langle i, k\rangle$. Nous définissons deux fonctions de retard δ_r (en lecture) et δ_w (en écriture) :

$$
\begin{aligned}
\delta_w : \quad & V^R \to \mathbb{N} \\
& T_i \mapsto \delta_w(T_i) |\ 0 \leq \delta_w(T_i) \\
& \text{le temps d'écriture de la donnée produite par } T\langle i, k\rangle \\
& \text{est } \sigma(T\langle i, k\rangle) + \delta_w(T_i) \\
\delta_r : \quad & V \to \mathbb{N} \\
& T_i \mapsto \delta_r(T_i) |\ 0 \leq \delta_r(T_i) \\
& \text{le temps de lecture de la donnée consommée par } T\langle i, k\rangle \\
& \text{est } \sigma(T\langle i, k\rangle) + \delta_r(T_i)
\end{aligned}
$$

Ces deux fonctions de retard dépendent du type de processeur et permettent de modéliser la plupart des architectures des processeurs actuels (VLIW, EPIC/IA64 et processeurs super-scalaires).

Dans notre modèle de tâches, le nombre d'occurrences est indéterminé et non borné. Nous recherchons une planification périodique de ces tâches pour générer un code final compact. On associe une période unique p entière (à déterminer) pour un ordonnancement périodique. La période p est entière et commune à toutes les tâches, ce qui simplifie la génération de code de la boucle finale. D'autres modèles d'ordonnancement multi-périodiques peuvent être employés (avec des périodes multiples ou des périodes rationnelles), mais au détriment de la taille du code généré. Ici, notre ordonnancement périodique avec une unique période p associe à chaque tâche générique T_i une date d'exécution pour uniquement la première occurrence de la tâche $T\langle i,0\rangle$ que nous notons : $\sigma_i = \sigma(T\langle i,0\rangle)$. La date d'exécution des autres occurrences $T\langle i,k\rangle$ est égale à $\sigma(T\langle i,k\rangle) = \sigma_i + k \times p$. L'ordonnancement périodique σ doit satisfaire aux contraintes classiques :

$$\forall e = (T_i, T_j) \in E : \sigma_i + \delta(e) \leq \sigma_j + \lambda(e) \times p \tag{6.1}$$

Classiquement, en additionnant ces inégalités sur un circuit C du DDG G, on trouve que la période p doit être supérieure ou égale à : $\max_{\text{circuit } C} \left\lceil \frac{\sum_{e\in C} \delta(e)}{\sum_{e\in C} \lambda(e)} \right\rceil$, Dans la suite, nous appelons cette quantité Période Minimale d'Exécution MEP. Calculer MEP pour un graphe cyclique est un problème polynomial [Hanen and Munier, 1995, Lawler, 1972] : comme le DDG est construit par une méthode d'analyse des flux de données par le compilateur, et comme le DDG représente les dépendances de données entre les instructions d'un programme, la valeur de MEP existe toujours et a une valeur positive.
Le problème usuel d'ordonnancement périodique de tâches consiste à rechercher un ordonnancement compatible avec un ensemble de contraintes (ressources, stockage limité,etc). Ici, nous nous limitons aux contraintes de stockage.

6.3.2/ GRAPHES DE RÉUTILISATION

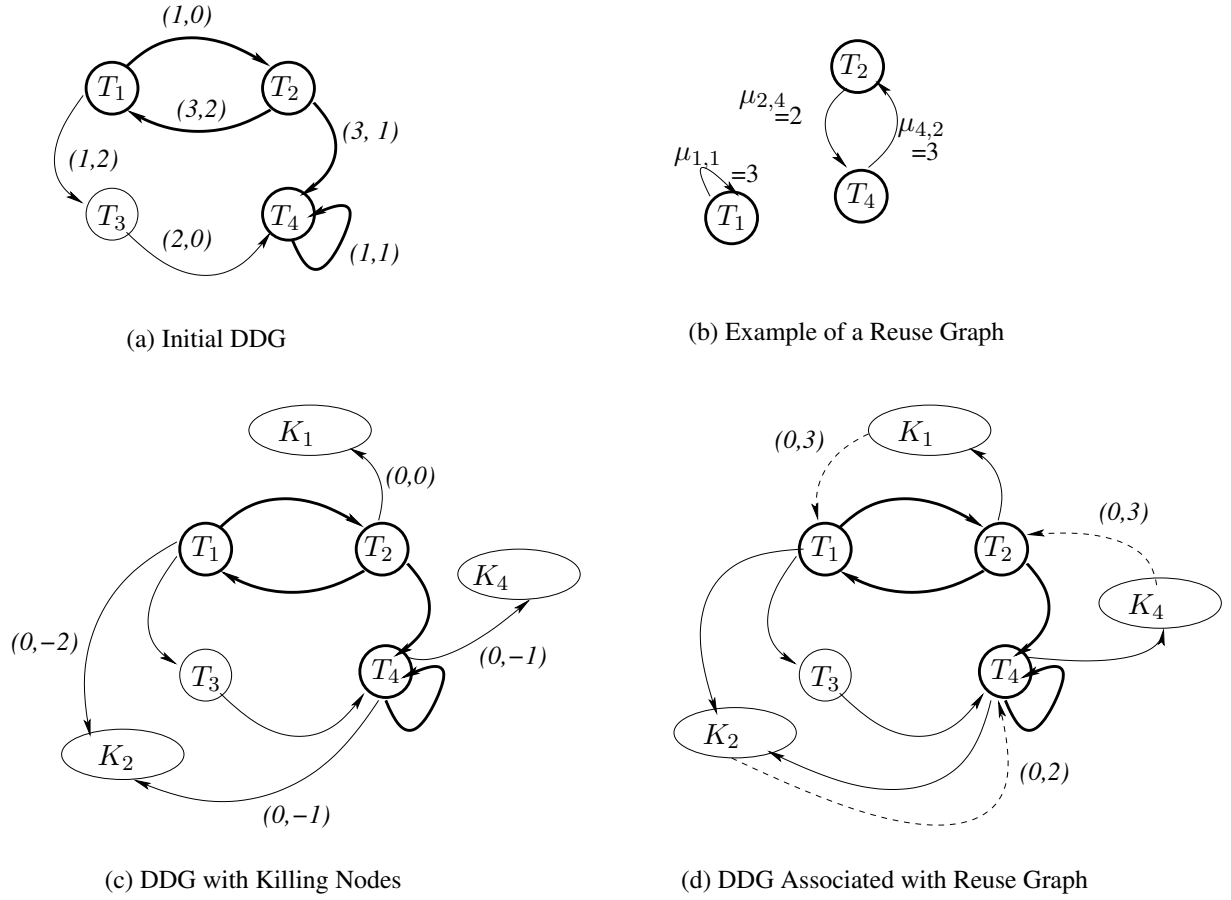
Touati et Eisenbeis [Touati and Eisenbeis, 2004, Touati, 2002] ont établi un cadre théorique, basé sur l'usage de graphes, qui fournit des résultats fondamentaux sur l'allocation de registres. Ils ont introduit notamment la notion de graphe de réutilisation que nous allons présenter dans la suite et qui nous permettra d'écrire le modèle de programmation linéaire en nombre entier pour notre problème d'ordonnancement.

La manière la plus simple de présenter la notion de graphe de réutilisation est de l'appliquer à un exemple. Pour des définitions et des résultats plus formels, il faut se référer aux travaux de Touati et Eisenbeis [Touati and Eisenbeis, 2004, Touati, 2002]. La figure 6.2(a) montre un DDG initial. Les tâches en gras sont celles qui produisent des résultats stockés dans des registres. Les dépendances de flux de données sont représentées par les arcs en gras. Par exemple, $Cons(T_2) = \{T_1, T_4\}$. Un couple de valeurs $(\delta(e), \lambda(e))$ est associé à chaque arc e. Dans cet exemple très simple, on suppose que le délai d'accessibilité aux registres est nul ($\delta_w = \delta_r = 0$). L'allocation de registre, qui détermine les tâches qui partagent le même registre, est modélisée à travers le graphe de réutilisation. Un exemple de graphe de réutilisation est donné à la figure 6.2(b). Les noeuds sont connectés entre eux par des arcs de réutilisation. Chaque arc (T_i, T_j) de réutilisation est caractérisé par une distance entière $\mu_{i,j}$. L'existence d'un arc de réutilisation (T_i, T_j) de distance $\mu_{i,j}$ signifie que les deux occurrences de tâches $T\langle i,k\rangle$ and $T\langle j, k+\mu_{i,j}\rangle$ partagent le même registre de destination pour le stockage de leurs résultats respectifs. Ainsi la connaissance du graphe de réutilisation nous permet

de définir complètement l'allocation de registre pour un DDG donné. Rappelons que pour être valide, le graphe de réutilisation doit satisfaire à deux contraintes principales [Touati and Eisenbeis, 2004, Touati, 2002] :

— il doit y avoir une bijection entre les noeuds ; autrement dit, le graphe doit être composé de circuits disjoints élémentaires,

— Le graphe DDG "associé" doit être ordonnançable, c'est à dire qu'un ordonnancement cyclique de période p doit être possible.



(a) Initial DDG

(b) Example of a Reuse Graph

(c) DDG with Killing Nodes

(d) DDG Associated with Reuse Graph

FIGURE 6.2 – Graphe de réutilisation et DDG associé

Décrivons à présent le lien entre le DDG "associé" et le graphe de réutilisation. Pour un graphe de réutilisation fixé, le partage de registres entre plusieurs tâches induit de nouvelles contraintes entre les tâches. Comme un arc (T_i, T_j) dans le graphe de réutilisation G^{reuse} traduit un partage de registre entre $T\langle i,k\rangle$ and $T\langle j,k+\mu_{i,j}\rangle$, il faut s'assurer que $T\langle j,k+\mu_{i,j}\rangle$ puisse écrire sur le même registre, une fois que le résultat de $T\langle i,k\rangle$ est "mort". On dit que le résultat d'une tâche est "mort" une fois que tous les tâches consommatrices de ce résultat l'ont lu et qu'il n'est par conséquent plus nécessaire de le maintenir dans le registre. Le dernier lecteur de la donnée résultat est appelé le "tueur". Si le DDG est déjà ordonnancé, il est aisé de calculer les dates de "mort" de chaque donnée (la date de "mort" est la date à laquelle le résultat est "mort", lu par tous ses consommateurs). En revanche, si le DDG n'est pas encore ordonnancé, comme dans notre cas, les dates de "mort" ne sont pas encore connues et on doit être en mesure de maintenir la validité de l'allocation de stockage pour tous les ordonnancements

périodiques possibles. Cet objectif passe par la création en deux étapes d'un DDG "associé" au graphe de réutilisation :

1. Premièrement, on ajoute au DDG initial des sommets fictifs appelés noeuds "tueurs" [de Dinechin, 1996] : le noeud "tueur" K_i pour une tâche $T_i \in V^R$ représente le dernier consommateur virtuel de T_i. Le noeud "tueur" K_i doit être toujours ordonné après tous les consommateurs de la tâche T_i. Pour traduire cette relation d'ordre dans l'ordonnancement, on ajoute l'ensemble des arcs $\{(T_j, K_i) | T_j \in Cons(T_i)\}$. La figure 6.2(c) montre le DDG après ajout des noeuds "tueurs". Pour chaque arc ajouté $e = (T_j, K_i)$, on fixe sa latence à $\delta(e) = \delta_r(T_j)$ et sa distance à $-\lambda$, où λ est la distance de l'arc $(T_i, T_j) \in E^R$. Cette distance négative est une convention [Touati and Eisenbeis, 2004, Touati, 2002] qui simplifie les formules mathématiques et n'influence pas les résultats fondamentaux obtenus sur le graphe de réutilisation.

2. Deuxièmement, une fois les noeuds "tueurs" insérés, on ajoute de nouveaux arcs qui résultent de l'allocation de registres établie dans le graphe de réutilisation. Pour chaque arc (T_i, T_j) dans le graphe de réutilisation G^{reuse}, on ajoute un arc $e = (K_i, T_j)$ dans le DDG "associé". L'arc ajouté a une latence égale à $\delta(e) = -\delta_w(T_j)$ et une distance égale à $\lambda(e) = \mu_{i,j}$. La figure 6.2(d) montre le DDG "associé" obtenu à partir du graphe de réutilisation de la figure 6.2(b), où les arcs de réutilisation ajoutés apparaissent en pointillé. Un lecteur averti peut noter que le circuit critique du DDG dans les figures 6.2(a) and (d) sont identiques et que $MEP = \frac{4}{2} = 2$ (un circuit critique est (T_1, T_2, T_1)).

Il a été montré dans [Touati and Eisenbeis, 2004, Touati, 2002] que si un graphe de réutilisation est valide, alors il correspond à une allocation périodique de registres utilisant exactement $\sum \mu_{i,j}$ registres. Le problème central est donc de produire un graphe de réutilisation valide avec un nombre de registres $\sum \mu_{i,j}$ minimal et une période $p = MEP$ minimale. Ce problème est un problème NP-complet [Touati, 2002]. Dans certains cas, le nombre R de registres disponibles est déjà fixé par les fabricants de processeurs. Le problème revient à chercher un graphe de réutilisation valide de telle sorte que $\sum \mu_{i,j} \leq R$ avec une période minimale. Notre méthode de résolution devient itérative : on résout le problème d'ordonnancement en commençant par une période $p = MEP$ et on l'incrémente successivement jusqu'à obtenir $\sum \mu_{i,j} \leq R$. Si la valeur de la période trouvée dépasse une certaine limite L, on dit que le problème n'admet pas de solution. Le recours possible au "spilling" dans ce cas pour stocker les variables en mémoire externe, ne fait pas l'objet de cette étude. Une recherche binaire de période p (entre MEP et L), plus efficace qu'une recherche incrémentale, peut être utilisée à condition de résoudre le problème d'ordonnancement de manière exacte pour chaque valeur de période p.

6.3.3/ PROGRAMME LINÉAIRE

Dans cette partie, nous présentons la formulation, sous forme de programme linéaire, du problème d'ordonnancement périodique, avec une période fixée $p \in \mathbb{N}$, et utilisant un nombre minimal de registres. Nous décrivons l'ensemble des variables et des contraintes du problème avant de proposer le programme linéaire dans sa globalité.

6.3.3.1/ VARIABLES

— Les variables $\sigma_i \in \mathbb{N}$ de date de début de tâche pour chaque tâche $T_i \in V$. Pour chaque tâche fictive $K_i \in V$, on associe les variables σ_{K_i} de date de début. Comme il s'agit d'un ordonnancement cyclique, on s'intéresse uniquement aux dates d'exécution des premières occurrences $\sigma_i = \sigma(T\langle i, 0\rangle)$ et les dates des autres occurrences $T\langle i, k\rangle$ sont obtenues par périodicité $\sigma(T\langle i, k\rangle) = \sigma_i + k \times p$;
— Les variables $\theta_{i,j}$ pour chaque paire de tâches $(T_i, T_j) \in V^R \times V^R$. $\theta_{i,j}$ est égal à 1 si et seulement si l'arc (T_i, T_j) est un arc de réutilisation, 0 sinon ;
— Les variables de distance de réutilisation $\mu_{i,j} \in \mathbb{N}$ pour chaque paire de tâches $(T_i, T_j) \in V^R \times V^R$.

6.3.3.2/ CONTRAINTES

— **Dépendances de données**
L'ordonnancement doit respecter les contraintes de précédence défini par le DDG

$$\forall e = (T_i, T_j) \in E : \sigma_j - \sigma_i \geq \delta(e) - p \times \lambda(e) \tag{6.2}$$

— **Dépendances de flux de données**
Chaque dépendance de flux $e = (T_i, T_j) \in E^R$ signifie que l'occurrence $T\langle j, k + \lambda(e)\rangle$ lit la donnée produite par $T\langle i, k\rangle$ à la date $\sigma_j + \delta_r(T_j) + (\lambda(e) + k) \times p$. On doit donc ordonnancer le noeud "tueur" K_i de la tâche T_i après tous les consommateurs de la tâche T_i. $\forall T_i \in V^R,\ \forall T_j \in Cons(T_i) | e = (T_i, T_j) \in E^R$:

$$\sigma_{K_i} \geq \sigma_j + \delta_r(T_j) + p \times \lambda(e) \tag{6.3}$$

— **Dépendances de stockage**
Il y a une dépendance de partage de registres entre T_i and T_j si (T_i, T_j) est un arc de réutilisation. $\forall (T_i, T_j) \in V^R \times V^R$:

$$\theta_{i,j} = 1 \Longrightarrow \sigma_{K_i} - \delta_w(T_j) \leq \sigma_j + p \times \mu_{i,j}$$

Cette implication peut s'écrire sous forme d'inégalités : $\forall (T_i, T_j) \in V^R \times V^R$,

$$\sigma_j - \sigma_{K_i} + p \times \mu_{i,j} + M_1 \times (1 - \theta_{i,j}) \geq -\delta_w(T_j) \tag{6.4}$$

où $M_1 \in \mathbb{N}$ une constante arbitraire suffisamment grande.
S'il n'y a pas de réutilisation de registres entre T_i et T_j, alors $\theta_{i,j} = 0$ et la valeur de la distance de réutilisation $\mu_{i,j}$ correspondante doit être nulle. Cette contrainte peut se formuler aussi sous forme d'inégalité :

$$\forall (T_i, T_j) \in V^R \times V^R :\ \mu_{i,j} \leq M_2 \times \theta_{i,j} \tag{6.5}$$

où $M_2 \in \mathbb{N}$ une constante arbitraire suffisamment grande.

— **Relation de réutilisation de registres**
La relation de réutilisation de registres est une bijection de V^R dans V^R. Un registre peut-être réutilisé par une seule tâche et une tâche réutilise un seul registre :

$$\forall T_i \in V^R : \sum_{T_j \in V^R} \theta_{i,j} = 1 \tag{6.6}$$

$$\forall T_j \in V^R : \sum_{T_i \in V^R} \theta_{i,j} = 1 \tag{6.7}$$

6.3.3.3/ FONCTION OBJECTIF

Il a été démontré dans [Touati and Eisenbeis, 2004] que le nombre de registres nécessaires, dans le cas d'un graphe de réutilisation valide, est égal à $\sum \mu_{i,j}$. Ici, on cherche donc à minimiser cette somme :

$$\begin{cases} \min \sum_{(T_i,T_j)\in V^R\times V^R} \mu_{i,j} & \\ \text{sous :} & \\ \sigma_j - \sigma_i \geq \delta(e) - p \times \lambda(e), & \forall e = (T_i, T_j) \in E \\ \sigma_{K_i} - \sigma_j \geq \delta_r(T_j) + p \times \lambda(e), & \forall e = (T_i, T_j) \in E^R \\ \sigma_j - \sigma_{K_i} + p \times \mu_{i,j} + M_1 \times (1 - \theta_{i,j}) \geq -\delta_w(T_j) & \forall e = (T_i, T_j) \in E^R \\ \mu_{i,j} \leq M_2 \times \theta_{i,j} & \forall e = (T_i, T_j) \in E^R \\ \sum_{T_j\in V^R} \theta_{i,j} = 1 & \forall T_i \in V^R \\ \sum_{T_i\in V^R} \theta_{i,j} = 1 & \forall T_j \in V^R \\ \sigma_i, \mu_{i,j} \in \mathbb{N} & \\ \theta_{i,j} \in \{0, 1\} & \end{cases} \tag{6.8}$$

6.4/ CONCLUSION

Nous avons proposé une formalisation exacte par programmation linéaire du problème générique d'ordonnancement périodique de tâches sous contraintes de stockage (problème NP-complet). Le problème d'ordonnancement périodique de tâches (instructions) cycliques avec minimisation du nombre de registres, dans sa forme la plus générale, peut se formuler sous forme d'un programme linéaire en nombre entiers où les variables de décision sont les dates de début de chacune des tâches, des variables binaires indiquant une réutilisation ou non de registres entre deux tâches (non nécessairement distinctes), et les distances de réutilisation.

Nous avons présenté un nouvel environnement formel s'appuyant sur la théorie des graphes qui permet l'optimisation (hors contraintes de ressources) du besoin de stockage dans le cas d'un ordonnancement cyclique de tâches. Notre problème théorique est dérivé d'une application concrète d'optimisation de registres dans les processeurs modernes à parallélisme d'instructions. Notre formalisme repose sur un modèle par graphes de dépendances où les contraintes de stockage sont décrites grâce à l'insertion de nouveaux arcs (appelés *arcs de réutilisation*) étiquetés avec des *distances* de réutilisation. Ce formalisme nous permet de fixer précisément la valeur exacte du besoin en stockage (nombre de valeurs simultanément en vie) pour quelque soit l'ordonnancement. Le calcul des arcs et des distances de réutilisation est défini par la période minimale d'ordonnancement souhaitée (resp. le débit maximal d'exécution) ainsi qu'avec le besoin en stockage.

Cette approche permet, pour la première fois dans le domaine d'optimisation des programmes bas niveau, de pouvoir contrôler exactement et précisément la taille de code induite par l'ordonnancement périodique d'instructions et la coupler au besoin de stockage et à la période de l'ordonnancement.

7

LE PROBLÈME DE MAXIMISATION DE LA DURÉE DE VIE D'UN RÉSEAU DE CAPTEURS

7.1/ PROBLÉMATIQUE

Le besoin d'observer/contrôler divers phénomènes tels que la température, la pression ou encore la détection d'intrusions est essentiel pour de nombreuses applications industrielles, militaires, scientifiques, et même grand public. Cette tâche est assurée essentiellement par les réseaux de capteurs dont la fonction est l'acquisition de l'information sur les phénomènes observés. Les réseaux de capteurs sans fil sont formés d'un ensemble de petits appareils électroniques, autonomes, équipés de capteurs et capable de communiquer entre eux sans fil. Il permettent ainsi de contrôler une région ou un phénomène d'intérêt, de fournir des informations utiles par la combinaison des mesures prises par les différents capteurs et de les communiquer ensuite via le support sans fil vers une station de base. Aujourd'hui des capteurs minuscules et bon marché peuvent être éparpillés sur des routes, des structures, des murs ou des machines, et sont capables de détecter une variété de phénomènes physiques. De nombreux domaines en font usage, principalement dans l'aéronautique, l'industrie pétrolière, l'automobile, l'environnement et la santé.

Les avancées technologiques permettent de déployer un réseau constitué d'un grand nombre de capteurs collaboratifs afin de surveiller une zone d'intérêt plus large. Cependant, son exploitation reste difficile et pose beaucoup de difficultés aussi bien d'un point de vue algorithmique que pratique (localisation, déploiement, collecte/fusion de données, couverture).

En particulier, la gestion de la consommation d'énergie est un élément clé pour l'optimisation du fonctionnement du réseau. Nous nous concentrerons ici sur l'optimisation de la durée de vie du réseau tout en maintenant la couverture de la zone d'intérêt.

Un nœud capteur contient :

- une unité de captage chargée d'enregistrer des grandeurs physiques, (chaleur, humidité, vibrations) et de les transformer en grandeurs numériques,
- une unité de traitement informatique et de stockage de données,
- un module de transmission sans fil,

— une ressource énergétique.

Selon le domaine d'application, il peut aussi contenir des modules supplémentaires tels qu'un système de localisation (GPS), ou bien un système générateur d'énergie (cellule solaire). L'énergie est la ressource la plus précieuse dans un réseau de capteurs, puisqu'elle influe directement sur la durée de vie des micro-capteurs et du réseau en entier. La plupart des noeuds capteurs utilisent des piles comme source d'énergie. A titre indicatif, ce sera souvent une pile AA normale d'environ 2,2-2,5 A*h* fonctionnant à 1,5 V. Toutefois, il est généralement impossible de recharger ces piles en raison de l'emplacement des nœuds, mais le plus souvent pour la simple raison que cette opération est pratiquement ou économiquement infaisable.

7.2/ ÉTAT DE L'ART

Les problèmes de couverture principalement traités dans la littérature peuvent être classés en trois grandes catégories [Li and Vasilakos, 2013] selon leur objectif de couverture. **La couverture de zone** [Misra et al., 2011] signifie que chaque point à l'intérieur de la zone doit être couvert. **La couverture de cibles** [Yang and Chin, 2014a] concerne la couverture d'un ensemble fini de points, appelés cibles. Et enfin, **la couverture de frontière** [He et al., 2014, Kim and Cobb, 2013] se concentre sur les problématiques de détection d'intrusion dans une zone d'intérêt. Sous certaines conditions, nous verrons qu'il est possible de transformer un problème de couverture de zone en un problème de couverture de cibles. Nous avons travaillé, à la fois sur des problèmes de couverture de zones et de cibles. Nous n'avons pas traité des problèmes d'intrusion. La suite du texte présente les principaux travaux de recherche traitant du problème de couverture, indistinctement, de zones ou de cibles.

Chaque capteur a un rayon de couverture R_s, ce qui signifie qu'il est en mesure de couvrir un disque de rayon R_s centré sur lui-même. Un nœud capteur peut communiquer avec son voisin si celui-ci se trouve dans sa zone de communication, c'est-à-dire à une distance euclidienne inférieure ou égale à R_c (rayon de communication). Dans notre étude, les rayons de communication R_c et de couverture R_s satisfont à la condition suivante : $R_c \geq 2 \cdot R_s$. Cette hypothèse forte nous permet d'affirmer d'après [Zhang and Hou, 2005], qu'une couverture complète d'une région convexe implique la connectivité des nœuds actifs. Nous pouvons ainsi concentrer tous nos efforts sur le problème de couverture.

L'approche principale pour diminuer la consommation d'énergie des capteurs et ainsi augmenter la durée de vie du réseau consiste à tirer partie du surnombre de capteurs déployés pour ne pas les activer tous à la fois. Les capteurs sont alors regroupés en ensembles dit "ensembles couvrants" [Wang, 2011], où chaque ensemble couvrant couvre entièrement la zone d'intérêt. Les ensembles couvrants disjoints ou non disjoints sont alors activés successivement. L'activité des nœuds du réseau peut être planifiée à l'avance pour toute la durée de vie du réseau ou mise à jour périodiquement, et l'ensemble couvrant activé est décidé au début de chaque période [Ling and Znati, 2009]. En fait, de nombreux auteurs [Vu, 2009, Yan et al., 2008, Padmavathy and Chitra, 2010] ont proposé des algorithmes pour ce mode périodique. Le choix des nœuds activés à chaque période diffèrent selon les besoins. L'accent peut être mis en priorité sur la couverture totale de la zone, sur le maintien de la

connectivité [Jaggi and Abouzeid, 2006], ou encore sur la gestion de l'énergie.

Des approches centralisées ou distribuées, ou encore un mixte des deux, ont été proposées pour étendre la durée de vie du réseau [Zhou et al., 2009]. Dans les algorithmes distribués [Vu et al., 2006, Qu and Georgakopoulos, 2013, Yang and Chin, 2014b], chaque capteur décide par lui même de son propre état de veille ou d'activité en échangeant des informations avec ses voisins. L'intérêt principal d'une telle approche est de limiter les communications longue portée (vers la station de base) et de réduire les coût de communication. Malheureusement, comme chaque capteur n'a qu'une vision partielle du réseau (information sur ses voisins proches uniquement), il peut prendre une décision qui conduit à une solution globale sous-optimale. Inversement, les algorithmes centralisés [Cardei and Du, 2005, Zorbas et al., 2010, Pujari, 2011] fournissent toujours des solutions quasi-optimales, car la station de base, qui réalise les calculs pour la planification, a collecté l'ensemble des informations sur les nœuds du réseau. Les algorithmes centralisés présente l'avantage de requérir très peu de puissance de calcul de la part du capteur, mais l'inconvénient de nécessiter un nombre important de communications avec la station de base. De manière évidente, plus le réseau est grand (c'est-à-dire avec un grand nombre de capteurs), plus le coût énergétique des consommations l'est aussi. Un étude récente [Padmavathy and Chitra, 2010] montre cependant qu'il existe un seuil sur le nombre de capteurs au delà duquel il parait intéressant d'employer une méthode centralisée plutôt qu'une distribuée car l'échange de nombreux messages entre voisins peut être considérablement énergivores.

Différents algorithmes de construction d'ensembles couvrants ont été développés ces dernières années. Il s'agit principalement d'heuristiques. Ces heuristiques s'appuient sur différents critères de sélection pour faire entrer un capteur dans un ensemble couvrant. L'un des critères récurrent [Berman et al., 2004a, Zorbas et al., 2010] est celui d'inclure en priorité dans un ensemble couvrant les capteurs couvrant les cibles les plus critiques (celles couvertes par le moins grand nombre de capteurs). Les autres approches relèvent de la programmation mathématique [Cardei et al., 2005, Xing et al., 2010, Pujari, 2011, Gentili and Raiconi, 2013, Yang and Liu, 2014] et des techniques associées (résolution avec une méthode Branch-and-Bound dans un solveur de programmes linéaires). Le problème, connu sous le nom de *Maximum Lifetime Problem* (MLP) est formulé comme un problème d'optimisation, l'objectif est généralement la maximisation du nombre d'ensembles couvrants sous des contraintes de couverture et d'énergie. La technique de génération de colonnes, largement connue pour gérer des programmes linéaires avec un grand nombre de variables, a également été utilisée [Castaño et al., 2014, Carrabs et al., 2015a] dans le cadre de cette problématique.

7.3/ MODÉLISATION

Dans cette partie, nous choisissons de nous restreindre au problème centralisé et aux modélisations sous forme de programmes linéaires.

7.3.1/ COUVERTURE DE CIBLES

Les premiers algorithmes fournis dans la littérature considèrent que les ensembles couvrants sont disjoints : un capteur appartient à un seul ensemble couvrant. Il est fait l'hypothèse que tous les capteurs sont homogènes en termes de capacité de communication, d'acquisition et de traitement des données, et du point de vue de leur niveau d'énergie. Les ensembles couvrants seront activés successivement, et chacun pendant la même période de temps. L'objectif fréquemment rencontré est celui de maximiser le nombre d'ensembles couvrants pour maximiser la durée de vie du réseau. D'autres modèles tiennent compte de la sous-couverture ou de la sur-couverture de certaines cibles pour formuler un objectif. Les auteurs [Berman et al., 2004a] et [Cardei et al., 2005] ont mis en évidence, chacun de leur côté, que l'usage d'ensembles couvrants non disjoints pouvait accroître la durée de vie du réseau. Dans ce cas, un capteur peut appartenir à plusieurs ensembles couvrants et la durée d'activation est calculée pour chaque ensemble couvrant.
Cette partie propose de synthétiser les différentes modélisations sous forme de programme linéaire qui ont été proposées dans la littérature pour le problème MLP qui a été prouvé NP-complet [Cardei et al., 2005].

7.3.1.1/ NOTATIONS

De manière à homogénéiser les différentes formulations, nous adoptons les notations suivantes dans tout le document. On considère un réseau de $|S|$ capteurs déployés aléatoirement et uniformément en grand nombre sur la zone d'intérêt et $|T|$ cibles.

- T : ensemble des cibles
- S : ensemble des capteurs
- $|T| = m$: nombre de cibles
- $|S| = n$: nombre de capteurs
- j : indice de capteur
- i : indice de cible
- E_j : niveau d'énergie (en unité de temps de vie) du capteur j
- α_{ij} : égal à 1 si la cible i est couverte par le capteur j, 0 sinon
- k : indice d'ensemble couvrant

Chaque capteur j peut être activé en continu pendant E_j unités de temps. Nous utilisons un modèle de couverture classique qui consiste à dire qu'une cible i est couverte par un capteur j si et seulement si la distance (distance euclidienne) entre i and j est inférieure au rayon de couverture R_s. On désigne par $T(j)$ l'ensemble des cibles couvertes par le capteur j, et inversement $S(i)$ l'ensemble des capteurs qui couvrent la cible i.

7.3.1.2/ FORMULATION POUR DES ENSEMBLES COUVRANTS DISJOINTS

Nous souhaitons obtenir le nombre maximal d'ensembles couvrants disjoints qu'il est possible de construire. Ce nombre est majoré par :

$$K = \min_{i=1..m} |S(i)| \tag{7.1}$$

En effet, chaque ensemble couvrant doit couvrir l'ensemble des cibles et un capteur ne peut appartenir qu'à un seul ensemble couvrant.

En fixant arbitrairement le nombre d'ensembles couvrants à une valeur K' comprise entre 1 et K, le problème de répartition des capteurs dans les ensembles couvrants est donné par la résolution d'un problème (7.2) de satisfaction de contraintes :

$$\begin{cases} \sum_{j \in S(i)} \mathbf{x_{j,k}} \geq 1, & \forall i \in T, \forall k = 1..K' \\ \sum_{k=1}^{k=K'} \mathbf{x_{j,k}} \leq 1, & \forall j \in S \\ \mathbf{x_{j,k}} \in \{0,1\} \end{cases} \tag{7.2}$$

Les variables $\mathbf{x_{j,k}}$ sont des variables binaires qui sont égales à 1 si le capteur j appartient à l'ensemble couvrant k et 0 sinon. La première contrainte indique que chaque cible doit être couverte dans chaque ensemble couvrant $k = 1..K'$. La seconde contrainte indique qu'un capteur doit appartenir à un seul ensemble couvrant. Pour un nombre K' (fixé arbitrairement) d'ensembles couvrants, il se peut que le problème ci-dessus n'ait pas de solution. Auquel cas, il faut diminuer la valeur K' et le problème (7.2) est résolu à nouveau. Nous avons proposé cette approche dans [Deschinkel, 2012] après avoir obtenu une borne inférieure de bonne qualité au moyen d'une heuristique. L'heuristique est détaillée dans le chapitre 9 de ce mémoire.

Objectif : Maximiser le nombre d'ensembles couvrants

Dans [Cardei and Du, 2005], les auteurs définissent le problème intitulé "Disjoint Set Covers" (DSC) pour le cas particulier du problème MLP pour des ensembles couvrants disjoints. Le problème s'énonce ainsi :
Étant donné un ensemble S de capteurs, il faut trouver une famille de taille maximale d'ensembles couvrants. Chaque ensemble couvrant C_k est un sous ensemble de S, chaque cible de T est couverte par C_k et pour tout ensemble couvrant C_k et C_l ($k \neq l$), $C_k \cap C_l = \emptyset$.

Dans [Cardei and Du, 2005], il est prouvé que le problème (DSC) est un problème NP-complet. Ce problème, pourtant cité fréquemment dans les articles de recherche, n'a pas à notre connaissance été formulé sous forme de programme linéaire. Nous nous sommes donc inspirés de la formulation proposée par [Xing et al., 2010] qui traite d'un problème un peu plus complexe, avec des capteurs hétérogènes équipés de plusieurs unités de captage, pour modéliser le problème (DSC).

On utilise les variables de décision suivantes :

- Variable binaire $\mathbf{z_k}$: $\mathbf{z_k} = 1$ signifie que C_k est un ensemble couvrant, $\forall k = 1...K$ où K est la borne supérieure définie par (7.1)
- Variable binaire $\mathbf{x_{j,k}}$: $\mathbf{x_{j,k}} = 1$ indique que le capteur j est actif dans l'ensemble couvrant C_k

La formulation du problème DSC sous forme de programme linéaire en nombres entiers est la suivante :

$$\begin{cases} \max \sum_{k=1}^{k=K} \mathbf{z_k} \\ \text{sous :} \\ \sum_{j \in S(i)} \mathbf{x_{j,k}} \geq \mathbf{z_k}, & \forall i \in T, \forall k \in K \\ \sum_{k=1}^{k=K} \mathbf{x_{j,k}} \leq 1, & \forall j \in S \\ \mathbf{x_{j,k}} \in \{0,1\} \\ \mathbf{z_k} \in \{0,1\} \end{cases} \tag{7.3}$$

La fonction objectif maximise le nombre total d'ensembles couvrants. La première contrainte indique que chaque cible est couverte au moins par un capteur dans chaque ensemble couvrant formé. La seconde contrainte garantie l'appartenance du capteur à un seul ensemble couvrant. Contrairement au problème de satisfaction de contraintes

défini en (7.2), la résolution du programme linéaire en nombre entiers (7.3) fournit une solution optimale qui nous permet de déduire le nombre d'ensembles couvrants qu'il est possible de générer ainsi que leur composition.

Objectif : Minimiser la sous-couverture et la sur-couverture

Les auteurs dans [Pedraza et al., 2006] abordent le problème (DSC) avec une approche différente et en relaxant, en partie, la contrainte de couverture de toutes les cibles pour chaque ensemble couvrant. En effet, certains applications ne nécessitent pas forcément la couverture à 100% de toute la région d'intérêt. Par exemple, une application dédiée à la détection de feu de forêt nécessite une couverture totale en saisons chaudes et peut requérir une couverture partielle en saisons pluvieuses [Li et al., 2011]. Un autre exemple, le suivi du mode de vie d'oiseaux peut nécessiter une couverture à 70% pendant leur sommeil et une couverture totale pendant les moments de la journée où ils sont actifs [Kumagai, 2004].
Pedraza et al. souhaitent construire un maximum d'ensembles couvrants disjoints C_k avec les propriétés suivantes : l'ensemble couvrant C_{k+1} généré doit couvrir un maximum de cibles mais jamais plus que le précédent C_k. Prenons pour illustrer cette propriété, un exemple de réseau avec 4 capteurs, $S = \{s_1, s_2, s_3, s_4\}$, 4 cibles $T = \{t_1, t_2, t_3, t_4\}$, et la matrice M de coefficients (α_{ij}) :

$$M = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

Une solution de couverture est : $C_1 = \{s_1, s_2\}$, $C_2 = \{s_3\}$ et $C_3 = \{s_4\}$. Dans ce cas, les ensembles couvrants couvrent respectivement 4,2 et 1 cibles. Une autre solution possible est de former les ensembles couvrants : $C_1 = \{s_1, s_3\}$ et $C_2 = \{s_2, s_4\}$, couvrant respectivement 4 et 3 cibles. La deuxième solution sera préférée à la première au sens de la définition d'optimalité apportée par les auteurs [Pedraza et al., 2006] et donnée dans la suite.

Définition d'optimalité
Une solution d'ensembles couvrants $C_1^1, C_2^1, ..., C_{L_1}^1$ est optimale si pour toute autre solution $C_1^2, C_2^2, ..., C_{L_2}^2$, on a :

- $|T(C_l^1)| \geq |T(C_l^2)|$ *pour chaque* $l \in \{1...min(L_1, L_2)\}$
- *et si* $|T(C_l^1)| = |T(C_l^2)|$ *pour chaque* $l \in \{1...min(L_1, L_2)\}$*, alors* $L_1 \geq L_2$

Ils proposent un programme linéaire (7.4) qui permet de construire des ensembles couvrants respectant le principe d'optimalité défini précédemment. La fonction objectif est une somme pondérée de variables traduisant une sous-couverture ($\mathbf{u_{i,k}}$) ou une sur-couverture des cibles ($\mathbf{o_{i,k}}$). Ils montrent que sous certaines conditions sur les poids $w_{o,i,k}$ et $w_{u,i,k}$, une solution optimale au sens de la définition précédente peut être trouvée.

$$\begin{cases} \min \sum_{k \in K} \sum_{i \in T} (w_{o,i,k} \mathbf{o_{i,k}} + w_{u,i,k} \mathbf{u_{i,k}}) & \\ \text{sous :} & \\ \sum_{j \in S(i)} \mathbf{x_{j,k}} - \mathbf{o_{i,k}} + \mathbf{u_{i,k}} = 1 & \forall i \in T, \forall k \in K \\ \sum_{k \in K} \mathbf{x_{j,k}} = 1 & \forall j \in S \\ \mathbf{o_{i,k}} \in \mathbb{N} & \forall i \in T, \forall k \in K \\ \mathbf{u_{i,k}} \in \{0, 1\} & \forall i \in T, \forall k \in K \\ \mathbf{x_{j,k}} \in \{0, 1\} & \forall j \in S, \forall k \in K \end{cases} \tag{7.4}$$

— $\mathbf{x_{j,k}}$: indique si le capteur j est dans l'ensemble couvrant C_k (1 si oui et 0 sinon)
— $\mathbf{o_{i,k}}$: *sur-couverture*, représente le nombre de capteurs moins un qui surveillent la cible i dans l'ensemble couvrant C_k
— $\mathbf{u_{i,k}}$: *sous-couverture*, indique si la cible i est couverte (1 si non et 0 si couverte) dans l'ensemble couvrant C_k

Si la cible i n'est pas couverte dans l'ensemble couvrant k, $\mathbf{u_{i,k}} = 1$, $\sum_{j \in S(i)} \mathbf{x_{j,k}} = 0$ et $\mathbf{o_{i,k}} = 0$ par définition, ainsi la contrainte d'égalité est satisfaite. Au contraire, si la cible i est couverte, $\mathbf{u_{i,k}} = 0$, $\mathbf{o_{i,k}} = \left(\sum_{j \in S(i)} \mathbf{x_{j,k}}\right) - 1$. Nous remarquons que le nombre à priori d'ensembles couvrants, noté ici K, doit être de valeur supérieure à celui calculé dans la formule (7.1) car le modèle (7.4) autorise la construction d'ensembles couvrants incomplets. Nous gardons néanmoins la notation K.

7.3.1.3/ FORMULATION POUR DES ENSEMBLES COUVRANTS NON DISJOINTS

Dans [Berman et al., 2004a] et [Cardei et al., 2005], les auteurs définissent le problème intitulé "Maximum Set Covers" (MSC) pour le cas particulier du problème MLP pour des ensembles couvrants non disjoints. Étant donné un ensemble S de capteurs, il faut trouver un nombre d'ensembles couvrants non nécessairement disjoints de durée d'activation totale la plus grande. Chaque ensemble couvrant C_k ($k = 1..K$) a une durée d'activation t_k, le temps durant lequel tous les capteurs de cet ensemble seront actifs pendant que les autres resteront en veille. Ici, la borne du nombre possible d'ensembles couvrants est différente et plus grande que celle calculée dans l'expression (7.1) pour le cas disjoint, cependant nous gardons la notation K. Il s'agit donc de maximiser la somme $\sum_{k=1..K} t_k$ sous les contraintes de couverture de toutes les cibles et sur les contraintes de réserve en énergie E_j des capteurs. Dans [Cardei et al., 2005], il est prouvé que le problème (MSC) est un problème NP-complet.
Le modèle mathématique associé à ce problème est le suivant :

$$\begin{cases} \max \sum_{k=1..K} \mathbf{t_k} \\ \text{sous :} \\ \sum_{k=1..K} \mathbf{x_{j,k}} \mathbf{t_k} \leq E_j, & \forall j \in S \\ \sum_{j \in S(i)} \mathbf{x_{j,k}} \geq 1, & \forall i \in T, \forall k = 1..K \\ \mathbf{t_k} \in \mathbb{R}^+ \\ \mathbf{x_{j,k}} \in \{0, 1\} \end{cases} \tag{7.5}$$

A nouveau, la variable $\mathbf{x_{j,k}}$ indique si le capteur j appartient ou non à l'ensemble couvrant C_k. La première contrainte garantit que le temps d'activation total de chaque capteur est inférieure à sa durée de vie initiale (E_j). La seconde contrainte assure que chaque cible $i \in T$ est couverte par au moins un capteur j dans chaque ensemble couvrant k. On remarque que la première contrainte n'est pas une contrainte linéaire à cause du terme "$\mathbf{x_{j,k}} \mathbf{t_k}$". Pour contourner cette difficulté, deux approches sont envisagées, une technique de relaxation est proposée dans [Cardei et al., 2005]. Tous les capteurs sont supposés avoir le même niveau d'énergie $E_j = 1$. La technique de relaxation consiste à définir la

variable $\mathbf{y_{j,k}} = \mathbf{x_{j,k}}\mathbf{t_k}$. Le problème devient :

$$\begin{cases} \max \sum_{k=1..K} \mathbf{t_k} \\ \text{sous :} \\ \sum_{k=1..K} \mathbf{y_{j,k}} \leq 1, & \forall j \in S \\ \sum_{j \in S(i)} \mathbf{y_{j,k}} \geq \mathbf{t_k}, & \forall i \in T, \forall k = 1..K \\ \mathbf{t_k} \in \mathbb{R}^+ \\ \mathbf{y_{j,k}} = 0 \text{ ou } \mathbf{t_k} \text{ et } \mathbf{y_{j,k}} \leq 1 \end{cases} \tag{7.6}$$

Le problème est relâché (la contrainte d'intégrité des variables $\mathbf{y_{jk}}$ est relâchée), sa version continue est :

$$\begin{cases} \max \sum_{k=1..K} \mathbf{t_k} \\ \text{sous :} \\ \sum_{k=1..K} \mathbf{y_{j,k}} \leq 1, & \forall j \in S \\ \sum_{j \in S(i)} \mathbf{y_{j,k}} \geq \mathbf{t_k}, & \forall i \in T, \forall k = 1..K \\ \mathbf{t_k} \in \mathbb{R}^+ \\ \mathbf{y_{j,k}} \in \mathbb{R}^+ \\ 0 \leq \mathbf{y_{j,k}} \leq \mathbf{t_k} \leq 1 \end{cases} \tag{7.7}$$

Une heuristique basée sur la résolution de programme linéaire est proposée.

De notre côté, nous avons développé une méthode reposant sur la technique de génération de colonnes [Deschinkel, 2012] qui sera détaillé dans le chapitre 10. Nous formulons l'hypothèse qu'il est possible de générer tous les ensembles couvrants non disjoints possibles : ils forment l'ensemble U et l'appartenance ou non d'un capteur j à un ensemble $u \in U$ est noté par l'indice binaire a_{ju}. La durée d'activation de l'ensemble u est notée t_u. Le problème MLP s'écrit :

$$\begin{cases} \max \sum_{u \in U} \mathbf{t_u} \\ \text{sous :} \\ \sum_{u \in U} a_{ju} \mathbf{t_u} \leq E_j, & \forall j \in S \\ \mathbf{t_u} \in \mathbb{R}^+ \end{cases} \tag{7.8}$$

7.3.2/ COUVERTURE DE LA ZONE D'INTÉRÊT

La couverture de zone d'intérêt signifie que chaque point à l'intérieur de la zone doit être couvert. En fait, il n'est pas facile de trouver un moyen de modéliser la couverture de tous les points de la zone. C'est pourquoi la plupart des approches consiste à ne considérer qu'un sous-ensemble de points suffisamment représentatifs de la zone à couvrir. Cette problématique est semblable à celle rencontrée pour l'optimisation et l'évaluation de la distribution de dose en curiethérapie concernant le choix du nombre et des positions des points de référence (voir la partie 5.1.3).
Par exemple, les auteurs [Deng et al., 2012] transforment le problème de couverture de zone en un problème de couverture de cibles. Les cibles sont les points d'intersection entre les disques de couverture des différents capteurs et avec les frontières de la zone d'intérêt. Dans [Huang and Tseng, 2005] les auteurs proposent un algorithme en $O(nd\ log\ d)$ pour calculer la couverture du périmètre de chaque capteur (il faudrait dire le périmètre du disque couvert par le capteur). d est le nombre maximal de voisins d'un

capteur et n le nombre total de capteurs.
Dans les travaux de thèse [Idrees, 2015], nous avons exploré ces deux pistes :

— couverture d'un ensemble discrets de points représentatifs de la zone d'intérêt,
— couverture du périmètre des capteurs.

Nous présentons dans les deux parties qui suivent les modèles d'optimisation développés dans les deux cas. Contrairement à ce qui a été présenté dans la partie précédente sur la couverture de cibles, nous nous sommes placés dans un cadre légèrement différent. Nous avons considéré un réseau, avec des capteurs homogènes en termes de capacité de communication, d'acquisition et de traitement des données, mais hétérogènes du point de vue de leur niveau d'énergie. Au lieu de planifier à l'avance l'activité des nœuds en générant de manière centralisée, dés le début, tous les ensembles couvrants, nous nous plaçons dans le cadre d'une planification périodique et semi-distribuée. Pour plus de détails sur le protocole utilisé, nous invitons le lecteur à se référer à la thèse [Idrees, 2015]. Dans la suite, nous ne présentons que les modèles d'optimisation.

7.3.2.1/ MODÈLE DE COUVERTURE DE POINTS PRIMAIRES

La modèle d'optimisation que nous proposons repose sur celui présenté dans [Pedraza et al., 2006] et déjà évoqué dans la partie 7.3.1.2. Cette-fois ci, la couverture de zone se traduit par la couverture d'un ensemble P points primaires. Les détails sur le choix et le nombre de points primaires sont donnés dans [Idrees et al., 2014]. Ces points primaires sont interprétés comme des cibles pour le modèle de [Pedraza et al., 2006].

Choix des points primaires
Connaissant la position du nœud capteur (centré au point de coordonnées (p_x, p_y)) et son rayon de couverture R_s, on définit jusqu'à 25 points primaires X_1 à X_{25} (voir figure 7.1). Les coordonnées de ces points sont les suivantes :

$X_1 = (p_x, p_y)$
$X_2 = (p_x + R_s * (1), p_y + R_s * (0))$
$X_3 = (p_x + R_s * (-1), p_y + R_s * (0))$
$X_4 = (p_x + R_s * (0), p_y + R_s * (1))$
$X_5 = (p_x + R_s * (0), p_y + R_s * (-1))$
$X_6 = (p_x + R_s * (\frac{-\sqrt{2}}{2}), p_y + R_s * (\frac{\sqrt{2}}{2}))$
$X_7 = (p_x + R_s * (\frac{\sqrt{2}}{2}), p_y + R_s * (\frac{\sqrt{2}}{2}))$
$X_8 = (p_x + R_s * (\frac{-\sqrt{2}}{2}), p_y + R_s * (\frac{-\sqrt{2}}{2}))$
$X_9 = (p_x + R_s * (\frac{\sqrt{2}}{2}), p_y + R_s * (\frac{-\sqrt{2}}{2}))$
$X_{10} = (p_x + R_s * (\frac{-\sqrt{2}}{2}), p_y + R_s * (0))$
$X_{11} = (p_x + R_s * (\frac{\sqrt{2}}{2}), p_y + R_s * (0))$
$X_{12} = (p_x + R_s * (0), p_y + R_s * (\frac{\sqrt{2}}{2}))$
$X_{13} = (p_x + R_s * (0), p_y + R_s * (\frac{-\sqrt{2}}{2}))$
$X_{14} = (p_x + R_s * (\frac{\sqrt{3}}{2}), p_y + R_s * (\frac{1}{2}))$
$X_{15} = (p_x + R_s * (\frac{-\sqrt{3}}{2}), p_y + R_s * (\frac{1}{2}))$
$X_{16} = (p_x + R_s * (\frac{\sqrt{3}}{2}), p_y + R_s * (\frac{-1}{2}))$
$X_{17} = (p_x + R_s * (\frac{-\sqrt{3}}{2}), p_y + R_s * (\frac{-1}{2}))$

$X_{18} = (p_x + R_s * (\frac{\sqrt{3}}{2}), p_y + R_s * (0))$
$X_{19} = (p_x + R_s * (\frac{-\sqrt{3}}{2}), p_y + R_s * (0))$
$X_{20} = (p_x + R_s * (0), p_y + R_s * (\frac{1}{2}))$
$X_{21} = (p_x + R_s * (0), p_y + R_s * (-\frac{1}{2}))$
$X_{22} = (p_x + R_s * (\frac{1}{2}), p_y + R_s * (\frac{\sqrt{3}}{2}))$
$X_{23} = (p_x + R_s * (\frac{-1}{2}), p_y + R_s * (\frac{\sqrt{3}}{2}))$
$X_{24} = (p_x + R_s * (\frac{-1}{2}), p_y + R_s * (\frac{-\sqrt{3}}{2}))$
$X_{25} = (p_x + R_s * (\frac{1}{2}), p_y + R_s * (\frac{-\sqrt{3}}{2})).$

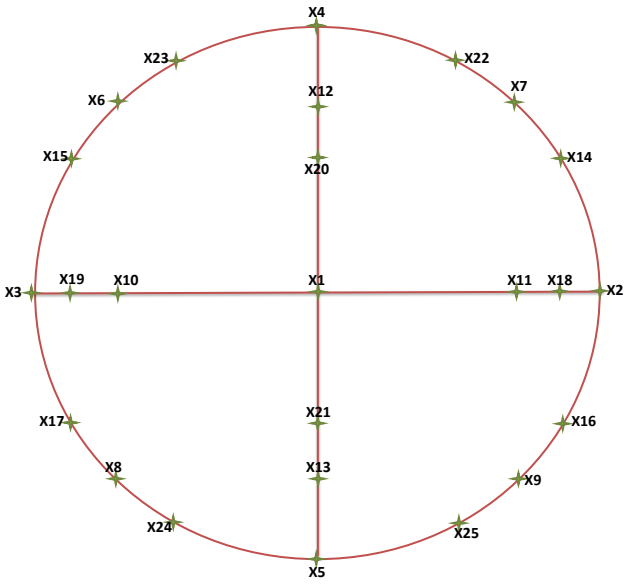


FIGURE 7.1 – Couverture du disque remplacé par la couverture d'au plus 25 points primaires

Dans notre étude, nous avons comparé les performances de notre protocole lorsque celui-ci (ou plus précisément le modèle d'optimisation) se restreint à la couverture de 5 points primaires, de 9, de 13, de 17, de 21 et de 25 points primaires. L'analyse de plusieurs critères de performance montre qu'il est plus efficace de n'utiliser que 5 points primaires.

Minimisation simultanée de la sous et sur-couverture des points primaires
Il n'est plus question d'ensemble couvrant C_k, puisque le processus de décision d'activation des capteurs est réalisé à chaque nouvelle période. Les variables binaires considérées sont les variables X_s, qui indiquent si parmi les capteurs ayant une provision d'énergie suffisante (notons $A \subseteq S$ l'ensemble de ces capteurs), le capteur s sera activé dans la prochaine période de mesure.

Pour un point primaire p, soit α_{sp} l'indice de couverture du point p.

$$\alpha_{sp} = \begin{cases} 1 & \text{si le point primaire } p \text{ est couvert} \\ & \text{par le capteur } s, \\ 0 & \text{sinon.} \end{cases} \tag{7.9}$$

Le nombre de capteurs actifs qui couvrent le point p est égal à $\sum_{s\in S} \alpha_{sp} * X_s$ où :

$$X_s = \begin{cases} 1 & \text{si le capteur } s \text{ est activé,} \\ 0 & \text{sinon.} \end{cases} \tag{7.10}$$

On définit la variable de sur-couverture Θ_p par :

$$\Theta_p = \begin{cases} 0 & \text{si le point primaire} \\ & p \text{ n'est pas couvert,} \\ \left(\sum_{s\in S} \alpha_{sp} * X_s\right) - 1 & \text{sinon.} \end{cases} \tag{7.11}$$

Plus précisément, Θ_p représente le nombre de capteurs actifs moins un qui couvrent le point primaire p.
La variable de sous-couverture U_p du point primaire p est défini par :

$$U_p = \begin{cases} 1 & \text{si le point primaire } p \text{ n'est pas couvert,} \\ 0 & \text{sinon.} \end{cases} \tag{7.12}$$

Notre problème d'optimisation est formulé ainsi :

$$\begin{cases} \min \sum_{p\in P}(w_\theta \Theta_p + w_U U_p) & \\ \text{sous :} & \\ \sum_{s\in S} \alpha_{sp} X_s - \Theta_p + U_p = 1, & \forall p \in P \\ \Theta_p \in \mathbb{N}, & \forall p \in P \\ U_p \in \{0,1\}, & \forall p \in P \\ X_s \in \{0,1\}, & \forall s \in A \end{cases} \tag{7.13}$$

La contrainte d'égalité indique que chaque point primaire p doit être couvert par au moins un capteur et, que si ce n'est pas toujours le cas, les variables de sous-couverture et de sur-couverture permettent d'équilibrer l'équation en prenant des valeurs positives. La fonction objectif est une somme pondérée de variables de sous et sur-couverture. L'objectif étant de limiter la sur-couverture en activant un nombre minimal de capteurs, mais tout en limitant simultanément la sous-couverture. En choisissant w_U bien plus grand que w_θ, la couverture d'un nombre maximal de points $p \in P$ est assurée. Et pour un même nombre de points p couverts, la solution avec un nombre minimal de capteurs actifs est préférée, ce qui permet de préserver de l'énergie à chaque période et donc d'étendre la durée de vie du réseau.

7.3.2.2/ MODÈLE DE COUVERTURE DU PÉRIMÈTRE DES CAPTEURS

La modélisation que nous proposons repose sur les principes de couverture des périmètres des disques couverts par les capteurs. Ces principes ont été décrits dans [Huang and Tseng, 2005]. Ils s'expriment ainsi : le périmètre d'un capteur (pour être plus précis, il faudrait dire le périmètre du disque couvert par le capteur) est dit couvert si tous les points sur son périmètre sont couverts par au moins un autre capteur que lui-même. Les auteurs [Huang and Tseng, 2005] ont montré que la zone de surveillance du réseau est l-couverte (chaque point dans la zone est couverte par au moins l capteurs) si et seulement si le périmètre de chaque capteur du réseau est l-couvert (son périmètre est couvert par l autres capteurs). Leur étude donne juste un algorithme de calcul de

couverture du périmètre d'un capteur mais ne propose pas d'optimiser la gestion des périodes d'activation des capteurs. Nous proposons un modèle d'optimisation qui répond à cette problématique.

Notations, définitions et modèle géométrique

Pour introduire plus facilement quelques définitions, nous partons d'un exemple de couverture d'un capteur 0 schématisé sur la figure 7.2(a). Sur cette figure, le capteur 0 a neuf voisins. Pour chaque voisin, deux points d'intersection ont été reportés sur le périmètre du disque couvert par le capteur 0. Ces points sont notés, pour chaque voisin i, par iL and iR, désignant respectivement l'intersection gauche ou droite du point de vue du voisin. L'ensemble de ces points d'intersection subdivise le périmètre du capteur 0 en portions que nous appellerons arcs ou intervalles.

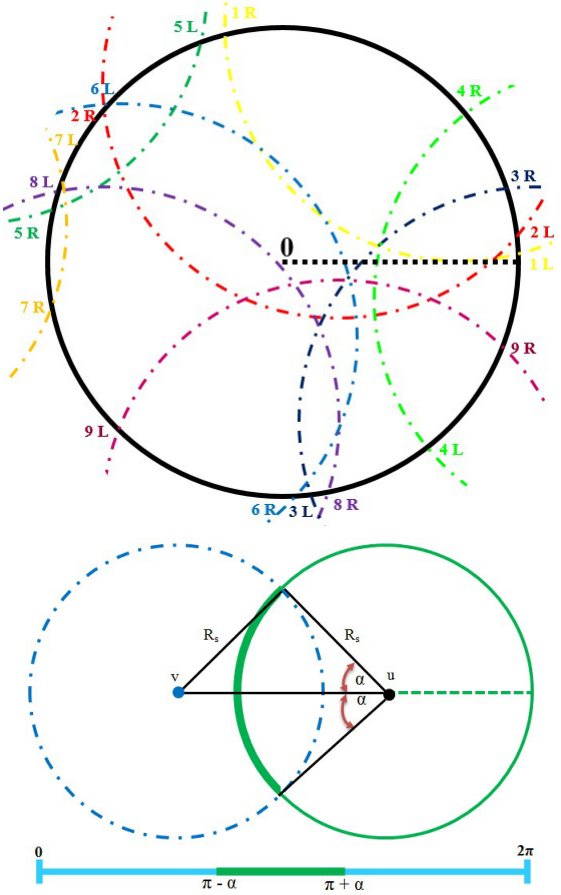


FIGURE 7.2 – (a) Couverture du périmètre du capteur 0 et (b) portion du périmètre de u couvert par le capteur v.

Figure 7.2(b) décrit les informations géométriques utiles pour déterminer la localisation des points d'intersection gauche et droite d'un capteur v voisin d'un capteur u et couvrant une partie de son périmètre. Le nœud v est à gauche du noeud u sur cette figure, avec les cordonnées suivantes : (v_x, v_y) et (u_x, u_y). La distance euclidienne entre les deux nœuds u et v est définie de cette manière :

$$Dist(u,v) = \sqrt{(u_x - v_x)^2 + (u_y - v_y)^2},$$

tandis que l'angle α est obtenu grâce à la formule :

$$\alpha = \arccos\left(\frac{Dist(u,v)}{2R_s}\right).$$

L'arc du périmètre de u couvert par v est entièrement défini par l'intervalle angulaire $[\pi - \alpha, \pi + \alpha]$.

Chaque couple de points d'intersection (iL, iR) est placé sur l'intervalle angulaire $[0, 2\pi)$ dans le sens inverse des aiguilles d'une montre, conduisant à un découpage de l'intervalle en sous-intervalles. La figure 7.2(a) illustre les arcs des neufs voisins du noeud 0 et le tableau 7.1 donne les coordonnées angulaires des points d'intersection gauche et droite sur l'intervalle $[0, 2\pi)$. La figure 7.3 représente le découpage du périmètre du capteur 0 en plusieurs intervalles pour lesquels le niveau de couverture (interprété ici comme le nombre de capteurs couvrant une portion du périmètre) est reporté. Par exemple, l'arc compris entre les points $5L$ et $6L$ a un niveau de couverture maximal pouvant atteindre 3 (cette valeur est donnée en bas de la figure 7.3). Cela signifie que cet arc sur le périmètre du capteur 0 peut être couvert par au moins deux autres capteurs. Le tableau 7.1 regroupe pour chaque intervalle de couverture (CI) le niveau maximal de couverture possible et les numéros de capteurs impliqués dans cette couverture.

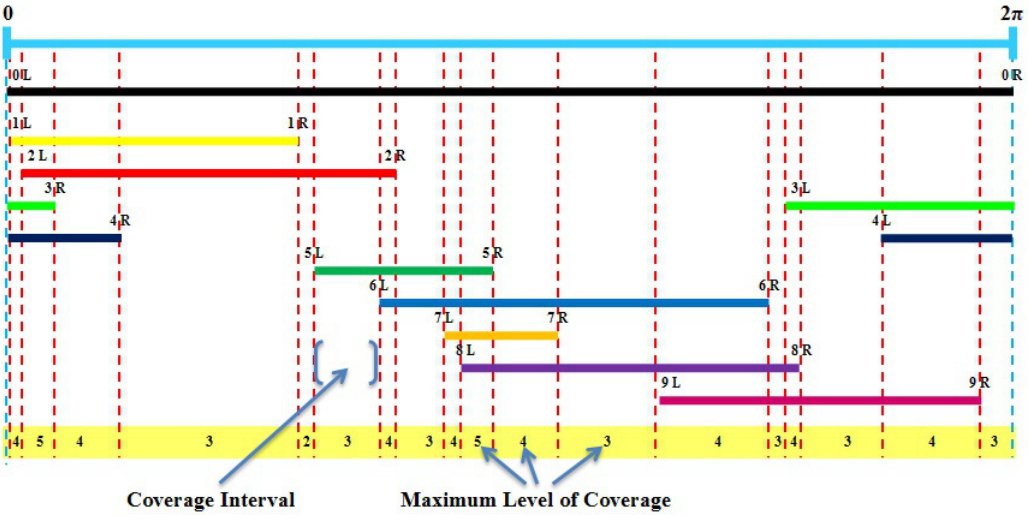


FIGURE 7.3 – Niveau maximal de couverture du capteur 0.

Formulation du problème de couverture

Dans cette partie, nous allons formuler mathématiquement le problème de couverture du périmètre des capteurs. Il a été démontré dans [Hung and Lui, 2010] que ce problème est NP-difficile. Dans cet article, les auteurs étudient la couverture du périmètre d'un large objet exigeant une surveillance. Pour la formulation que nous proposons, l'objet à surveiller est le capteur lui-même (ou plus précisément son disque de couverture).

Les notations suivantes sont utilisées pour écrire le modèle complet. I_j représente l'ensemble des intervalles de couverture définis dans la partie 7.3.2.2 pour le capteur j. Pour un intervalle de couverture c, le paramètre binaire a_{cs}^j indique si le capteur s participe à la couverture de l'intervalle c du capteur j.

$$a_{cs}^j = \begin{cases} 1 & \text{si le capteur } s \text{ participe à la} \\ & \text{couverture de l'intervalle } c \text{ du capteur } j, \\ 0 & \text{sinon.} \end{cases} \tag{7.14}$$

TABLE 7.1 – Intervalles de couverture et capteurs impliqués

angle α	Extrémité gauche de l'intervalle	Extrémité droite de l'intervalle	Niveau maximal de couverture	Ensemble de capteurs impliqués dans la couverture de l'intervalle				
0.0291	1L	2L	4	0	1	3	4	
0.104	2L	3R	5	0	1	3	4	2
0.3168	3R	4R	4	0	1	4	2	
0.6752	4R	1R	3	0	1	2		
1.8127	1R	5L	2	0	2			
1.9228	5L	6L	3	0	2	5		
2.3959	6L	2R	4	0	2	5	6	
2.4258	2R	7L	3	0	5	6		
2.7868	7L	8L	4	0	5	6	7	
2.8358	8L	5R	5	0	5	6	7	8
2.9184	5R	7R	4	0	6	7	8	
3.3301	7R	9R	3	0	6	8		
3.9464	9R	6R	4	0	6	8	9	
4.767	6R	3L	3	0	8	9		
4.8425	3L	8R	4	0	3	8	9	
4.9072	8R	4L	3	0	3	9		
5.3804	4L	9R	4	0	3	4	9	
5.9157	9R	1L	3	0	3	4		

Remarquons que $a^j_{cj} = 1$ par définition de l'intervalle.

Plusieurs autres variables sont définies. Chaque variable binaire X_s précise si le capteur s (parmi un ensemble $A \subseteq S$ de capteurs toujours en vie au moment de la prise de décision) sera actif durant la période de surveillance considérée ($X_s = 1$ si le capteur s est activé, 0 sinon). La variable M^j_c mesure la sous-couverture de l'intervalle c pour le capteur j. Parallèlement, la sur-couverture pour le même intervalle c du capteur j est notée V^j_c. Pour maintenir un niveau de couverture égal à l tout le long du périmètre du capteur j, au moins l capteurs impliqués dans la couverture de chacun de ses intervalles $c \in I_j$ doivent être actifs. D'après les notations précédentes, le nombre de capteurs actifs dans l'intervalle c du capteur j est donné par l'expression $\sum_{s \in A} a^j_{cs} X_s$.

Pour étendre la durée de vie du réseau, l'objectif est d'activer un nombre minimal de capteurs pour chaque période tout en assurant la meilleure couverture de la zone d'intérêt. Au fur et à mesure des périodes, le nombre de capteurs en vie (avec une provision d'énergie suffisante pour tenir éveillé une période) diminue et il devient impossible d'atteindre le niveau de couverture souhaité. C'est pourquoi les variables M^j_c et V^j_c sont introduites dans le modèle, elles mesurent la déviation entre le nombre requis de capteurs pour chaque intervalle de couverture et le nombre effectif. Nous cherchons à minimiser ces déviations, d'une part pour activer un nombre minimal de capteurs pour assurer la couverture totale de la zone, et si la couverture totale ne peut pas être atteinte, nous essayons de nous en approcher.

Le problème d'optimisation est formulé de la manière suivante :

$$\begin{aligned} &\text{Min} \sum_{j\in S}\sum_{c\in I_j}(\alpha_c^j\ M_c^j + \beta_c^j\ V_c^j) \\ &\text{sous :} \\ &\sum_{s\in A}(a_{cs}^j\ X_s) + M_c^j \geq l \quad \forall j \in S, \forall c \in I_j \\ &\sum_{s\in A}(a_{cs}^j\ X_s) - V_c^j \leq l \quad \forall j \in S, \forall c \in I_j \\ &X_s \in \{0,1\}, \forall s \in A \\ &M_c^j, V_c^j \in \mathbb{R}^+ \end{aligned} \tag{7.15}$$

Admettons qu'un niveau de couverture l soit requis pour un capteur j. On dit que ce capteur est sous-couvert (respectivement sur-couvert) si le niveau de couverture de l'un de ses intervalles est moins grand (respectivement plus grand) que l. Si le capteur j est sous-couvert, cela signifie qu'il y au moins un de ses intervalles de couverture (disons c) pour lequel le nombre de capteurs actifs (noté l^c) couvrant l'intervalle c est inférieur à l et dans ce cas : $M_c^j = l - l^c$, $V_c^j = 0$. Inversement, si le capteur j est sur-couvert, il y au moins un de ses intervalles de couverture (disons d) pour lequel le nombre de capteurs actifs (noté l^d) couvrant l'intervalle d est supérieur à l et dans ce cas : $M_d^j = 0$, $V_d^j = l^d - l$. Les poids α_c^j and β_c^j sont positifs ou nuls et sont fixés selon l'importance accordée à la satisfaction des niveaux de couverture des différents intervalles. Par exemple, si on veut favoriser la couverture d'une partie de la zone d'intérêt, on peut mettre des poids plus forts aux intervalles des capteurs présents sur cette partie. Cette formulation de programme linéaire entier mixte est inspiré du modèle développé en curiethérapie LDR [Lee et al., 1999] pour optimiser la distribution de dose dans la tumeur et autour. Ce problème est décrit dans la partie 5.2.2 de ce document. Le choix des valeurs pour les poids α et β doit être fait selon les besoins de l'application. α doit être suffisamment élevé pour empêcher la sous-couverture et atteindre un taux de couverture de la zone le plus grand possible. β doit être suffisamment élevé pour éviter la sur-couverture et limiter ainsi le nombre de capteurs actifs.

7.4/ CONCLUSION

Ce chapitre aborde le problème de maximisation de la durée de vie d'un réseau de capteurs sous l'angle de la recherche opérationnelle. Nous nous sommes intéressés aux formulations sous forme de programme linéaire qui permettent une génération centralisée d'ensembles couvrants (groupe de capteurs activés sur une période). Les ensembles couvrants générés, disjoints ou non disjoints, couvrent totalement ou partiellement des cibles ou des points spécifiques appelés points primaires. Nous présentons des modèles existants dans la littérature et proposons de nouveaux modèles dans les cas suivants :

- maximisation du nombre d'ensembles couvrants disjoints couvrant la totalité des cibles (programmes linéaires (7.2) et (7.3))
- maximisation du nombre d'ensembles couvrants non disjoints par une technique de génération de colonnes (programme linéaire (7.5))

- minimisation simultanée de la sous-couverture et sur-couverture de points primaires (programme linéaire (7.13))
- maximisation de la couverture du périmètre de chaque capteur induisant une couverture de la zone d'intérêt (programme linéaire (7.15))

II

RÉSOLUTION PAR DÉCOMPOSITION DU PROBLÈME

8

UNE HEURISTIQUE EN DEUX TEMPS POUR LA MINIMISATION DU NOMBRE DE REGISTRES : SIRALINA

L'utilisation d'une méthode classique de Branch and Bound pour résoudre le programme linéaire (6.8) permet de traiter, en pratique, uniquement des petites instances de problèmes (en pratique, des DDG de taille maximale de 12 noeuds), sinon les temps de calcul deviennent prohibitifs et les besoins en mémoire explosent. C'est pourquoi nous avons développé une heuristique efficace de résolution appelée SIRALINA.

Notre stratégie de résolution repose sur l'analyse du modèle de contraintes. Le problème fait apparaître des contraintes d'ordonnancement et des contraintes d'affectation, et les distances de réutilisation sont le lien entre les deux ensembles de contraintes. Nous proposons une décomposition du problème en deux sous-problèmes :

— Un problème d'ordonnancement : trouver un ordonnancement pour lequel les distances de réutilisation sont les plus petites possibles.
— Un problème d'affectation : sélectionner les paires de tâches qui partageront le même registre processeur.

8.1/ PRÉALABLES

Si une paire de tâches $(T_i, T_j) \in V^R \times V^R$ est reliée par un arc de réutilisation, sa distance de réutilisation doit satisfaire l'inégalité (6.4), où $\theta_{i,j} = 1$. Cette inégalité fixe une borne inférieure pour chaque distance de réutilisation potentielle. Si $(T_i, T_j) \in V^R \times V^R$ est un arc de réutilisation (E^{reuse} représente l'ensemble des arcs de réutilisation) alors :

$$\forall (T_i, T_j) \in E^{reuse}, \quad \mu_{i,j} \geq \frac{1}{p}(\sigma_{K_i} - \delta_w(T_j) - \sigma_j) \tag{8.1}$$

Si $(T_i, T_j) \in V^R \times V^R$ n'est pas un arc de réutilisation alors $\mu_{i,j} = 0$ d'après l'inégalité 6.5.

$$\forall (T_i, T_j) \notin E^{reuse}, \quad \mu_{i,j} = 0$$

L'agrégation des contraintes 8.1pour chaque arc de réutilisation fournit une borne inférieure à la fonction objectif

$$z = \sum_{(T_i,T_j) \in V^R \times V^R} \mu_{i,j} \geq \frac{1}{p} \sum_{(T_i,T_j) \in E^{reuse}} (\sigma_{K_i} - \delta_w(T_j) - \sigma_j)$$

La partie droite de cette inégalité peut se décomposer en deux sommes par bijection de la relation de réutilisation de V^R dans V^R :
$\sum_{(T_i,T_j)\in E^{reuse}} \sigma_{K_i} - \delta_w(T_j) - \sigma_j$

$$\begin{aligned} &= \sum_{i\in V^R} \sigma_{K_i} - \sum_{j\in V^R} (\delta_w(T_j) + \sigma_j) \\ &= \left(\sum_{i\in V^R} \sigma_{K_i} - \sum_{j\in V^R} \sigma_j\right) - \sum_{j\in V^R} \delta_w(T_j) \end{aligned} \tag{8.2}$$

Cette expression nous permet de déduire une borne inférieure du nombre de registres nécessaires. Il parait ici judicieux de chercher un ordonnancement pour lequel cette valeur est minimale. Dans cette expression, nous pouvons ignorer le terme constant $\sum_{j\in V^R} \delta_w(T_j)$ Nous nous intéressons donc au *Problème d'ordonnancement (P)* ainsi défini :

$$\begin{cases} \min \sum_{i\in V^R} \sigma_{K_i} - \sum_{j\in V^R} \sigma_j \\ \text{sous :} \\ \sigma_j - \sigma_i \geq \delta(e) - p \times \lambda(e), & \forall e = (T_i, T_j) \in E \\ \sigma_{K_i} - \sigma_j \geq \delta_r(T_j) + p \times \lambda(e), & \forall e = (T_i, T_j) \in E^R \end{cases} \tag{8.3}$$

La matrice de contraintes du système 8.3 est une matrice d'incidence [Schrijver, 1987] du graphe DDG auquel on a ajouté les noeuds "tueurs" (voir Figure 6.2(c)). Le nombre de variables entières est $O(|V|)$ et le nombre de contraintes linéaires est $O(|E|)$. Comme il s'agit d'une matrice d'incidence, c'est aussi une matrice totalement unimodulaire (le déterminant de chaque sous matrice carrée est égale à 0 ou à $\pm$ 1) et la résolution du programme linéaire en nombre entiers par la méthode du simplexe fournit une solution optimale entière au problème [Schrijver, 1987].

Une fois les valeurs optimales σ_i^* trouvées pour les tâches $T_i \in V^R$ et les valeurs optimales $\sigma_{K_i}^*$ pour chaque noeud "tueur" K_i, la valeur minimale de chaque distance de réutilisation potentielle est égale à $\overline{\mu_{i,j}} = \lceil \frac{\sigma_{K_i}^* - \delta_w(T_j) - \sigma_j^*}{p} \rceil$ d'après l'inégalité 8.1. Connaissant la valeur de distance de réutilisation $\overline{\mu_{i,j}}$ si la tâche T_j réutilise le registre libérée par T_i, l'allocation de stockage consiste à déterminer quelle tâche réutilise quel registre libéré. Ce problème peut être modélisé comme un problème d'affectation linéaire. Nous nous intéressons donc au *Problème d'affectation linéaire (A)* ainsi défini :

$$\begin{cases} \min \quad \sum_{(T_i,T_j)\in V^R\times V^R} \overline{\mu_{i,j}} \times \theta_{i,j} \\ \text{sous :} \\ \sum_{T_j\in V^R} \theta_{i,j} = 1, & \forall T_i \in V^R \\ \sum_{T_i\in V^R} \theta_{i,j} = 1, & \forall T_j \in V^R \\ \theta_{i,j} \in \{0, 1\} \end{cases} \tag{8.4}$$

où $\overline{\mu_{i,j}}$ est une valeur fixe pour chaque arc $e = (T_i, T_j) \in V^R \times V^R$.

8.2/ ÉTAPES DE RÉSOLUTION DE L'HEURISTIQUE SIRALINA

Pour une période donnée $p \in \mathbb{N}$, nous proposons de résoudre le problème de minimisation de registres avec l'heuristique suivante :

- Résoudre le problème (P) pour obtenir les valeurs optimales σ_i^* pour chaque tâche $T_i \in V^R$ et les valeurs optimales $\sigma_{K_i}^*$ pour chaque noeud "tueur" K_i.
- Calculer les distances $\overline{\mu_{i,j}} = \left\lceil \frac{\sigma_{K_i}^* - \delta_w(T_j) - \sigma_j^*}{p} \right\rceil$ pour chaque couple $(T_i, T_j) \in V^R \times V^R$.
- Résoudre le problème d'affection linéaire (A) avec la méthode Hongroise [Kuhn, 1955] (voir annexe B) qui permet une résolution en temps polynomial ($O(|V^R|^3)$) pour obtenir les valeurs d'affectation optimale $\theta_{i,j}^*$.
- Si $\theta_{i,j}^* = 1$ pour la tâche $(T_i, T_j) \in V^R \times V^R$, alors (T_i, T_j) devient un arc de réutilisation avec une distance égale à $\mu_{i,j} = \overline{\mu_{i,j}}$.

8.3/ RÉSULTATS EXPÉRIMENTAUX

Dans cette partie, nous présentons les nombreux résultats de tests réalisés sur des milliers de DDG issus de benchmarks bien connus. Nos benchmarks proviennent d'une sélection de cinq familles d'application : 2 des familles correspondent à des applications embarquées ((MEDIABENCH et LAO), les 3 autres à des applications générales de calcul intensif (SPEC2000-CINT, SPEC2000-CFP et SPEC2006). Ces applications se composent de nombreux fichiers de programmes écrits en langage C et C++ à optimiser par compilation. Notre heuristique SIRALINA a été incorporée dans un dispositif de compilation industrielle de la société STMicroelectronics.

8.3.1/ CADRE EXPÉRIMENTAL

Pour évaluer l'efficacité de notre heuristique, nous la comparons avec deux autres heuristiques existantes (désignées ici par H2 et H3) dont la méthodologie est détaillée dans [Touati and Eisenbeis, 2004]. Nous comparons également les résultats obtenus par SIRALINA à ceux obtenus par la résolution du programme linéaire entier avec la méthode Branch and Bound. Comme il est impossible de résoudre à l'optimalité des instances de DDG de plus de 12 noeuds en un temps de calcul raisonnable, nous avons limité le temps de résolution à 10 secondes (temps maximal autorisé pour compiler une seule boucle de programme informatique). La valeur de la solution retenue est alors la meilleure solution trouvée dans les 10 secondes : cela définit une heuristique naïve basée sur le modèle linéaire entier. Cette heuristique naïve est appelée H1. Comparativement, SIRALINA fournit toujours une solution approchée en moins d'une seconde.

Le nombre total de DDGs testés est de 6748. Leurs différentes tailles (nombre de noeuds et d'arcs) sont représentées par des diagrammes de quartiles (voir figure 8.1). Ce type de diagramme [Crawley, 2007] est utile pour décrire graphiquement la répartition des données au moyen de cinq chiffres significatifs (la plus petite valeur, le quartile inférieur ($Q1 = 25\%$), la médiane ($Q2 = 50\%$), le quartile supérieur ($Q3 = 75\%$), et les plus grandes valeurs). La partie gauche de la figure 8.1 représente le nombre de noeuds par famille, tandis que la partie droite représente le nombre d'arcs. Les 5 "boîtes" correspondent aux 5 familles de référence (LAO, Mediabench, SPEC, etc.) ; 1) le petit trait horizontal inférieur

est la valeur minimale, 2) le rectangle représente le premier et le troisième quartile, 3) le trait horizontale coupant le rectangle représente la médiane, 4) le dernier point supérieur représente la valeur maximale. On peut remarquer par exemple que 50% des DDGs testés ont entre 15 et 30 noeuds, et entre 50 et 100 arcs. On remarque aussi que la taille du plus grand DDG est supérieure à 500 noeuds et 2000 arcs.

Nous cherchons à minimiser le nombre de registres utilisés dans un processeur de type VLIW (processeur de la famille ST231 de STMicroelectronics). Cette famille de processeurs possède des registres de type BR (registres de branchements) et GR (registres de type entier). Les registres de type BR sont chargés de stocker les données booléennes du programme et les registres de type GR les autres données numériques (entiers). SIRALINA est capable d'optimiser chaque type de registre séparément. Par exemple, lorsque l'on considère le type GR, l'ensemble de tâches V^R du DDG devient l'ensemble des tâches produisant des résultats de type GR. Pour BR, V^R est l'ensemble des tâches écrivant dans des registres de type BR. La distribution du nombre de noeuds et d'arcs pour chaque type (BR ou GR) est donnée par les différents diagrammes de la figure 8.2. Le nombre moyen de tâches produisant des résultats de type BR est d'environ 4, et le nombre moyen d'arcs est égal à 5. Ces valeurs sont multipliées par 5 et plus pour les registres de type GR (environ 20 noeuds et 30 des arcs).

Nous avons utilisé le solveur commercial de programmes linéaires ILOG CPLEX 10.2. Les tests ont été effectués sur une station Linux équipée d'un processeur Pentium 2.33 Ghz processor, et 4 GB de mémoire vive.

8.3.2/ COMPARAISONS AVEC D'AUTRES HEURISTIQUES

Cette partie présente une synthèse des comparaisons de performances entre SIRALINA les différentes méthodes (H1, H2 et H3). H2 et H3 sont d'autres heuristiques basées sur la résolution de programmes linéaires entiers simplifiés [Touati and Eisenbeis, 2004]. Ces heuristiques sont non polynomiales mais permettent de résoudre des instances de taille moyenne (ce qui est impossible avec la méthode exacte). Nous avons dû limiter leur temps d'exécution à une seconde pour une valeur de période p, sinon les temps de calcul sont trop élevés (de l'ordre de quelques minutes à plusieurs heures suivant la taille du DDG) : en compilation interactive, il ne faut pas dépasser une seconde de calcul pour optimiser une boucle (un DDG) pour une période.

Le tableau 8.1 (respectivement 8.3) donne le nombre de problèmes (en pourcentage) pour lequel le nombre de registres de type BR (resp. GR) fourni par SIRALINA pour la période minimale $p = MEP$ (période minimale d'éxécution- voir chapitre 6) est strictement inférieur au nombre de registres calculé avec les trois autres méthodes. Le tableau 8.2 (respectivement 8.4) donne le pourcentage de cas où SIRALINA fournit le même résultat que les autres méthodes. On s'aperçoit aisément que dans la plupart des cas, SIRALINA calule au moins une solution aussi bonne que les trois autres méthodes. SIRALINA surpasse toujours l'heuristique 3. Pour les registres de type BR, il y a peu de cas où SIRALINA est strictement meilleure que l'heuristique 1. Lorsque le nombre de nœuds V^R est plus élevé, à savoir pour les registres de type GR, SIRALINA est plus efficace. Nous nous attendions à ce constat : le cas avec des registres de type BR peut être résolu à l'optimalité avec l'heuristique 1 dans le temps limite car le nombre de nœuds V^R et d'arcs E^R est relativement faible. Mais pour les grandes instances, avec des registres de type GR, SIRALINA produit des résultats nettement meilleurs.

Famille de benchmarks	#benchmarks	H1	H2	H3
LAO	286	2%	27%	56%
MEDIABENCH	1168	0%	19%	41%
SPEC2000-CINT	2297	1%	24%	63%
SPEC2000-CFP	293	0%	15%	48%
SPEC2006	2704	0%	29%	67%

TABLE 8.1 – SIRALINA vs. autres Heuristiques (Registre de type BR) - % d'instances où SIRALINA est strictement meilleure

Famille de benchmarks	#benchmarks	H1	H2	H3
LAO	286	63%	53%	44%
MEDIABENCH	1168	81%	69%	59%
SPEC2000-CINT	2297	69%	55%	37%
SPEC2000-CFP	293	78%	71%	52%
SPEC2006	2704	68%	49%	33%

TABLE 8.2 – SIRALINA vs. autres Heuristiques(Registre de type BR) - % d'instances où SIRALINA produit le même résultat

Famillle de benchmarks	#benchmarks	H1	H2	H3
LAO	286	49%	68%	86%
MEDIABENCH	1168	34%	67%	87%
SPEC2000-CINT	2297	37%	74%	90%
SPEC2000-CFP	293	31%	55%	82%
SPEC2006	2704	36%	80%	92%

TABLE 8.3 – SIRALINA vs. autres Heuristiques (Registres GR) - % d'instances où SIRALINA est strictement meilleure

Famille de benchmarks	#benchmarks	H1	H2	H3
LAO	286	33%	19%	14%
MEDIABENCH	1168	54%	28%	13%
SPEC2000-CINT	2297	50%	22%	10%
SPEC2000-CFP	293	54%	35%	18%
SPEC2006	2704	52%	16%	8%

TABLE 8.4 – SIRALINA vs. autres Heuristiques(Registres GR) - % d'instances où SIRALINA produit le même résultat

Pour chaque groupe d'instances, nous calculons le pourcentage de déviation moyenne (en nombre de registres requis) entre SIRALINA et les autres heuristiques x ($x = 1, 2$ ou 3). Ce pourcentage de déviation moyenne est calculé ainsi :

$$\frac{\sum_{\text{DDG}}(SR(x) - SR(SIRALINA))}{\sum_{\text{DDG}} SR(x)} \times 100\%$$

où $SR(x) = \sum \mu_{i,j}$ correspond au nombre de registres requis pour la période minimale ($p = MEP$) calculé avec la méthode x. Ce pourcentage permet de quantifier l'amélioration

obtenue avec SIRALINA. Les résultats sont fournis dans le tableau 8.6 pour les registres de type BR et les registres de type GR. Nous pouvons faire les observartions suivantes :

— Pour le type GR, SIRALINA surpasse toutes les autres méthodes. La méthode 1 qui résout le programme linéaire entier (limité à 10 secondes) est incapable de fournir des solutions meilleures que celles de SIRALINA. Ce résultat est attendu, étant donné que le nombre moyen de tâches de type GR dépasse 12.

— Pour le type BR, SIRALINA est plus performante que les heuristiques 2 et 3 mais moins bonne que l'heuristique 1. Ce résultat s'explique par le nombre réduit de tâches BR qui permet au solveur de fournir une solution exacte dans le temps imparti.

Famille de benchmarks	H1	H2	H3
LAO	-25%	+8%	+65%
MEDIABENCH	-17%	+23%	+44%
SPEC2000-CINT	-25%	+23%	+49%
SPEC2000-CFP	-20%	+24%	+47%
SPEC2006	-27%	+26%	+52%

TABLE 8.5 – Pourcentage de déviation moyenne entre SIRALINA et les autres méthodes (Registre de type BR)

Famille de benchmarks	H1	H2	H3
LAO	+27%	+47%	+75%
MEDIABENCH	+13%	+35%	+65%
SPEC2000-CINT	+7%	+33%	+62%
SPEC2000-CFP	+7%	+29%	+61%
SPEC2006	+8%	+34%	+63%

TABLE 8.6 – Pourcentage de déviation moyenne entre SIRALINA et les autres méthodes (Registre de type GR)

Dans cette partie, nous avons étudié le problème de minimisation du nombre de registres pour une période fixée $p = MEP$. Dans la partie suivante, nous suivons une démarche différente, nous fixons un nombre limite de registres (donné par le fabricant) et nous étudions l'accroissement minimale de période nécessaire pour utiliser le nombre de registres à disposition.

8.3.3/ ÉTUDE DE LA PERTE DU PARALLÉLISME DE TÂCHES

Le problème étudié ici est de trouver la plus petite période p pour laquelle le nombre de registres requis $\sum \mu_{i,j}$ sera inférieur ou égal au nombre de registres R effectivement disponibles. On cherche si une solution existe avec $p = MEP$ et on fait croître p si nécessaire jusqu'à obtenir $\sum \mu_{i,j} \leq R$. L'augmentation de la période p conduit à une perte de parallélisme de tâches que nous analysons dans cette partie. Nous distinguons trois cas :

— Cas 1 ($\sum \mu_{i,j} > R$) : la demande de stockage est strictement supérieure au nombre de registres disponibles quelque soit la période p ($MEP \leq p \leq L$). On dit dans cas là que le problème de stockage n'a pas de solution.

— Cas 2 ($\sum \mu_{i,j} \leq R$) : la demande de stockage est plus petite ou égale au nombre de registres disponibles pour la période minimale ($p = MEP$).

— Cas 3 : La demande de stockage est plus grande que le nombre de registres disponibles pour la période ($p = MEP$), mais il existe une période p' (la plus petite possible mais supérieure à p) pour laquelle la demande de stockage est plus petite que le nombre de registres disponibles. Dans ce cas, nous exprimons la perte de parallélisme par le pourcentage de déviation : $\frac{p'-MEP}{MEP} \times 100\%$.

Comme exemple concret, prenons le nombre de registres dans le processeur VLIW ST231 de STMicroelectronics. Le nombre de registres disponibles de type GR est égal à 61, celui de type BR à 8. Les tableaux 8.7 et 8.8 indiquent le nombre de benchmarks dans chaque famille qui se trouvent dans le cas 1 pour les registres de type BR ou GR. Le nombre de cas sans solution est rare avec SIRALINA, ce qui signifie qu'une architecture de processeur avec 61 registres GR et 8 registres BR semble être suffisante pour éviter de recourir à la mémoire centrale. En effet, dans la plupart des cas, l'augmentation de la période permet de n'utiliser que le nombre de registres à disposition.

Famille de benchmarks	SIRALINA	H1	H2	H3
LAO	2	5	11	24
MEDIABENCH	0	1	2	29
SPEC2000-CINT	0	0	0	4
SPEC2000-CFP	0	0	0	2
SPEC2006	0	0	0	25

TABLE 8.7 – Nombre de problèmes sans solution (8 registres de type BR disponibles)

Nous examinons plus en détails la perte de parallélisme dans le cas 3. Nous observons une perte avec SIRALINA pour seulement 5 instances sur 6748 avec les registres de type BR. (La perte de parallélisme est égale à 5.6%, 544%, 526%, 255%, 255%). Pour ces cinq instances, les méthodes H1 et H2 fournissent un nombre de registres inférieur au nombre disponible pour la période minimale. La perte de parallélisme avec SIRALINA peut paraître énorme pour ces 5 instances, mais rappelons que les heuristiques H1 et H2 s'avèrent efficaces pour traiter un nombre réduit de tâches de type BR. Pour les registres de type GR, nous identifions seulement trois instances pour lesquelles une perte de parallélisme est observée. Les résultats obtenus sur ces trois instances sont détaillés dans le tableau 8.9. La perte de parallélisme est indiquée entre parenthèses. Ce tableau montre que lorsque SIRALINA est capable de trouver une solution avec une perte de parallélisme (cas 3), les autres méthodes ne fournissent pas de solution (cas 1). Et mieux encore, pour la troisième instance, SIRALINA trouve une solution sans perte de parallélisme (cas 2) alors que H1 conduit à une perte de parallélisme et les méthodes H2 et H3 ne donnent pas de solution (cas 1).

Famille de benchmarks	SIRALINA	H1	H2	H3
LAO	4	2	3	69
MEDIABENCH	0	0	3	34
SPEC2000-CINT	1	0	3	62
SPEC2000-CFP	0	0	0	8
SPEC2006	0	0	10	95

TABLE 8.8 – Nombre de problèmes sans solution (61 registres de type GR disponibles)

Nom de l'instance	SIRALINA	M1	M5	M3
LAO-polysyn	Cas 3 (642%)	Pas sol	Pas sol	Pas sol
MEDIABENCH-gsm-long-term	Cas 3 (162%)	Pas sol	Pas sol	Pas sol
MEDIABENCH-ghostscript	Cas 2	Cas 3 (5.6%)	Pas sol	Pas sol

TABLE 8.9 – Perte de parallélisme (61 registres de type GR disponibles)

8.4/ CONCLUSION

Le calcul d'une allocation optimale de stockage périodique est intraitable dans la pratique (problème NP-complet). Nous avons proposé une heuristique en deux étapes nommée SIRALINA. Une première étape calcule les variables de planification et permet de calculer les distances de réutilisation potentielles si l'arc de réutilisation correspondant est ajouté. Ensuite, une deuxième étape résout un problème d'affectation linéaire en utilisant la méthode hongroise afin de sélectionner les arcs de réutilisation appropriés. Cette heuristique améliore grandement notre capacité à résoudre le problème pour de grandes instances. L'amélioration provient essentiellement du fait que SIRALINA commence par calculer les valeurs minimales μ avant de fixer les arcs de réutilisation, tandis que les méthodes précédentes procédaient dans le sens contraire.

SIRALINA a été mise en œuvre dans le compilateur industriel de STmicroelectronics pour la génération et l'optimisation de code embarqué. Des expériences pratiques sur des benchmarks de référence (SPEC2000, Mediabench, LAO, SPEC2006) montrent que SIRALINA fournit des solutions satisfaisantes avec un temps de résolution rapide (moins d'une seconde). Dans presque tous les cas, SIRALINA réussit à limiter le nombre de registres utilisés par rapport au nombre de registres disponibles dans le processeur embarqué ST231. Et ceci sans aucune perte de parallélisme, c'est à dire que la période calculée dans presque tous les DDGs est égal à la période minimale $p = MEP$. Dans certains cas critiques, SIRALINA conduit à une perte de parallélisme alors que les autres méthodes ne fournissent pas de solution.

Voici quelques pistes de travail qui avaient été envisagées au cours de notre étude :

— Étudier la structure particulière des contraintes de modèle exact afin d'envisager l'application d'une relaxation lagrangienne.
— Considérer des contraintes supplémentaires sur le graphe de réutilisation pour tenir compte de la particularité de certains processeurs. (Par exemple, pour un processeur à registres rotatifs, le circuit de réutilisation doit être hamiltonien)

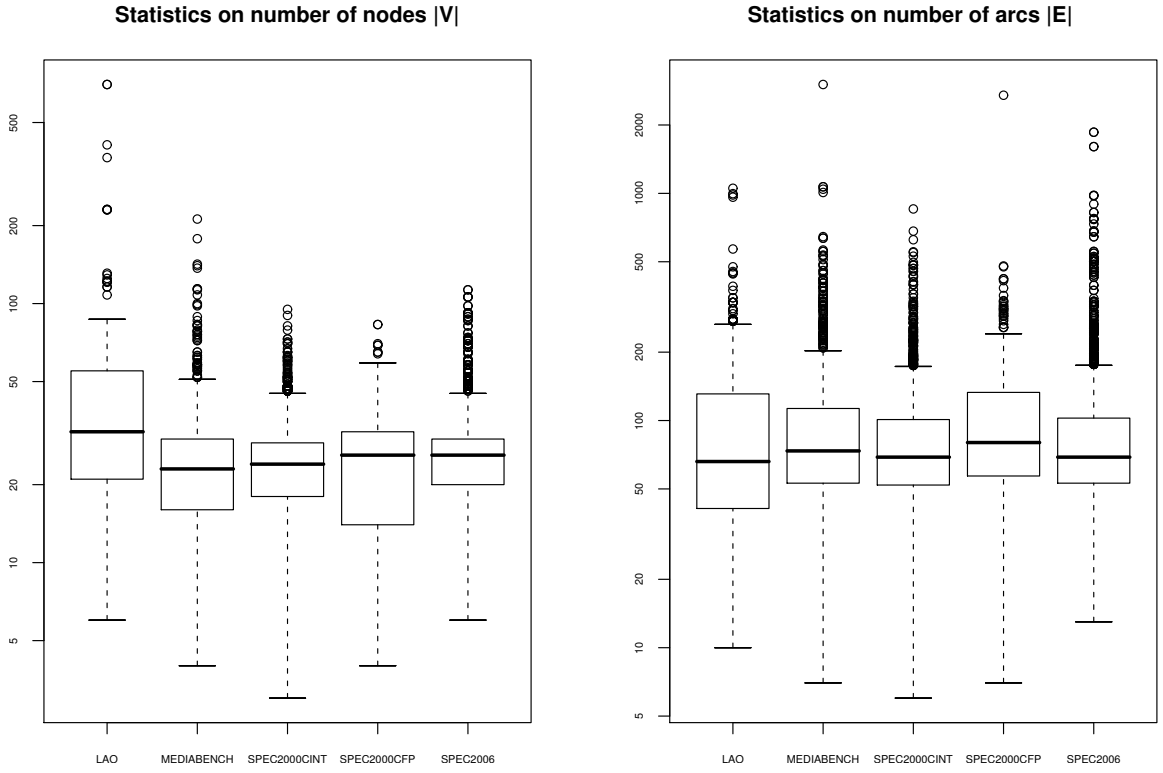


FIGURE 8.1 – Taille des DDG

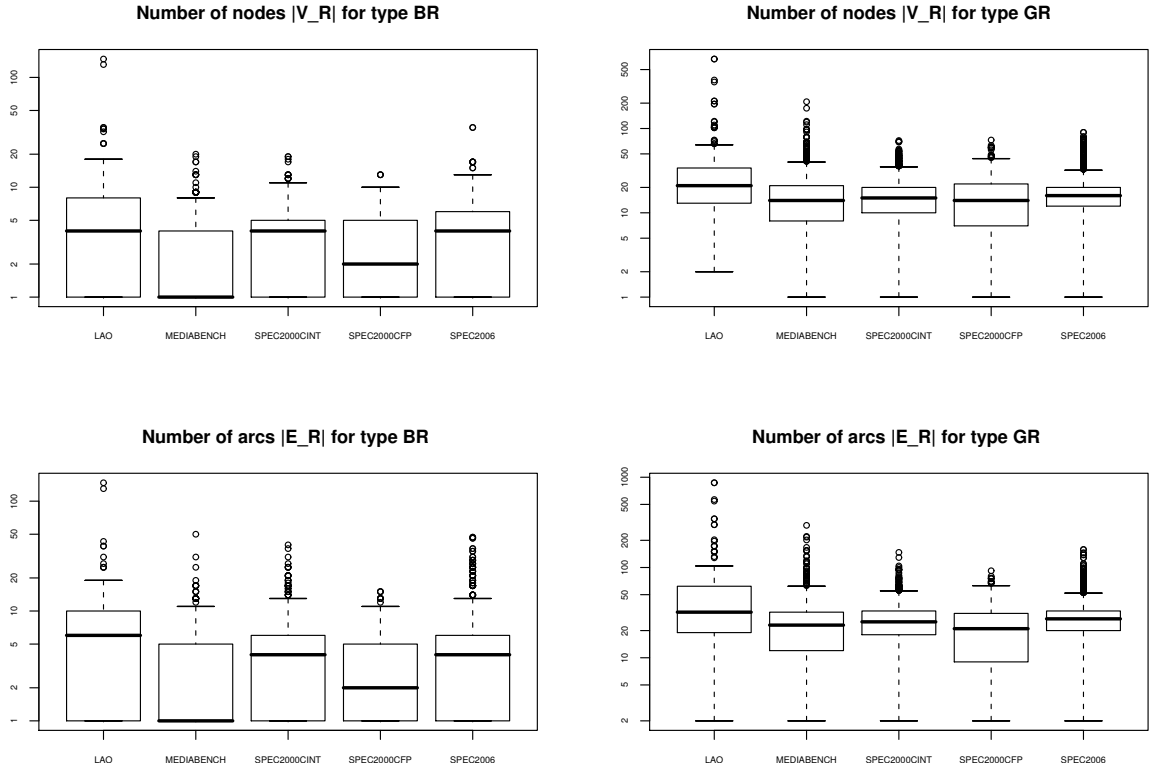


FIGURE 8.2 – $|V^R|$ et $|E^R|$ pour les registres de type BR et GR

9

HEURISTIQUE DE GÉNÉRATION PARALLÈLE D'ENSEMBLES COUVRANTS DISJOINTS

9.1/ PRÉALABLES

Dans cette étude, nous considérons un ensemble S de capteurs déployés dans une zone d'intérêt et devant surveiller un ensemble T de cibles fixées. Nous utilisons les notations proposées dans le chapitre 7. Nous supposons que les capteurs sont capables de se localiser et de déterminer les cibles qu'ils sont en mesure de couvrir. Ainsi pour chaque capteur j, l'ensemble des cibles couvertes par ce capteur est connue et notée $T(j)$. Inversement, l'ensemble des capteurs couvrant une cible i est défini et noté $S(i)$. Un capteur s couvre une cible t, si la distance euclidienne $d(t, s)$ est inférieure à R_s, le rayon de couverture du capteur. La figure 9.1 représente 4 capteurs s_1, s_2, s_3, s_4 de même rayon de couverture, et t_1, t_2, t_3 sont les cibles couvertes par ces capteurs. $T(1) = \{t_1\}$, $T(2) = \{t_2\}$, $T(3) = \{t_1, t_3\}$ et $T(4) = \{t_2, t_3\}$. De la même manière, nous avons : $S(1) = \{s_1, s_3\}$, $S(2) = \{s_2, s_4\}$, $S(3) = \{s_3, s_4\}$.

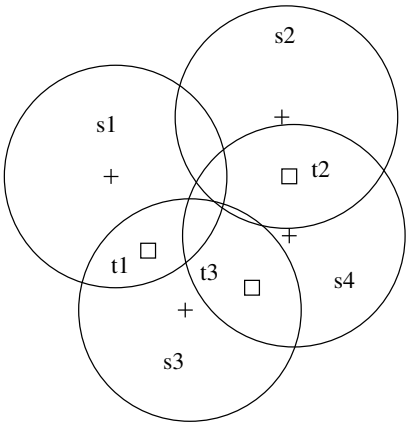


FIGURE 9.1 – Capteurs et cibles

Nous supposons que les durées de vie initiales de chaque capteur sont identiques. On entend par durée de vie d'un capteur la période de temps durant laquelle il peut rester actif (jusqu'à épuisement de sa batterie). Pour augmenter la durée de vie du réseau,

on répartit l'ensemble des capteurs dans des ensembles disjoints, appelés ensembles couvrants, où chaque ensemble couvrant C_k est capable de surveiller l'ensemble des cibles sous surveillance. Chaque ensemble couvrant (ensemble de capteurs) est activé successivement tandis que les autres capteurs restent en sommeil. Les calculs pour le groupement et l'activation des capteurs sont réalisés de manière centralisée par une station de base qui diffuse l'information aux différents capteurs. Formellement, notre problème consiste à trouver un nombre maximal K d'ensembles couvrants tels que :

$$C_k \subseteq S, \forall k = 1..K$$

$$C_k \cap C_l = \emptyset, \forall k, l : 1 \leq k, l \leq K \text{ et } k \neq l$$

et

$$\forall i \in T, \forall k = 1..K, C_k \cap S(i) \neq \emptyset$$

Pour notre application comportant $|S| = n$ capteurs et $|T| = m$ cibles à surveiller, le nombre maximal d'ensembles couvrants disjoints pourrait être, dans le meilleur des cas, égal à :

$$K = \min_{i=1..m} |S(i)|$$

car chaque ensemble couvrant doit surveiller toutes les cibles et qu'un capteur peut faire partie d'un seul ensemble couvrant (rappels de la partie 7.3.1.2).

9.2/ MÉTHODE DE RÉSOLUTION

9.2.1/ PRINCIPES

L'algorithme 1 de construction des ensembles couvrants disjoints est donné en pseudo-code dans la partie 9.4. Au début de l'algorithme, on considère un nombre K d'ensembles couvrants notés C_k (pour $k = 1..K$). Chaque ensemble est initialement vide (ne contient aucun capteur) et chaque itération de l'algorithme consiste à ajouter, parmi les capteurs non encore affectés (ensemble $\overline{S}$ de capteurs disponibles), un capteur dans chaque ensemble couvrant de manière à couvrir un maximum de cibles.

Pour ce faire, nous calculons à chaque itération et pour chaque cible i non encore couverte dans tous les ensembles couvrants ($\overline{T}$ désigne l'ensemble des cibles non encore couvertes par tous les ensembles couvrants) un taux de criticité $R(i)$, que nous définissons comme le rapport entre le nombre de capteurs non encore affectés (disponibles) à un ensemble couvrant, capables de surveiller la cible i, et le nombre d'ensembles couvrants ne réalisant pas encore la couverture de la cible i. Nous désignons par $NS(i)$ l'ensemble des capteurs non encore affectés et capables de surveiller la cible i. Nous désignons par $NC(i)$ l'ensemble des ensembles couvrants ne réalisant pas la surveillance de la cible i.

$$R(i) = \frac{|NS(i)|}{|NC(i)|} \tag{9.1}$$

Nous choisissons parmi l'ensemble des cibles $\overline{T}$ non encore couvertes par tous les ensembles couvrants, la cible la plus critique, c'est-à-dire avec le taux de criticité le plus faible. En cas d'égalité de taux entre plusieurs cibles, le choix est fait aléatoirement. Une fois la cible critique choisie, les capteurs capables de surveiller cette cible doivent être

répartis dans chaque ensemble couvrant. Cette répartition peut être totalement aléatoire ou faite de manière judicieuse pour faire diminuer à chaque itération le nombre de cibles non encore couvertes par tous les ensembles couvrants.

Pour réaliser cette répartition des capteurs dans chaque ensemble couvrant, nous associons un coût p_{jk} d'affectation du capteur j à l'ensemble couvrant C_k et qui représente le nombre supplémentaire de cibles que l'ensemble couvrant C_k pourra couvrir si le capteur j lui est affecté. On désigne par $T(C_k)$ l'ensemble des cibles couvertes par l'ensemble couvrant C_k. Ainsi $\overline{T} \setminus T(C_k)$ représente l'ensemble des cibles non encore couvertes par l'ensemble couvrant C_k. $T(j)$ correspond aux cibles couvertes par le capteur j. Le nombre supplémentaire de cibles couvertes par l'ajout du capteur j dans l'ensemble couvrant C_k est donc égal à $|T(j) \setminus T(C_k)|$.

Nous sommes face à un problème classique d'affectation linéaire (voir l'annexe B) qui consiste à placer exactement un capteur dans chaque ensemble couvrant de telle manière à ce que le coût total d'affectation soit maximisé. Dans notre cas, le nombre de capteurs à répartir peut être différent du nombre d'ensembles couvrants. Si le nombre de capteurs est plus grand que le nombre d'ensembles couvrants, nous créons artificiellement des ensembles couvrants supplémentaires avec des coûts nuls d'affectation entre les capteurs et ces ensembles. Si inversement, le nombre de capteurs à placer est inférieur au nombre d'ensembles couvrants, nous ajoutons des capteurs fictifs avec des coûts d'affectation nuls. Au cours de la résolution, si nous rencontrons cette dernière situation, cela signifie que certains ensembles couvrants ne permettront pas de surveiller l'ensemble des cibles du domaine à surveiller. Nous dirons que ce sont des ensembles couvrants incomplets.
La phase finale de l'algorithme consiste à comptabiliser le nombre total d'ensembles couvrants permettant de couvrir l'ensemble des cibles. Parmi tous les ensembles construits avec la procédure décrite précédemment, certains sont dits incomplets, ils ne permettent pas de couvrir toutes les cibles. Un ensemble couvrant est dit incomplet si le nombre de cibles qu'il couvre est inférieur à m (nombre total de cibles), autrement dit, si $|T(C_k)| \neq |T|$. A la fin de l'algorithme, ils ne seront pas comptabilisés car ils ne vérifient pas la contrainte que nous nous sommes fixée, garantissant que chaque ensemble couvrant surveille toutes les cibles du domaine. Le nombre d'ensembles couvrants disjoints obtenus avec notre heuristique est noté K_{heur}.

9.2.2/ EXEMPLE

Pour illustrer notre méthode, nous présentons un exemple simple avec seulement 20 capteurs et 10 cibles. Le tableau 9.1 donne la liste des capteurs surveillant chacune des cibles. Ici le nombre maximal d'ensembles couvrants possibles est égal à $K = 4$. La cible la plus critique est la cible 1 (en effet, $R(1) = \frac{4}{4} = 1$). Nous affectons chaque capteur j de $S(1)$ à chaque ensemble couvrant ($k = 1..4$) comme présenté dans le tableau 9.2.

TABLE 9.1 – Ensemble des capteurs surveillant chaque cible

cible i	Capteurs dans $S(i)$
t_1	s_1, s_3, s_4, s_5
t_2	$s_6, s_8, s_{13}, s_{14}, s_{20}$
t_3	$s_2, s_5, s_9, s_{10}, s_{12}$
t_4	$s_1, s_6, s_7, s_8, s_{13}, s_{15}$
t_5	$s_4, s_{17}, s_{18}, s_{19}, s_{20}$
t_6	$s_2, s_5, s_{11}, s_{14}, s_{16}$
t_7	$s_8, s_9, s_{10}, s_{17}, s_{18}$
t_8	$s_6, s_7, s_{10}, s_{14}, s_{20}$
t_9	s_1, s_2, s_3, s_4, s_5
t_{10}	$s_2, s_4, s_8, s_{12}, s_{14}, s_{17}, s_{19}$

TABLE 9.2 – Capteurs et cibles couvertes pour chaque ensemble couvrant après la première itération

k	C_k	$T(C_k)$
1	s_1	$\{t_1, t_4, t_9\}$
2	s_3	$\{t_1, t_9\}$
3	s_4	$\{t_1, t_5, t_9, t_{10}\}$
4	s_5	$\{t_1, t_3, t_6, t_9\}$

Nous faisons une mise à jour des ensembles $\overline{T} \leftarrow \overline{T} \setminus \{t_1, t_9\}$, car les cibles t_1 et t_9 sont couvertes par tous les ensembles. $\overline{S} \leftarrow \overline{S} \setminus \{s_1, s_3, s_4, s_5\}$. Puis nous obtenons les nouveaux ensembles $NS(i)$ et $NC(i)$ pour chaque cible i et le taux de criticité correspondant (voir tableau 9.3).

TABLE 9.3 – Après la première itération : capteurs disponibles pour surveiller la cible i, ensembles couvrants ne surveillant pas la cible i, taux de criticité $R(i)$

Cible $i \in \overline{T}$	$NS(i)$	$NC(i)$	$R(i)$
t_2	$s_6, s_8, s_{13}, s_{14}, s_{20}$	$\{C_1, C_2, C_3, C_4\}$	5/4
t_3	s_2, s_9, s_{10}, s_{12}	$\{C_1, C_2, C_3\}$	4/3
t_4	$s_6, s_7, s_8, s_{13}, s_{15}$	$\{C_2, C_3, C_4\}$	5/3
t_5	$s_{17}, s_{18}, s_{19}, s_{20}$	$\{C_1, C_2, C_4\}$	4/3
t_6	$s_2, s_{11}, s_{14}, s_{16}$	$\{C_1, C_2, C_3\}$	4/3
t_7	$s_8, s_9, s_{10}, s_{17}, s_{18}$	$\{C_1, C_2, C_3, C_4\}$	5/4
t_8	$s_6, s_7, s_{10}, s_{14}, s_{20}$	$\{C_1, C_2, C_3, C_4\}$	5/4
t_{10}	$s_2, s_8, s_{12}, s_{14}, s_{17}, s_{19}$	$\{C_1, C_2, C_4\}$	6/3

Les cibles les plus critiques sont t_2, t_7 et t_8 d'après le tableau 9.3. Nous choisissons aléatoirement la cible t_7. Nous devons répartir des capteurs de $NS(7) = \{s_8, s_9, s_{10}, s_{17}, s_{18}\}$

dans chaque ensemble couvrant k de $NC(7) = \{C_1, C_2, C_3, C_4\}$.
Nous calculons $p_{jk} = |T(j) \setminus T(C_k)|$ pour chaque capteur $j \in NS(7)$ et pour chaque ensemble couvrant $C_k \in NC(7)$. p_{jk} représente le nombre supplémentaire de cibles que l'ensemble couvrant C_k pourra surveiller si le capteur j lui est affecté. Les coûts d'affectation linéaire sont résumés dans le tableau 9.4.

TABLE 9.4 – Matrice des coûts d'affectation

	C_1	C_2	C_3	C_4
s_8	3	**4**	3	4
s_9	2	2	**2**	1
s_{10}	**3**	3	3	2
s_{17}	3	3	1	**3**
s_{18}	1	1	1	1

Une solution optimale du problème d'affectation linéaire consiste à affecter les capteurs s_{10}, s_8, s_9, s_{17} respectivement aux ensembles couvrants C_1, C_2, C_3, C_4. Remarquons qu'il y a plusieurs solutions optimales possibles pour ce problème d'affectation. Une autre solution est de placer s_{17} dans C_1, s_{10} dans C_2, s_9 dans C_3, et s_8 dans C_4. Nous faisons une mise à jour des ensembles $\overline{T} \leftarrow \overline{T} \setminus \{t_7\}$, $\overline{S} \leftarrow \overline{S} \setminus \{s_{10}, s_8, s_9, s_{17}\}$. Puis nous recommençons le processus avec les informations fournies dans les tableaux 9.5 et 9.6, et ainsi de suite. A la fin de l'algorithme, nous avons traité l'ensemble des cibles et réparti les capteurs dans les différents ensembles couvrants. Seuls les ensembles couvrants complets (couvrant toutes les cibles) seront comptabilisés.

TABLE 9.5 – Capteurs et cibles couvertes dans chaque ensemble couvrant après 2 itérations

k	C_k	$T(C_k)$
1	$\{s_1, s_{10}\}$	$\{t_1, t_3, t_4, t_7, t_8, t_9\}$
2	$\{s_3, s_8\}$	$\{t_1, t_2, t_4, t_7, t_9, t_{10}\}$
3	$\{s_4, s_9\}$	$\{t_1, t_3, t_5, t_7, t_9, t_{10}\}$
4	$\{s_5, s_{17}\}$	$\{t_1, t_3, t_5, t_6, t_7, t_9, t_{10}\}$

9.3/ OBTENTION D'UNE SOLUTION OPTIMALE

Pour mesurer la qualité de la solution trouvée K_{heur} par notre heuristique, nous cherchons une manière de construire un nombre optimal K_{opt} d'ensembles couvrants disjoints. Nous savons que ce nombre est borné par $K = \min_{1..m} |S(i)|$. Nous savons également que le nombre d'ensembles disjoints K_{heur} obtenus avec notre heuristique constitue une borne inférieure.

TABLE 9.6 – Après deux itérations : capteurs disponibles, ensembles couvrants ne surveillant pas la cible, et taux de criticité

Cible $i \in \overline{T}$	$NS(i)$	$NC(i)$	$R(i)$
t_2	$s_6, s_{13}, s_{14}, s_{20}$	$\{C_1, C_3, C_4\}$	4/3
t_3	s_2, s_{12}	$\{C_2\}$	2/1
t_4	s_6, s_7, s_{13}, s_{15}	$\{C_3, C_4\}$	4/2
t_5	s_{18}, s_{19}, s_{20}	$\{C_1, C_2\}$	3/2
t_6	$s_2, s_{11}, s_{14}, s_{16}$	$\{C_1, C_2, C_3\}$	4/3
t_8	s_6, s_7, s_{14}, s_{20}	$\{C_2, C_3, C_4\}$	4/3
t_{10}	$s_2, s_{12}, s_{14}, s_{19}$	$\{C_1\}$	4/1

Nous avons donc un encadrement de la valeur optimale :

$$K_{heur} \leq K_{opt} \leq K \tag{9.2}$$

Pour trouver la valeur optimale K_{opt}, nous effectuons une recherche dichotomique entre K_{heur} et K. A chaque étape de cette recherche dichotomique, nous vérifions s'il existe une partition possible des capteurs en K' ensembles couvrants. Pour cela, nous résolvons le problème 7.2 de satisfaction de contraintes formulé dans la partie 7.3.1.2 et redonné ici :

$$\begin{cases} \sum_{j \in S(i)} \mathbf{x_{j,k}} \geq 1, & \forall i \in T, \forall k = 1..K' \\ \sum_{k=1}^{k=K'} \mathbf{x_{j,k}} \leq 1, & \forall j \in S \\ \mathbf{x_{j,k}} \in \{0, 1\} \end{cases} \tag{9.3}$$

Les variables de décision $\mathbf{x_{j,k}}$ sont des variables binaires égales à 1 si le capteur j appartient à l'ensemble couvrant C_k et égales à 0 dans le cas contraire. Le premier type de contrainte $\sum_{j \in S(i)} \mathbf{x_{j,k}} \geq 1, \forall i \in T, \forall k = 1..K'$ garantit que chaque ensemble couvrant surveille toutes les cibles. Le second groupe de contraintes indique que chaque capteur est présent au maximum dans un seul ensemble couvrant. Si ce problème n'a pas de solution réalisable pour une valeur K' donnée, nous devons recommencer avec une valeur de K' inférieure. L'algorithme 2 fourni dans la partie 9.4 présente la recherche dichotomique permettant de trouver la valeur optimale K_{opt} d'ensembles couvrants disjoints pouvant être formés. A chaque itération de la recherche dichotomique, on résout le problème 9.3 de satisfaction de contraintes. Nous avons intérêt à limiter le nombre d'appels à la résolution d'un tel problème, car celle-ci s'avère très coûteuse en temps de calcul. Plus la borne inférieure obtenue avec l'algorithme 1 sera bonne, plus vite la solution optimale sera trouvée.

9.4/ ALGORITHMES DE RÉSOLUTION

Cette partie donne les deux algorithmes qui permettent de générer le nombre optimal d'ensembles couvrants disjoints. L'algorithme 1 construit heuristiquement en parallèle des ensembles couvrants disjoints. Connaissant une borne inférieure et une borne supérieure, l'algorithme 2 par recherche dichotomique et par résolution d'un problème de satisfaction de contraintes fournit un nombre optimal d'ensembles couvrants.

9.5/ RÉSULTATS EXPÉRIMENTAUX

Dans cette partie, nous évaluons l'efficacité de nos algorithmes au moyen de simulations. Nous simulons un réseau de capteurs et de cibles déployés sur une surface de $500m^2$. Nous supposons que le rayon de couverture des capteurs est égal à $150m$. Dans les différents scénarios, on fait varier le nombre de capteurs de 50 à 200 avec un incrément de 50 et le nombre de cibles de 30 à 120 avec un incrément de 30. Les conditions suivantes sont satisfaites : chaque capteur recouvre au moins une cible et chaque cible est couverte par au moins un capteur. La connectivité du réseau est assurée et tous les capteurs sont capables de communiquer avec la station de base. Pour un nombre donné de capteurs et de cibles, nous générons 100 topologies différentes de réseau. Nous avons réalisé nos calculs sur une station de travail Linux avec un processeur AMD Athlon 64 X2 Dual Core 4000+ de 2.1 GHz.

9.5.1/ NOMBRE D'ENSEMBLES COUVRANTS DISJOINTS

Nous mesurons le nombre moyen d'ensembles couvrants K_{heur} et K_{opt} sur les 100 instances par scénario. Comme l'algorithme 1 intègre une part d'aléatoire (dans le choix de cible critique en cas d'égalité), il est exécuté 50 fois pour chaque instance. Le tableau 10.2 montre que notre algorithme est très efficace car il est capable de fournir une solution approchée K_{heur} égale à la solution optimale dans la majorité des cas. Nous avions également implémenté un algorithme de construction aléatoire des ensembles couvrants [Deschinkel and Hakem, 2013] et celui-ci fournissait des résultats éloignés de plus de 38% de l'optimum ($\frac{\overline{K_{opt}}-\overline{K_r}}{\overline{K_{opt}}}$ où K_r est le nombre d'ensembles couvrants construits aléatoirement) pour un nombre de capteurs égal à 200 et un nombre de cibles égal à 120.

9.5.2/ COMPARAISON DES TEMPS D'EXÉCUTION

Nous comparons les temps CPU d'exécution des 3 méthodes (aléatoire, heuristique, exacte). Le tableau 9.8 donne les moyennes de temps (en secondes) pour les 16 scénarios. La méthode dite "exacte" consiste à effectuer une recherche dichotomique entre K_{min} and K et à résoudre le problème de satisfaction de contrainte à chaque pas. Nous distinguons deux cas de figure pour le temps d'exécution de la méthode exacte. Dans le premier cas, la recherche dichotomique commence en considérant le nombre d'ensembles générés par notre heuristique ($K_{min} = K_{heur}$). Dans le second cas, K_{min} correspond au nombre obtenu par l'algorithme de génération aléatoire. Nous observons que le temps de calcul augmente logiquement avec le nombre de capteurs et de cibles. Notre algorithme 1 permet d'atteindre dans la plupart des cas le nombre optimal d'ensembles couvrants si bien qu'il n'est pas nécessaire de recourir à d'autres calculs (résolution du problème de satisfaction de contraintes), les temps d'exécution sont tous quasi nuls pour la méthode exacte dans ce cas (voir colonne 5). En revanche, ils peuvent dépasser 9 minutes quand la borne inférieure n'est pas bonne (voir colonne 6).

TABLE 9.7 – Nombre d'ensembles couvrants disjoints obtenus : aléatoirement K_r, heuristiquement K_{heur} et à l'optimum K_{opt}

N	M	K_r	K_{heur}	K_{opt}
50	30	2.97	3.46	3.46
	60	2.44	2.96	2.96
	90	2.19	2.60	2.60
	120	2.06	2.49	2.49
100	30	7.16	8.84	8.84
	60	5.54	7.54	7.54
	90	4.97	6.99	6.99
	120	4.65	6.70	6.70
150	30	10.75	13.50	13.50
	60	8.80	12.03	12.03
	90	7.89	11.34	11.34
	120	7.42	10.91	10.91
200	30	14.53	19.03	19.03
	60	11.65	16.94	16.98
	90	10.44	16.11	16.11
	120	9.67	15.53	15.59

9.6/ CONCLUSION

Nous nous sommes intéressés au problème de maximisation de la durée de vie d'un réseau de capteurs. Nous avons proposé une heuristique centralisée qui construit conjointement plusieurs ensembles couvrants disjoints. Notre algorithme inclut la résolution d'un problème d'affectation linéaire dans le but de couvrir un maximum de cibles dans chaque ensemble à chaque itération. Les simulations ont montré l'excellent comportement de notre algorithme qui produit des solutions optimales dans la plupart des scénarios avec une complexité en $O(max(n,m)^4)$. La qualité de la borne inférieure obtenue permet de réduire significativement la recherche du nombre optimal d'ensembles couvrants, car il faut très peu recourir à la résolution du problème de satisfaction de contraintes.

TABLE 9.8 – Temps d'exécution (en secondes) pour les différentes méthodes

N	M	Aléatoire	Heuristique	Exacte ($K_{min} = K_{heur}$)	Exacte ($K_{min} = K_r$)
50	30	0.010	0.023	0.000	0.026
	60	0.019	0.031	0.000	0.053
	90	0.026	0.037	0.000	0.058
	120	0.036	0.046	0.000	0.030
100	30	0.026	0.145	0.000	1.445
	60	0.056	0.176	0.000	2.443
	90	0.086	0.205	0.000	3.540
	120	0.118	0.227	0.000	4.070
150	30	0.041	0.503	0.000	17.478
	60	0.100	0.653	0.000	25.733
	90	0.160	0.698	0.000	42.646
	120	0.222	0.768	0.000	58.182
200	30	0.063	1.270	0.000	123.600
	60	0.150	1.660	16.190	251.467
	90	0.250	1.730	0.000	345.877
	120	0.352	1.936	83.26	543.568

Algorithme 1 Algorithme de génération d'ensembles couvrants disjoints

Entrée: Un ensemble de cibles T et de capteurs S
Sortie: Des ensembles couvrants disjoints $C_1, .., C_{K_{heur}}$

{INITIALISATION}
Pour tous ensemble couvrant $k = 1..K$ **faire**
{Chaque ensemble couvrant est initialement vide}
$C_k \leftarrow \emptyset$
{ L'ensemble des cibles couvertes par chaque ensemble couvrant est vide}
$T(C_k) \leftarrow \emptyset$
Fin Pour
{L'ensemble des capteurs disponibles contient tous les capteurs déployés dans la zone}
$\overline{S} \leftarrow S$
{L'ensemble des cibles non couvertes contient toutes les cibles de la zone}
$\overline{T} \leftarrow T$
Pour tous cibles $i = 1..m$ **faire**
{Tous les ensembles couvrants ne couvrent pas la cible i}
$NC(i) \leftarrow \{C_1, .., C_K\}$
{Tous les capteurs couvrant i peuvent être affectés}
$NS(i) \leftarrow S(i)$
Fin Pour
{PHASE ITERATIVE}
{Tant qu'il existe une cible non encore couverte dans un des ensembles couvrants et non encore "traitée", et qu'il reste des capteurs disponibles}
Tant que $\overline{T} \neq \emptyset$ et $\overline{S} \neq \emptyset$ **faire**
Pour tous cibles $i \in \overline{T}$ **faire**
Calculer $R(i) = \frac{|NS(i)|}{|NC(i)|}$
Fin Pour
{Choisir la cible la plus critique}
$i^* = \min_{i \in \overline{T}} R(i)$
Pour tous $j \in NS(i^*)$ **faire**
Calculer $p_{jk} = |T(j) \setminus T(C_k)\}|$ pour chaque $C_k \in NC(i^*)$
Fin Pour
Résoudre le problème d'affectation linéaire pour affecter un capteur $j \in NS(i^*)$ dans chaque ensemble couvrant $k \in NC(i^*)$ avec les coûts p_{jk}
Mettre à jour $C_k, T(C_k) \quad \forall k = 1..K$
Mettre à jour $\overline{T}, \overline{S}$
Mettre à jour $NS(i), NC(i) \quad \forall i \in \overline{T}$
Fin Tant que
{PHASE FINALE}
$K_{heur} \leftarrow K$
{Certains ensembles couvrants ne surveillent pas toutes les cibles. Il ne faut pas comptabiliser ces ensembles couvrants "incomplets"}
Pour tous ensemble couvrant C_k **faire**
Si $|T(C_k)| \neq T|$ **Alors**
$K_{heur} \leftarrow K_{heur} - 1$
Fin Si
Fin Pour

Algorithme 2 Recherche dichotomique

Entrée: Un ensemble de cibles T, un ensemble de capteurs S
Sortie: Des ensembles couvrants $C_1, C_2, ..., C_{K_{opt}}$

$K_{min} \leftarrow K_{heur}$
$K_{opt} \leftarrow K_{min}$
Tant que $K_{min} != K_{max}$ **faire**
  $K \leftarrow \lceil \frac{K_{min}+K_{max}}{2} \rceil + 1$
  Résoudre le problème (9.3)
  Si le problème (9.3) n'a pas de solution **Alors**
    $K_{max} \leftarrow K - 1$
  Sinon
    $K_{min} \leftarrow K$
    $K_{opt} \leftarrow K$
    Solution y^*_{jk} de (9.3) sauvegardée
  Fin Si
Fin Tant que
Pour tous $k = 1..K_{opt}$ **faire**
  $C_k \leftarrow \cup_{(s_j/y^*_{jk}=1)}\{s_j\}$
Fin Pour

10

MÉTHODE DE GÉNÉRATION DE COLONNES POUR CONSTRUCTION D'ENSEMBLES COUVRANTS NON DISJOINTS

10.1/ DESCRIPTION DU PROBLÈME

10.1.1/ FORMULATION

Le problème de surveillance de l'ensemble des cibles par des ensembles de capteurs disjoints peut se formuler sous forme d'un programme linéaire. Les variables sont les suivantes : t_u est la durée d'activation de l'ensemble couvrant u, cela signifie que tous les capteurs dans l'ensemble couvrant u seront activés durant une période de temps t_u. On note U l'ensemble des ensembles couvrants *élémentaires*. Le problème s'écrit ainsi :

$$\begin{cases} \max \sum_{u \in U} \mathbf{t_u} \\ \text{sous :} \\ \sum_{u \in U} a_{ju} \mathbf{t_u} \leq E_j, \quad \forall j \in S \\ \mathbf{t_u} \in \mathbb{R}^+ \end{cases} \tag{10.1}$$

La fonction objectif maximise la durée totale d'activation des ensembles couvrants. Chaque contrainte impose une durée d'activation limitée pour chaque capteur. L'indice binaire a_{ju} vaut 1 si le capteur j appartient à l'ensemble couvrant u et 0 sinon. Un ensemble couvrant *élémentaire* correspond à un ensemble de capteurs pour lesquels toutes les cibles sont couvertes et la suppression de l'un ou l'autre de ces capteurs ne permet plus de couvrir toutes les cibles. Le nombre d'ensembles couvrants élémentaires croit rapidement avec la taille du problème (nombre de capteurs et de cibles).

10.1.2/ EXEMPLE

Pour illustrer notre problème, nous présentons un exemple simple avec seulement 10 capteurs et 4 cibles. Le tableau 10.1 donne la couverture des cibles par chaque capteur. On se limite pour l'instant à deux ensembles couvrants $C_0 = \{0, 3, 9\}$ et $C_1 = \{0, 4, 8\}$.

Cible	Capteurs
0	3,4,9
1	4,6,9
2	0,2
3	1,3,5,8

TABLE 10.1 – Couverture des cibles par les capteurs

Le programme linéaire qui correspond à cet exemple simple est :

$$\begin{cases} \max & t_0 + t_1 & & \\ \text{sous :} & & & \\ & t_0 + t_1 & \leq 1.00 & \text{(capteur 0)} \\ & t_0 & \leq 1.00 & \text{(capteur 3)} \\ & t_1 & \leq 1.00 & \text{(capteur 4)} \\ & t_1 & \leq 1.00 & \text{(capteur 8)} \\ & t_0 & \leq 1.00 & \text{(capteur 9)} \\ t_0, t_1 \in \mathbb{R}^+ & & & \end{cases} \tag{10.2}$$

t_0 et t_1 sont respectivement les durées de vie des ensembles couvrants C_0 et C_1. La partie droite de chaque inégalité correspond à la durée de vie maximale de chaque capteur, arbitrairement fixée à 1 (1 unité de temps) pour l'exemple. Dans cet exemple, nous n'avons pas énumérer tous les ensembles couvrants. Nous résolvons le programme linéaire avec seulement deux ensembles couvrants et la durée de vie du réseau obtenue est égale à 1 avec $t_0^* = 0.5$ et $t_1^* = 0.5$. Cela signifie que les capteurs de l'ensemble couvrant C_0 sont activés durant 0.5 unités de temps et les capteurs de l'ensemble couvrant C_1 également. Les capteurs $(1, 2, 5, 6, 7)$ n'apparaissent pas dans le PL car ils ne font partie d'aucun ensemble couvrant généré. Seul le capteur 0 a consommé toute son énergie. Si nous avions généré plus d'ensembles couvrants, nous aurions atteint une durée de vie du réseau égale à 2. Par exemple, en ajoutant les ensembles couvrant $C_2 = (2, 3, 4)$ et $C_3 = (1, 2, 9)$, la planification optimale est obtenue pour $t_0^* = t_1^* = t_2^* = t_3^* = 0.5$.

10.1.3/ MÉTHODE DE GÉNÉRATION DE COLONNES

Comme l'ensemble U des ensembles couvrants élémentaires peut croître rapidement avec le nombre de capteurs et de cibles, nous proposons d'utiliser une méthode de génération de colonnes [Dantzig and Wolfe, 1960a] (voir Annexe A) pour résoudre (10.1). Cela signifie que nous résolvons un problème maître restreint (PMR) avec seulement un sous-ensemble $U' \subseteq U$ d'ensembles élémentaires couvrants et nous enrichirons ce sous-ensemble U' au fur et à mesure. Pour un sous-ensemble $U' \subseteq U$ donné et les multiplicateurs duaux $\pi_j \equiv \pi_j(U')$ associés aux capteurs j, le problème auxiliaire (AUX) consiste à chercher un ensemble couvrant $u \in U \setminus U'$ avantageux (dit "attractif"), c'est à dire avec le coût réduit r_u le plus grand.

$$r_u = (1 - \sum_{j \in S} a_{ju} \pi_j)$$

Si $r_u > 0$ alors l'ensemble couvrant u est intéressant et il est ajouté à la formulation du problème restreint, sinon le problème (10.1) est résolu à l'optimalité.

Le problème auxiliaire revient à chercher un ensemble couvrant u qui maximise $r = (1 - \sum_{j \in u} \pi_j)$, ou qui minimise $\sum_{j \in u} \pi_j$. Si cette somme est inférieure à 1, alors l'ensemble couvrant u est ajouté au problème restreint. Nous formulons le problème AUX comme un programme linéaire en nombre entiers avec des variables binaires y_j pour chaque capteur j qui valent 1 si le capteur j fait partie de l'ensemble couvrant u, et 0 sinon. Les contraintes de garantie de couverture de chaque cible s'écrivent ainsi :

$$\sum_{j \in S(i)} y_j \geq 1 \quad \forall i \in T$$

Le problème AUX est formulé ainsi :

$$\begin{cases} \min \sum_{j \in S} \pi_j y_j \\ \text{sous :} \\ \sum_{j \in S(i)} y_j \geq 1 \quad \forall i \in T \\ y_j \in \{0,1\} \quad \forall j \in S \end{cases} \tag{10.3}$$

La formulation du problème AUX correspond à celle du problème classique de couverture par ensembles [Caprara et al., 1998]. Le problème PMR est formulé comme un PL (10.1) où l'ensemble U d'ensembles couvrants élémentaires doit être remplacé par le sous-ensemble U' qui contient initialement un plus petit nombre d'ensemble couvrants. Le problème PMR (10.1) et le problème AUX (10.3) sont alors résolus successivement et le sous-ensemble U' grossit jusqu'à ce qu'il ne soit plus possible de générer de nouveaux ensembles couvrants avantageux. La solution optimale, c'est-à-dire la planification d'ensembles couvrants qui maximise la durée de vie du réseau, est alors trouvée. Le problème PMR, par sa formulation en PL, peut être résolu en temps polynomial $O(n^3 m^3)$ avec l'algorithme proposé par Ye [Ye, 1991]. En revanche, la résolution du problème AUX (classé dans les problèmes NP-hard [Garey and Johnson, 1990]) peut prendre un temps de calcul inacceptable. Notre idée est d'accélérer la génération d'ensembles couvrants attractifs au moyen d'une heuristique adaptée. Pour mesurer l'efficacité de notre approche, nous avons développé trois méthodes, appelées respectivement, Méthode Exacte (E) , Méthode Heuristique (H) et Méthode Mixte (M). La méthode (E) consiste à résoudre chaque fois le problème AUX jusqu'à l'optimalité à l'aide d'un solveur d'IP dédié. Pour la méthode (H), nous verrons par la suite comment nous utilisons les multiplicateurs duaux pour générer des ensembles couvrants attractifs. Dans la méthode (M), si l'heuristique proposée ne permet pas de générer un ensemble couvrant attractif, le problème AUX est résolu à l'optimalité avec le solveur. Tandis que les méthodes (E) et (M) fournissent des solutions optimales, la méthode (H) se contente d'une solution approchée. L'approche par génération de colonnes proposée en trois versions est expliquée plus en détails dans la partie suivante.

10.2/ MÉTHODE DE RÉSOLUTION

L'approche par génération de colonnes nécessite de construire un sous-ensemble initial U' d'ensembles couvrants élémentaires. Le nombre et le type d'ensembles couvrants générés initialement n'aura pas d'impact sur la solution optimale finale. Nous choisissons de construire un ensemble couvrant élémentaire en procédant en deux étapes. Dans une première étape, un ensemble couvrant est généré, puis la deuxième étape élimine

les capteurs superflus (qui peuvent être supprimés de l'ensemble tout en garantissant la couverture complète des cibles).

10.2.1/ GÉNÉRATION D'UN ENSEMBLE COUVRANT ÉLÉMENTAIRE

L'algorithme 3 génère un ensemble couvrant, qui permet de couvrir toutes les cibles, mais qui n'est pas nécessairement élémentaire. Certains capteurs peuvent être de trop (il est possible de supprimer un capteur sans perdre en qualité de couverture). La vérification de capteurs superflus est réalisée par l'algorithme 4. Cet algorithme sera appliqué pour chaque ensemble couvrant généré quelque soit son mode de génération (généré initialement ou généré avec une heuristique de recherche d'ensemble couvrant attractif)

Algorithme 3 Cover_Set_Generation(u)

Entrée: Un ensemble de cibles T, un ensemble de capteurs S
Sortie: Un ensemble couvrant u

$V \leftarrow T$
$u \leftarrow \emptyset$
Tant que V n'est pas $\emptyset$ **faire**
  Sélectionner une cible $i \in V$ aléatoirement
  $V \leftarrow V \setminus \{i\}$
  Sélectionner aléatoirement un capteur $j \in S(i)$ qui couvre la cible i
  $u \leftarrow u \cup \{j\}$
  Pour tous cibles $h \in T(j)$ **faire**
    $V \leftarrow V \setminus \{h\}$
  Fin Pour
Fin Tant que

10.2.2/ GÉNÉRATION D'UN ENSEMBLE COUVRANT ÉLÉMENTAIRE ATTRACTIF

Une fois les ensembles couvrants élémentaires générés pour former un ensemble initial, le processus de génération de colonnes consiste à introduire des nouveaux ensembles couvrants (nouvelles colonnes) attractifs dans le problème PMR. Cette tâche est réalisée par la résolution exacte du problème AUX (voir l'algorithme 5) ou en utilisant une heuristique (algorithmes 6 ou 7). Nous avons développé deux heuristiques qui reposent sur le même principe, celui de produire des ensembles couvrants en ajoutant des capteurs les moins coûteux possibles (au sens des multiplicateurs duaux π_j) de manière à ce que le coût réduit résultant soit le plus grand possible.

Dans l'heuristique 1, nous sélectionnons d'abord une cible puis nous choisissons un capteur de coût minimal pour couvrir cette cible. Le processus est répété jusqu'à couvrir toutes les cibles. L'algorithme 6 de complexité $O(mn)$ correspond à celui de l'heuristique 1. Dans l'heuristique 2, le choix se fait d'abord sur le capteur de coût minimal. S'il y a plusieurs capteurs de même coût, le choix est fait aléatoirement parmi eux. Puis nous regardons les cibles couvertes par ce capteur et nous répétons le processus jusqu'à la couverture de toutes les cibles. L'algorithme 7 correspond à cette deuxième heuristique. Comme ces algorithmes intègrent une part d'aléatoire, ils doivent être appliqués plusieurs

Algorithme 4 Check_Elementary_Cover_Set(u)

Entrée: Un ensemble couvrant u
Sortie: Un ensemble couvrant élémentaire u

```
w ← u
Tant que w n'est pas ∅ faire
    Sélectionner un capteur j ∈ w aléatoirement
    Vérifier si la suppression du capteur j est possible (toutes les cibles restent couvertes)
    Si oui Alors
        u ← u \ {j}
    Fin Si
    w ← w \ {j}
Fin Tant que
```

fois (pas plus qu'un certain nombre d'itérations notées $Nb\_Max\_Iterations$) jusqu'à ce qu'un ensemble couvrant au coût réduit positif soit trouvé. Notons que ces deux méthodes de génération ne produisent pas nécessairement des ensembles couvrants élémentaires et qu'il faut bien appliquer l'algorithme 4 pour éliminer les capteurs superflus.

10.2.3/ PROCESSUS GLOBAL

L'algorithme 8 présente la méthode de résolution s'appuyant sur le principe de génération de colonnes et comprenant les 3 versions (E, H1 et H2, et M).

10.3/ RÉSULTATS EXPÉRIMENTAUX

Nous avons implémenté les trois versions de l'algorithme 8 présentées dans la partie 10.2.3. Nous avons réalisé nos calculs sur une station de travail Linux avec un processeur AMD Athlon 64 X2 Dual Core 4000+ de 2.1 GHz. Les résolutions des programmes LP et IP sont respectivement portés par la méthode du simplexe et la méthode Branch-and-Bound dans le solveur GLPK (GNU linear Programming Kit) [Mahkorin, 2010]).
Dans cette partie, nous évaluons les performances de nos algorithmes au moyen de simulations. Nous simulons un réseau de capteurs et des cibles déployés aléatoirement sur une aire carrée de $500m^2$. Nous supposons que les capteurs ont tous le même rayon de couverture égal à $150m$. Dans les différents scénarios, nous faisons varier le nombre de capteurs déployés n entre 50 et 200 avec un incrément de 50. Le nombre de cibles m varient entre 30 et 120 avec un incrément de 30. Chaque capteur a une durée de vie égale à 1. Les conditions suivantes sont satisfaites : chaque capteur recouvre au moins une cible et chaque cible est couverte par au moins un capteur. La connectivité du réseau est assurée et tous les capteurs sont capables de communiquer avec la station de base. Nous mesurons la durée de vie du réseau, les temps d'exécution des 3 versions. Pour chaque scénario, les résultats sont des moyennes sur 10 instances (nous avons généré 10 topologies différentes pour chaque scénario). Dans les algorithmes, nous avons fixé Nb_Max_Ite à 10. Le nombre d'ensembles couvrants générés initialement est égal à 10 ($N = 10$).

Algorithme 5 Generation_Attractive_CoverSet_Exact(π,u,r)

Entrée: Multiplicateurs duaux π_j pour chaque capteur $j \in S$
Sortie: Ensemble couvrant u et coût réduit associé r

```
u ← ∅
r ← 1
Résoudre le problème (10.3) avec les multiplicateurs duaux π_j
y*_j, j ∈ S sont les valeurs optimales
Pour tous j ∈ S faire
  Si y*_j = 1 Alors
    {Le capteur j est activé}
    u ← u ∪ {j}
    r ← r − π_j
  Fin Si
Fin Pour
```

10.3.1/ COMPARAISON DES TEMPS D'EXÉCUTION

Nous comparons et commentons les temps d'éxecution des différentes versions. Le tableau 10.2 donne la durée de vie optimale du réseau et les temps de résolution en secondes des trois versions (en fait, pour la version heuristique, il y a 2 résultats, pour l'heuristique 1 et pour l'heuristique 2) pour les 16 scénarios. Les résultats sont cohérents avec ceux observés dans la littérature : la durée de vie du réseau et le temps d'exécution augmentent avec la densité de capteurs, la durée de vie diminue avec le nombre de cibles pour un nombre fixe de capteurs car ces derniers sont plus sollicités. On observe que la méthode mixte peut être jusqu'à 6 fois plus rapide que la méthode exacte. Et les temps de calcul avec les deux heuristiques sont nettement moins élevés que les deux autres versions E et M dès que le nombre de cibles dépasse 60. L'heuristique H1 a un temps de calcul légèrement plus faible que l'heuristique 2. La méthode M fait appel en moyenne 1.83 fois à la résolution exacte du problème AUX , ce qui est très peu mais suffisamment pour ralentir son temps d'exécution.

10.3.2/ COMPARAISONS DE DURÉE DE VIE

Nous comparons la valeur optimale de durée de vie obtenue avec la méthode exacte E avec celles obtenues par les deux heuristiques H1 et H2. Et nous concluons que H1 est vraiment efficace car cette méthode fournit la valeur optimale dans tous les cas simulés avec des temps de calcul réduit. H2 trouve la valeur optimale à l'exception de deux cas sur les 160 et la différence est inférieure à 1%.
Nous avons utilisé les mêmes instances que celles utilisées dans la génération d'ensembles couvrants disjoints du chapitre 9. Comme nous avons arbitrairement choisi des durées de vie de capteur toutes égales à 1, nous pouvons comparer le nombre moyen d'ensembles couvrants disjoints générés dans le chapitre 9 à la durée de vie moyenne du réseau obtenue avec des ensembles couvrants non disjoints. Contrairement à ce qui été suggéré par l'exemple dans [Cardei et al., 2005] et [Berman et al., 2004b] , l'usage d'ensembles couvrants non disjoints n'accroît pas systématiquement la durée de vie du réseau. Sur l'ensemble des instances testées, nous obtenons qu'en moyenne, la durée de vie du réseau est augmentée de $2,5\%$ avec des ensembles couvrants non disjoints. Toutefois, pour certains scénarios (par exemple 100 capteurs, 60 cibles), la durée de vie

Algorithme 6 Generation_Attractive_CoverSet_Heuristic-1(π,u,r)

Entrée: Multiplicateurs duaux π_j pour chaque capteur $j \in S$
Sortie: ensemble couvrant généré u et coût réduit associé r

$V \leftarrow T$
$u \leftarrow \emptyset$
$r \leftarrow 1$
Tant que V is not $\emptyset$ **faire**
  Sélectionner une cible $i \in V$ aléatoirement
  Sélectionner un capteur $j \in S(i)$ de coût minimal (π_j)
  $V \leftarrow V \setminus \{i\}$
  $u \leftarrow u \cup \{j\}$
  $r \leftarrow r - \pi_j$
  Pour tous cibles $h \in T(j)$ **faire**
    $V \leftarrow V \setminus \{h\}$
  Fin Pour
Fin Tant que

dans le cas non disjoint est inférieure de $4,7\%$ au cas disjoint.

10.3.3/ Comparaison du nombre d'ensembles couvrants générés

Le tableau 10.3 donne le nombre d'ensembles couvrants générés et nous constatons que ce nombre est plus élevé avec les heuristiques. Nous pouvions nous attendre à un tel résultat car la méthode exacte génère toujours, à chaque itération, l'ensemble couvrant le plus attractif, si bien que la durée maximale est obtenue en moins d'itérations (moins d'appels à la résolution du problème AUX). Néanmoins la méthode Heuristique (H1 ou H2) reste très compétitive car elle produit une bonne solution en peu de temps. Et il peut être parfois intéressant d'avoir un grand nombre d'ensembles couvrants de manière à ce que les capteurs oscillent fréquemment entre période d'activité et d'inactivité comme cela est suggéré dans [Slijepcevic and Potkonjak, 2001].

10.4/ Conclusion

L'efficacité énergétique est cruciale dans un réseau de capteurs où la durée de vie de la batterie est limitée. Nous avons étudié la possibilité de prolonger la vie du réseau en organisant les capteurs dans des ensembles couvrants non-disjoints qui fonctionnent successivement afin de surveiller toutes les cibles. Nous avons formulé ce problème avec un programme linéaire où les variables sont les temps d'activation des ensembles couvrants et nous avons proposé d'utiliser la technique de génération de colonnes pour le résoudre. Au lieu de résoudre le problème auxiliaire à l'optimalité pour générer des ensembles couvrants aux coût réduits positifs, nous avons conçu une heuristique efficace (version H1). Les résultats des simulations montrent les performances de l'heuristique qui obtient de très bonnes solutions très rapidement.
Notre approche par génération de colonnes a été reprise très récemment dans l'étude de [Carrabs et al., 2015b] où les auteurs proposent de résoudre le problème AUX avec un algorithme génétique. Ils utilisent nos jeux de données pour mesurer l'efficacité de leur

Algorithme 7 Generation_Attractive_CoverSet_Heuristic-2(π,u,r)

Entrée: Multiplicateurs duaux π_j pour chaque capteur $j \in S$
Sortie: Ensemble couvrant généré u et coût réduit associé r

```
V ← T
u ← ∅
r ← 1
Tant que V n'est pas ∅ faire
    Sélectionner un capteur j ∈ S de coût minimal (π_j)
    u ← u ∪ {j}
    r ← r − π_j
    Pour tous cibles h ∈ T(j) faire
        V ← V \ {h}
    Fin Pour
Fin Tant que
```

méthode. Celle-ci est exécutée sur une machine différente de la nôtre et le solveur utilisé est IBM ILOG CPLEX 12.5. Ce dernier offre des performances reconnues en PL et bien supérieures à celles du solveur GLPK que nous avons utilisé. Il n'en reste pas moins que les gains obtenus avec l'algorithme génétique sont significatifs.

			Temps d'exécution			
n	m	Durée de vie	Heuristique 1	Heuristique 2	Mixte	Exacte
50	30	3.8	0.30	0.35	0.12	0.25
	60	3.0	0.53	0.73	0.52	1.03
	90	2.8	0.82	1.42	1.55	2.95
	120	2.7	1.20	2.14	4.03	8.40
100	30	8.7	2.97	3.6	1.03	3.29
	60	7.2	4.25	5.56	8.41	26.53
	90	6.9	6.82	10.48	74.19	243.95
	120	6.7	9.70	15.3	220.64	749.46
150	30	14.7	14.51	17.40	4.94	17.17
	60	12.3	22.21	28.29	48.96	315.66
	90	11.8	30.61	45.16	525.21	2365.65
	120	11.3	48.15	72.28	1987.04	9249.81
200	30	19.6	34.85	40.35	9.50	38.80
	60	17.3	56.34	74.45	126.39	750.40
	90	16.6	132.46	162.51	1297.82	8229.53
	120	15.5	105.87	145.83	4393.04	28942.49

TABLE 10.2 – Durée de vie et Temps d'exécution (en secondes) entre les 3 versions

	H1	H2	Mixte	Exacte
Minimum	1.00	1.00	1.00	1.00
Premier quartile	15.00	17.75	16.00	10.00
Médiane	38.00	37.5	34.00	24.50
Troisième quartile	74.50	75	69.50	58.25
Maximum	131.00	135.00	124.00	130.00

TABLE 10.3 – Nombre d'ensembles couvrants générés

Algorithme 8 Méthode de résolution

$U \leftarrow \emptyset$
{Génération de N ensembles couvrants}
Pour $e = 0$ to N **faire**
  Cover_Set_Generation(u)
  Check_Elementary_Cover_Set(u)
  $U \leftarrow U \cup u$
Fin Pour
Restricted_Master_Problem_Resolution(U)
$Stop \leftarrow 0$
Tant que ($Stop = 0$) **faire**
  $r \leftarrow 0$
  {Recherche d'ensemble couvrant attractif (3 versions)}

  Version 1 : Méthode Exacte
  Generation_Attractive_CoverSet_Exact(π,u,r)

  Version 2 : Heuristique (heuristique 1 ou 2)
  $Nb\_Ite \leftarrow 0$
  Tant que (($r <= 0$) and ($Nb\_Ite \leq Nb\_Max\_Ite$)) **faire**
    Generation_Attractive_CoverSet_Heuristic(π,u,r)
    $Nb\_Ite \leftarrow Nb\_Ite + 1$
  Fin Tant que

  Version 3 : Méthode mixte
  $Nb\_Ite \leftarrow 0$
  Tant que (($r <= 0$) and ($Nb\_Ite \leq Nb\_Max\_Ite$)) **faire**
    Generation_Attractive_CoverSet_Heuristic(π,u,r)
    $Nb\_Ite \leftarrow Nb\_Ite + 1$
  Fin Tant que
  Si ($r <= 0$) **Alors**
    Generation_Attractive_CoverSet_Exact(π,u,r)
  Fin Si

  Si ($r <= 0$) **Alors**
    {La méthode n'a pas permis de générer un ensemble couvrant attractif}
    $Stop \leftarrow 1$
  Sinon
    {L'ensemble couvrant attractif est ajouté}
    $U \leftarrow U \cup \{u\}$
    Restricted_Master_Problem_Resolution(U)
  Fin Si
Fin Tant que

III

MODÉLISATION SOUS FORME DE PROBLÈMES DE FLOTS

11

PROBLÈME DE FLOT À COÛT MINIMUM DANS SIRALINA

11.1/ PRÉAMBULE

La première étape de l'heuristique SIRALINA (voir chapitre 8) nécessite la résolution d'un programme linéaire par la méthode du simplexe. Hors il n'est pas toujours possible de disposer ou d'utiliser un solveur de programmes linéaires à cause de contraintes techniques ou de droits de licence logicielle. Aussi, il est important de prévoir des algorithmes appropriés que l'on peut mettre en oeuvre indépendamment de l'utilisation d'un logiciel de solveur linéaire. Cette partie montre comment écrire notre problème d'ordonnancement comme un problème de flot dans un réseau pour lequel on peut utiliser une méthode algorithmique de résolution.

Le réseau considéré ici est le DDG auquel on ajoute les noeuds "tueurs" K_i et les arcs de E^k. Nous notons par E^k l'ensemble des arcs allant des consommateurs aux nœuds "tueurs" :

$$E^k = \{e = (T_j, K_i) | T_j \in Cons(T_i)\}$$

Considérons le problème d'ordonnancement (P) décrit par le système(8.3). Pour écrire le problème dual de (P), renommons les membres droits des inégalités :

$$\begin{array}{ll} a(e) = \delta(e) - p \times \lambda(e), & \forall e = (T_i, T_j) \in E \\ b(e) = \delta_r(T_j) + p \times \lambda(e), & \forall e = (T_j, K_i) \in E^k \end{array}$$

Pour faciliter l'écriture du problème dual (voir Annexe A), nous supposons que les variables $\sigma \in \mathbb{Z}$. La variable σ correspond à une date. Son signe n'est pas important car il suffit de décaler toutes les dates d'un facteur constant pour obtenir des dates positives. Le problème d'ordonnancement s'écrit ainsi :

$$\left\{ \begin{array}{ll} \min & (\sum_{i \in V^R} 1 \times \sigma_{K_i} \\ & - \sum_{j \in V^R} 1 \times \sigma_j \\ & - \sum_{j \in V \setminus V^R} 0 \times \sigma_j) \\ \text{sous :} & \\ \sigma_j - \sigma_i \geq a(e), & \forall e = (T_i, T_j) \in E \\ \sigma_{K_i} - \sigma_j \geq b(e), & \forall e = (T_j, K_i) \in E^k \\ \sigma \in \mathbb{Z} & \end{array} \right. \tag{11.1}$$

On associe à chaque contrainte linéaire du problème primal, une variable duale $f(e) \in \mathbb{N}$ pour $e \in E \cup E^k$. Pour chaque variable primale $\sigma \in \mathbb{Z}$, on obtient une contrainte d'égalité dans le problème dual. Les contraintes sont les suivantes [1] :

— Contraintes duales pour les variables primales σ_{K_i} ($T_i \in V^R$) :

$$\forall T_i \in V^R, \sum_{?\xrightarrow{e\in E^k} K_i} f(e) = 1$$

— Contraintes duales pour les variables primales σ_j ($T_j \in V^R$) :

$$\forall T_j \in V^R, \sum_{?\xrightarrow{e\in E} j} f(e) - \sum_{j\xrightarrow{e\in E} ?} f(e) - \sum_{j\xrightarrow{e\in E^k} ?} f(e) = -1$$

— Contraintes duales pour les variables primales σ_j ($T_j \in V \setminus V^R$) :

$$\forall T_j \in V \setminus V^R, \sum_{?\xrightarrow{e\in E} j} f(e) - \sum_{j\xrightarrow{e\in E} ?} f(e) - \sum_{j\xrightarrow{e\in E^k} ?} f(e) = 0$$

— La fonction objectif duale est :

$$\max \sum_{e\in E} a(e) \times f(e) + \sum_{e\in E^k} b(e) \times f(e)$$

11.2/ FORMULATION DU PROBLÈME DE FLOT À COÛT MINIMUM

Dans cette partie, nous montrons que le problème dual correspond à un problème de flot à coût minimum. Nous allons reprendre chacune des contraintes du problème dual et montrer en quoi elle correspond à une contrainte de flot.

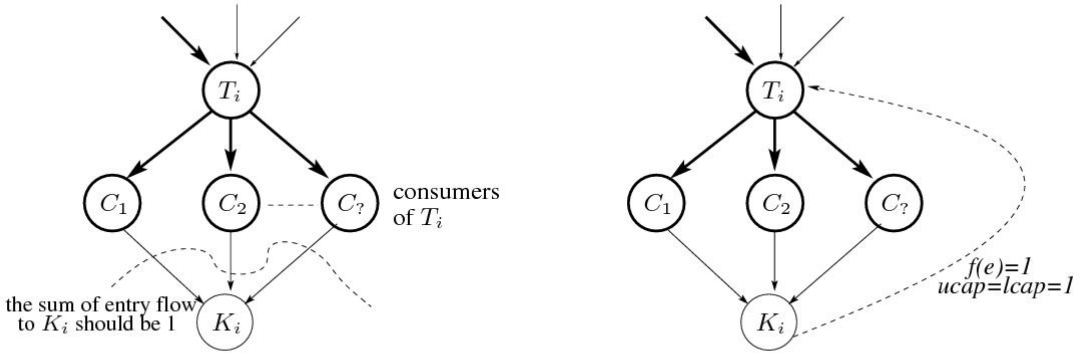
Contraintes duales associées aux variables primales σ_{K_i} ($T_i \in V^R$) :

$$\forall T_i \in V^R, \sum_{?\xrightarrow{e\in E^k} K_i} f(e) = 1$$

Cet ensemble de contraintes indique que le flot entrant au noeud K_i doit être égal à 1. La figure 11.1(a) illustre ce fait. On ajoute des arcs fictifs entre K_i and $T_i \in V^R$ de coût nul, et de capacité minimale et maximale égale à 1 (voir figure 11.1(b)). Les arcs additionnels (K_i, T_i) sont les uniques arcs sortant de chaque noeud K_i, on désigne par E_f l'ensemble de ces arcs. on peut maintenant écrire les équations de conservation de flot au noeud K_i :

$$\forall T_i \in V^R, \sum_{?\xrightarrow{e\in E^k} K_i} f(e) - \sum_{K_i\xrightarrow{e\in E^f} ?} f(e) = 0$$

1. Pour ces contraintes, nous utilisons la notation '?' pour désigner un noeud arbitraire : par exemple $? \xrightarrow{e} j$ désigne un arc e d'extrémité terminale j.



(a) Constraints on entry flow to K_i (b) Adding fictitious arcs to impose the flow to be equal to

FIGURE 11.1 – Contraintes de flot au noeud "tueur"

Contraintes duales pour les variables primales σ_j ($T_j \in V^R$) :

$$\forall T_j \in V^R, \sum_{?\xrightarrow{e\in E} j} f(e) - \sum_{j\xrightarrow{e\in E} ?} f(e) - \sum_{j\xrightarrow{e\in E^k} ?} f(e) = -1$$

$$\sum_{?\xrightarrow{e\in E} j} f(e) + 1 - \sum_{j\xrightarrow{e\in E} ?} f(e) - \sum_{j\xrightarrow{e\in E^k} ?} f(e) = 0$$

Pour le noeud $T_j \in V^R$, nous savons que le flot entre K_j and T_j est égal à 1 (voir le paragraphe précédent, figure 11.1(b)). Nous pouvons donc écrire la contrainte de flot de cette manière :

$$\forall T_j \in V^R, \sum_{?\xrightarrow{e\in E\cup E^f\cup E^k} j} f(e) - \sum_{j\xrightarrow{e\in E\cup E^f\cup E^k} ?} f(e) = 0$$

Contraintes duales pour les variables primales σ_j ($T_j \in V \setminus V^R$) :

$$\forall T_j \in V \setminus V^R, \sum_{?\xrightarrow{e\in E} j} f(e) - \sum_{j\xrightarrow{e\in E} ?} f(e) - \sum_{j\xrightarrow{e\in E^k} ?} f(e) = 0$$

Comme aucun arc de $E^f \cup E^k$ entre dans T_j, et qu'aucun arc de E^k ne quitte T_j pour des tâches $T_j \in V \setminus V^R$, nous pouvons réécrire la contrainte de conservation de flot au noeud T_j :

$$\forall T_j \in V \setminus V^R, \sum_{?\xrightarrow{e\in E\cup E^f\cup E^k} j} f(e) - \sum_{j\xrightarrow{e\in E\cup E^f\cup E^k} ?} f(e) = 0$$

Fonction objectif :
Puisque nous avons ajouté l'ensemble $E^f = \{e = (K_i, T_i) | T_i \in V^R\}$ d'arcs fictifs, la fonction objectif devient :

$$\max \sum_{e\in E} a(e) \times f(e) + \sum_{e\in E^k} b(e) \times f(e) + \sum_{e\in E^f} 0 \times f(e)$$

Pour se ramener à un problème de flot à coût minimum, il suffit de changer le problème de maximisation à un problème de minimisation en inversant les signes :

$$\min \sum_{e \in E} (-a(e)) \times f(e) + \sum_{e \in E^k} (-b(e)) \times f(e) + \sum_{e \in E^f} 0 \times f(e)$$

Nous avons donc montré que le problème dual correspond à la formulation d'un problème de flot à coût minimum (voir Annexe C). Le réseau concerné est le DDG auquel ont été ajouté les noeuds "tueur" K_i, les arcs entrant aux noeuds "tueur" $E^k = \{e = (T_j, K_i) | T_j \in Cons(T_i)\}$ et les arcs fictifs $E^f = \{e = (K_i, T_i) | T_i \in V^R\}$. On désigne par $f(e)$ le flot entier sur chaque arc $e \in E \cup E^k \cup E^f$. Les capacités minimales de tous les arcs sont fixées à zéro en dehors des arcs de E^f. Les capacités maximales sont non bornées sauf pour les arcs de E^f. Pour l'ensemble E^f, les capacités minimales et maximales sont fixées à 1. Le problème de flot à coût minimum peut être résolu par plusieurs algorithmes polynomiaux [Ahuja and ans James B. Orlin, 1993]. Une fois la valeur optimale de flot trouvée, nous pouvons en déduire les valeurs optimales des variables primales.

11.3/ RETOUR AU PROBLÈME D'ORDONNANCEMENT INITIAL

La résolution du problème de flot à coût minimum fournit la valeur optimale du flot $f^*(e), \forall e \in E \cup E^k$ (on ne s'intéresse pas à la valeur du flot sur les arcs E^f qui sera toujours égale à 1). Pour retrouver les valeurs des variables σ du problème primal (problème d'ordonnancement), nous utilisons les relations primales/duales suivantes :

- Si $f^*(e) > 0$ alors la contrainte correspondante dans le problème primal est une contrainte d'égalité.
- Si $f^*(e) = 0$ alors la contrainte correspondante dans le problème primal est une inégalité.

Formellement, on obtient les relations suivantes :

- Si $e = (T_j, T_j) \in E$ alors :
  - si $f^*(e) > 0$ alors $\sigma_j - \sigma_i = \delta(e) - p \times \lambda(e) \Rightarrow \sigma_j = \sigma_i +$ a(e).
  - Si $f^*(e) = 0$ alors $\sigma_j - \sigma_i \geq \delta(e) - p \times \lambda(e)$.
- Si $e = (K_i, T_j) \in E^k$ alors :
  - Si $f^*(e) > 0$ alors $\sigma_{K_i} - \sigma_j = \delta_r(T_j) + p \times \lambda(e) \Rightarrow \sigma_{K_i} = \sigma_j +$ b(e).
  - Si $f^*(e) = 0$ alors $\sigma_{K_i} - \sigma_j \geq \delta_r(T_j) + p \times \lambda(e)$.

L'ensemble de ces équations traduit un problème de potentiel dans le DDG (page 316 de [Ahuja and ans James B. Orlin, 1993]). Ici la fonction potentielle est parfaitement définie à travers la fonction σ. Rappelons que le potentiel d'un sommet du graphe est équivalent à la valeur du plus long chemin entre un noeud source et le reste des noeuds. Le problème de plus long chemin dans un graphe $G = (V, E)$ peut être résolu par l'algorithme de Bellman-Ford de complexité égale à $O(|E| \times |V|)$ [Cormen et al., 1990].

11.4/ A PROPOS DU CHOIX DE L'ALGORITHME DE FLOT DE COÛT MINIMUM

Nous avons implémenté un algorithme de résolution pour le problème de flot de coût minimum. Il existe plusieurs algorithmes polynomiaux de complexités

différentes [Ahuja and ans James B. Orlin, 1993], mais il n'est pas clairement établi lequel d'entre eux est le plus efficace en pratique. Nous avons choisi l'algorithme de "cost-scaling" de Goldberg et Tarjan [Goldberg and Tarjan, 1990] pour sa démonstration expérimentale d'efficacité [Goldberg, 1992] (Il s'exécute en $O(n^3 log(n.C))$, où n est la taille du réseau et C le coût maximal) et sa facilité d'implémentation.
Nous avons comparé les temps d'exécution de l'heuristique SIRALINA dans les deux cas de figure, l'un avec l'utilisation d'un solveur de programme linéaire, l'autre avec la résolution par un algorithme de flot de coût minimum. Nous avons lancé 30 exécutions pour chaque instance sur une même machine (pour 6748 instances) : ces 30 essais sont recommandés par le test de Student [Jain, 1991] pour tenir compte de la variabilité des temps de résolution d'une exécution à une autre pour la même instance. Sur la base de ces 30 essais par instance, nous sommes en mesure d'affirmer avec un niveau de confiance de 90% que le temps de résolution de SIRALINA en utilisant la méthode simplexe est meilleur en moyenne que celui obtenu en mettant en oeuvre l'algorithme de "cost scaling" de recherche de flot à coût minimum. En moyenne, nous avons constaté que dans le premiers cas, SIRALINA est 70% plus rapide. Ces résultats étaient inattendus. Nous espérions obtenir une meilleure performance avec la résolution du problème de flot de coût minimum par la méthode du "cost scaling" (nous avions essayé la méthode du "mean cycle cancelling" [Goldberg and Tarjan, 1989] et les résultats étaient bien pires). Il se peut que le choix de l'algorithme utilisé soit à l'origine de ce résultat et que l'étude de comparaison pourrait être poursuivie en implémentant d'autres algorithmes.

12

PROBLÈME DE COUVERTURE DE CIBLES : UN PROBLÈME DE FLOT ?

Je me suis intéressée à la manière dont le problème de couverture pouvait être abordé en utilisant les modèles et les outils de la théorie des graphes (voir Annexe C). Mon attention a été retenue par le modèle de graphe proposé par Cardei [Cardei and Du, 2005] dont l'article a été cité plus de 1000 fois (source : Google scholar). Les auteurs montrent qu'il est possible de formuler le problème MLP pour des ensembles couvrants disjoints comme un problème de flot maximal dans graphe particulier. En examinant plus précisément la manière dont le graphe est construit, nous avons remarqué que le flot obtenu sur certaines arêtes de ce graphe correspond à la notion de sur-couverture ou de sous-couverture décrite dans la partie 7.3.1 pour le modèle (7.4). Nous avons donc cherché à savoir si le modèle d'optimisation (7.4) pouvait avoir une interprétation en termes de problème de flot. Et nous avons montré que ce dernier peut effectivement s'interpréter comme un problème de flot à coût minimum.

Dans un premier temps, nous allons présenter les différentes étapes de construction du graphe. Puis nous détaillerons les différentes formulations d'optimisation du flot.

12.1/ CONSTRUCTION DU GRAPHE

Nous expliquons la manière de construire le graphe pour lequel un problème de flot maximal est exprimé.

Étape 1

Soit un graphe direct orienté $\tilde{G} = (\tilde{V}, \tilde{E})$ avec l'ensemble des sommets $\tilde{V} = S \cup T$. L'ensemble $S = \{S_1, S_2, ..., S_n\}$ désigne les capteurs et l'ensemble $T = \{T_1, T_2, ..., T_m\}$ les cibles. Il y a un arc $(S_j, T_i) \in \tilde{E}$ si et seulement si T_i est une cible surveillée par le capteur S_j (dit autrement $T_i \in T(j)$). On ajoute également un sommet X et des arcs qui relient chaque sommet cible T_i au sommet X. Tous les arcs du graphe ont une capacité égale à 1. La figure 12.1 donne un exemple de graphe $\tilde{G}$ avec 3 capteurs et 3 cibles : le capteur S_1 couvre les cibles T_1 et T_2, S_2 couvre les cibles T_1, T_2 et T_3, S_3 couvre T_2 et T_3.

Étape 2

Dessiner K copies du graphe $\tilde{G}$, nommées $G_1, G_2, ..., G_K$. Notons aussi que le nombre K fait référence à la borne maximale d'ensembles couvrants disjoints calculée en (7.1). Dans ces K copies, le premier indice dans la dénomination du sommet correspond

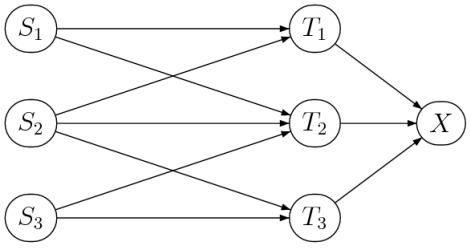


FIGURE 12.1 – Graphe capteurs-cibles $\tilde{G}$

sommet	indice
S_0	noeud source
S_{0j}	$j = 1..n$
S_{kj}	$k = 1..K, j = 1..n$
T_{ki}	$k = 1..K, i = 1..m$
X_k	$k = 1..K$
O	noeud puits de sur-couverture
C	noeud puits de couverture

TABLE 12.1 – Ensemble des sommets du graphe

au numéro de la copie. Par exemple, un sommet $S_j \in \tilde{G}$, est copié K fois, et appelé $S_{1j}, S_{2j}, ..., S_{Kj}$ dans $G_1, G_2, ..., G_K$.

Étape 3

- Créer un noeud source S_0 et pour chaque capteur $S_j \in S$, créer un sommet S_{0j}. Puis connecter la source S_0 aux sommets S_{0j} avec un arc de capacité égale au degré de S_j dans $\tilde{G}$ (ce degré est aussi égal au nombre de cibles couvertes par le capteur j, donc à la cardinalité de l'ensemble $T(j)$).
- Connecter aussi chaque sommet S_{0j} avec les K sommets S_{kj} (pour $1 \leq k \leq K$) et affecter à chaque arc une capacité égale au degré de S_j dans $\tilde{G}$.

Étape 4

- Créer deux noeuds O et C.
- Connecter chaque sommet T_{ki} au sommet O avec une capacité égale au nombre de capteurs $|S|$.
- Connecter chaque sommet X_k $(1 \leq k \leq K)$ au sommet C et affecter à l'arc une capacité égale au nombre de cibles $|T|$.

Nous regroupons dans les tableaux 12.1 et 12.1 les ensembles de sommets et d'arcs qui constituent le graphe $G = (V, E)$.

Le graphe G construit à partir du graphe $\tilde{G}$ de la figure 12.1 est représenté sur la figure 12.2.

arcs	indice	capacité
S_0S_{0j}	$j = 1..n$	$\|T(j)\|$
$S_{0j}S_{kj}$	$j = 1..n, k = 1..K$	$\|T(j)\|$
$S_{kj}T_{ki}$	$k = 1..K, j = 1..n, i \in T(j)$	1
$T_{ki}X_k$	$k = 1..K, i = 1..m$	1
$T_{ki}O$	$k = 1..K, i = 1..m$	n
X_kC	$k = 1..K$	m

TABLE 12.2 – Ensemble des arcs du graphe G

12.2/ FLOT RÉALISABLE

Un flot f **réalisable** dans le graphe $G = (V, E)$ précédemment construit, est défini comme une fonction de E à valeur entière qui satisfait les contraintes suivantes :

— Contrainte de capacité : pour tous les arcs $(u, v) \in E$, $0 \leq f_{uv} \leq c_{uv}$. Comparé au problème de flot classique, il y a une condition supplémentaire : pour tout sommet v, $f_{uv} \in \{0, c_{uv}\}$.

— Contrainte de conservation du flot : $\forall v \in V \setminus \{S_0, O, C\}, \sum_{u:(u,v)\in E} f_{uv} - \sum_{u:(v,u)\in E} f_{vu} = 0$

L'objectif est de maximiser le flot reçu au sommet C.

12.3/ FLOT MAXIMAL

Les auteurs [Cardei and Du, 2005] ont démontré le théorème suivant : Soit un ensemble $S = \{S_1, S_2, ..., S_n\}$ de n capteurs et un ensemble de m cibles $T = \{S_1, S_2, ..., T_m\}$, le problème MPL a K^* ensembles couvrants disjoints si et seulement si le flot maximal réalisable reçu au point C du graphe G est égal à mK^*. Le problème de flot maximal s'écrit comme un programme linéaire mixte en nombres entiers :

$$\begin{cases} \max \sum_{k=1..K} \mathbf{f_{X_kC}} & \\ \text{sous :} & \\ (1) \quad 0 \leq \mathbf{f_{uv}} \leq c_{uv} & \forall (u,v) \in E \\ (2) \quad \sum_{u:(u,v)\in E} \mathbf{f_{uv}} - \sum_{u:(v,u)\in E} \mathbf{f_{vu}} = 0 & \forall v \in V; v \neq \{S_0, O, C\} \\ (3) \quad \mathbf{f_{S_{kj}T_{ki_1}}} = \mathbf{f_{S_{kj}T_{ki_2}}} = ... = \mathbf{f_{S_{kj}T_{ki_j}}} & \forall j \in S, k = 1..K, \text{ et } T(j) = T_{i_1}, T_{i_2}, ..., T_{i_j} \\ (4) \quad \mathbf{f_{T_{k1}X_k}} = \mathbf{f_{T_{k2}X_k}} = ... = \mathbf{f_{T_{km}X_k}} & k = 1..K \end{cases} \tag{12.1}$$

avec :

— $\mathbf{f_{S_{kj}T_{ki_p}}} \in \mathbb{N}$ pour tout $j \in S$, $k = 1..K$ et p tel que $T_{i_p} \in T(j)$

— $\mathbf{f_{T_{ki}X_k}} \in \mathbb{N}$ pour tout $k = 1..K$ et $i \in T$

— toutes les autres variables $\in \mathbb{R}$

Les deux dernières contraintes (3) et (4) assurent que pour tout $v \neq O$, le flot $f_{uv} \in \{0, c_{uv}\}$. Remarquons également que dans chaque composante G_k, tous les flots $f_{T_{kp}X_k}$ ont obligatoirement la même valeur, soit 1 ou 0. Et par conséquent le flot f_{X_kC} a obligatoirement la valeur m ou 0.

Les auteurs [Cardei and Du, 2005] proposent une heuristique pour construire le nombre maximal d'ensembles couvrants. La première étape consiste à résoudre le problème (12.1), le flot maximal calculé en C est égal à f_C (cette valeur est obligatoirement un multiple de m). Le nombre maximal d'ensembles couvrants sera égal

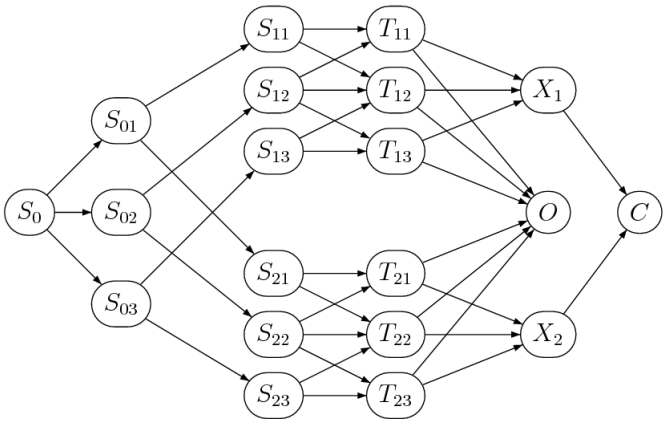


FIGURE 12.2 – Graphe G

à $n_e = \frac{fc}{m}$. Il est aisé de déduire ensuite la composition de chaque ensemble couvrant. C_k est un ensemble couvrant si et seulement si le flot sur l'arc $X_k C$ n'est pas nul. Le capteur j appartient à l'ensemble couvrant C_k si et seulement si le flot sur l'arc $S_{0j}S_{kj}$ n'est pas nul.

Nous rappelons qu'il est possible d'obtenir le nombre optimal d'ensembles couvrants disjoints par la résolution du programme linéaire en nombre entiers (7.3) présenté dans la partie 7.3.1 et redonné ici :

$$\begin{cases} \max \sum_{k=1}^{k=K} \mathbf{z_k} \\ \text{sous :} \\ \sum_{j \in S(i)} \mathbf{x_{j,k}} \geq \mathbf{z_k}, & \forall i \in T, \forall k \in K \\ \sum_{k=1}^{k=K} \mathbf{x_{j,k}} \leq 1, & \forall j \in S \\ \mathbf{x_{j,k}} \in \{0,1\} \\ \mathbf{z_k} \in \{0,1\} \end{cases} \tag{12.2}$$

12.4/ FLOT À COÛT MINIMUM

Le modèle proposé par [Pedraza et al., 2006] permet d'étendre la durée de vie du réseau en divisant l'ensemble des capteurs en un nombre maximal d'ensembles couvrants tels que chaque nouvel ensemble couvrant C_{k+1} couvre le plus de cibles possibles, mais jamais plus que l'ensemble précédent C_k. Pour atteindre cet objectif, les auteurs proposent une formulation sous forme de programme linéaire en nombre entier. Ce modèle (7.4) déjà décrit dans la partie 7.3.1.2 est présenté à nouveau :

$$\begin{cases} \min \sum_{k\in K} \sum_{i\in T} (w_{o,i,k}\mathbf{o_{i,k}} + w_{u,i,k}\mathbf{u_{i,k}}) & \\ \text{sous :} & \\ (1) \quad \sum_{j\in S(i)} \mathbf{x_{j,k}} - \mathbf{o_{i,k}} + \mathbf{u_{i,k}} = 1 & \forall i \in T, \forall k \in K \\ (2) \quad \sum_{k\in K} \mathbf{x_{j,k}} = 1 & \forall j \in S \\ \quad \mathbf{o_{i,k}} \in \mathbb{N}, & \forall i \in T, \forall k \in K \\ \quad \mathbf{u_{i,k}} \in \{0,1\}, & \forall i \in T, \forall k \in K \\ \quad \mathbf{x_{j,k}} \in \{0,1\}, & \forall j \in S, \forall k \in K \end{cases} \tag{12.3}$$

— $\mathbf{x_{j,k}}$: indique si le capteur j est dans l'ensemble couvrant C_k (1 si oui et 0 sinon)
— $\mathbf{o_{i,k}}$: *sur-couverture*, le nombre de capteurs moins un qui surveille la cible i dans l'ensemble couvrant C_k
— $\mathbf{u_{i,k}}$: *sous-couverture*, indique si la cible i est couverte (1 si non et 0 si couverte) dans l'ensemble couvrant C_k

Nous montrons que le problème (12.3) peut être aussi interprété comme un problème de flot de coût minimum dans le graphe G construit dans la partie 12.1. Pour être tout à fait rigoureux, le graphe utilisé n'est pas exactement le graphe G, le graphe considéré doit être constitué d'un nombre K de copies de $\tilde{G}$ supérieur car les ensembles couvrants incomplets sont autorisés (voir partie 7.3.1.2).

Soit un graphe $H = (V', E')$ avec des capacités $c(u, v)$ et des coûts unitaires de transport p_{uv} pour les arcs $(u, v) \in E'$. La production en un sommet source v est égale à d_v et la demande en un sommet puits v est égale à d_v. La formulation générale d'un problème de flot de coût minimum est donnée par :

$$\begin{cases} \min \sum_{(u,v)\in E'} p_{uv} f_{uv} & \\ \text{sous :} & \\ 0 \leq f_{uv} \leq c_{uv} & \forall (u,v) \in E' \\ \sum_{u:(u,v)\in E'} f_{uv} - \sum_{u:(v,u)\in E'} f_{vu} = 0 & \forall v \in V'; v \text{ n'est pas une source, ni un puits} \\ \sum_{u:(u,v)\in E'} f_{uv} - \sum_{u:(v,u)\in E'} f_{vu} = -d_v & v \text{ est une source} \\ \sum_{u:(u,v)\in E'} f_{uv} - \sum_{u:(v,u)\in E'} f_{vu} = d_v & v \text{ est un puits} \end{cases} \tag{12.4}$$

Dans le graphe G, le flot sur l'arc $T_{ki}X_k$, égal à 0 ou 1, indique si la cible T_i est couverte ou non dans l'ensemble couvrant C_k. Il peut donc être mis en relation avec la variable de sous-couverture, nous avons $\mathbf{f_{T_{ki}X_k}} = (1 - \mathbf{u_{i,k}})$. De la même manière le flot $\mathbf{f_{T_{ki}O}}$ représente le nombre de capteurs moins un qui couvrent la cible T_i. Nous pouvons donc l'associer à la variable $\mathbf{o_{i,k}}$ du programme linéaire (12.3) : $\mathbf{f_{T_{ki}O}} = \mathbf{o_{i,k}}$. Le nombre de capteurs couvrant la cible i dans l'ensemble C_k est égal à $\sum_{j\in S(i)} \mathbf{x_{j,k}}$ pour le modèle 12.3. Dans le modèle de flot, cette quantité correspond à la somme des flots entrant au sommet T_{ki} et s'écrit $\sum_{(S_{kj},T_{ki})} \mathbf{f_{S_{kj}T_{ki}}}$ pour chaque ensemble couvrant C_k.
La contrainte de conservation de flot au noeud T_{ki} s'écrit :

$$\left(\sum_{(S_{kj},T_{ki})} \mathbf{f_{S_{kj}T_{ki}}} \right) - \mathbf{f_{T_{ki}O}} - \mathbf{f_{T_{ki}X_k}} = 0$$

D'après ce qui a été écrit précédemment, cette contrainte peut aussi s'écrire :

$$\sum_{j\in S(i)} \mathbf{x_{j,k}} - (1 - \mathbf{u_{i,k}}) - \mathbf{o_{i,k}} = 0$$

Autrement dit :

$$\sum_{j \in S(i)} \mathbf{x_{j,k}} - \mathbf{o_{i,k}} + \mathbf{u_{i,k}} = 1$$

On retrouve la contrainte (1) du problème (12.3). Dans ce problème, on admet que certains ensembles couvrants ne couvrent pas la totalité des cibles, on dit que ces ensembles sont incomplets. Par rapport au problème de flot maximal (12.1) présenté dans la partie 12.3, la contrainte (4) qui impose que toutes les cibles soient couvertes dans un ensemble couvrant peut être relâchée. En revanche, les contraintes (1), (2) et (3) sont conservées. Et il faut ajouter une contrainte indiquant que la valeur du flot à l'entrée de chaque sommet S_0S_{0j} est égal au nombre de cibles couvertes par chaque capteur j. Ces quatre contraintes réunies assurent qu'un capteur appartient à un seul ensemble couvrant. On retrouve les contraintes (2) du problème (12.3). Elles assurent aussi qu'un capteur placé dans un ensemble couvrant couvre toutes les cibles situées sur son disque de couverture.
En choisissant judicieusement les coûts sur les arcs (voir le tableau 12.3), la fonction objectif du problème de flot à coût minimum s'écrit pour le graphe G :

$$\sum_{(T_{ki},X_k) \in E} -w_{u,i,k}\mathbf{f_{T_{ki}X_k}} + \sum_{(T_{ki},O) \in E} w_{o,i,k}\mathbf{f_{T_{ki}O}}$$

Nous pouvons réécrire cette fonction objectif en remplaçant $\mathbf{f_{T_{ki}X_k}}$ par $(1 - \mathbf{u_{i,k}})$ et $\mathbf{f_{T_{ki}O}}$ par $\mathbf{o_{i,k}}$. Nous obtenons :

$$\sum_{(T_{ki},X_k) \in E} -w_{u,i,k}(1 - \mathbf{u_{i,k}}) + \sum_{(T_{ki},O) \in E} w_{o,i,k}\mathbf{o_{i,k}}$$

$$= \sum_{(T_{ki},X_k) \in E} (-w_{u,i,k}) + \sum_{(T_{ki},X_k) \in E} w_{u,i,k}\mathbf{u_{i,k}} + \sum_{(T_{ki},O) \in E} w_{o,i,k}\mathbf{o_{i,k}}$$

$$= \sum_{k \in K}\sum_{i \in T}(-w_{u,i,k}) + \sum_{k \in K}\sum_{i \in T} w_{u,i,k}\mathbf{u_{i,k}} + \sum_{k \in K}\sum_{i \in T} w_{o,i,k}\mathbf{o_{i,k}}$$

La somme $w = \sum_{k \in K}\sum_{i \in T}(-w_{u,i,k})$ est un terme constant. L'objectif à minimiser est :

$$\sum_{k \in K}\sum_{i \in T}(w_{o,i,k}\mathbf{o_{i,k}} + w_{u,i,k}\mathbf{u_{i,k}}) + w$$

Nous retrouvons l'objectif du problème (12.3) à un terme près w. Nous venons de transformer le problème (12.3) en problème de flot à coût minimum dans le graphe G avec la contrainte additionnelle (3). La formulation complète sous forme de problème de flot à coût minimum est la suivante :

$$\begin{cases} \min & \sum_{(T_{ki},X_k) \in E} -w_{u,i,k}\mathbf{f_{T_{ki}X_k}} + \sum_{(T_{ki},O) \in E} w_{o,i,k}\mathbf{f_{T_{ki}O}} & \\ \text{sous :} & & \\ (1) \quad 0 \leq \mathbf{f_{uv}} \leq c_{uv} & & \forall (u,v) \in E \\ (2) \quad \sum_{u:(u,v) \in E} \mathbf{f_{uv}} - \sum_{u:(v,u) \in E} \mathbf{f_{vu}} = 0 & & \forall v \in V; v \neq \{S_0, O, C\} \\ (3) \quad \mathbf{f_{S_{kj}T_{ki_1}}} = \mathbf{f_{S_{kj}T_{ki_2}}} = ... = \mathbf{f_{S_{kj}T_{ki_j}}} & & \forall j \in S, k = 1..K, \text{ et } T(j) = T_{i_1}, T_{i_2}, ..., T_{i_j} \\ (4) \quad \mathbf{f_{S_0S_{0j}}} = |T(j)| & & \forall j \in S \end{cases} \tag{12.5}$$

arcs	index	coût
S_0S_{0j}	$j = 1..n$	0
$S_{0j}S_{kj}$	$j = 1..n, k = 1..K$	0
$S_{kj}T_{ki}$	$k = 1..K, j = 1..n, i \in T(j)$	0
$T_{ki}X_k$	$k = 1..K, i = 1..m$	$-w_{u,i,k}$
$T_{ki}O$	$k = 1..K, i = 1..m$	$w_{o,i,k}$
X_kC	$k = 1..K$	0

TABLE 12.3 – Coûts sur les arcs du graphe $G = (V, E)$

12.5/ COMPARAISON DES MODÈLES

Nous simulons un réseau de capteurs et de cibles déployés sur une surface de $500m^2$. Nous supposons que le rayon de couverture des capteurs est égal à $150m$. Dans les différents scénarios, on fait varier le nombre de capteurs de 10 à 200 et le nombre de cibles de 10 à 120. Pour un nombre donné de capteurs et de cibles (un type de scénario), nous générons 20 topologies différentes de réseau. Nous avons réalisé nos calculs sur une station de travail Linux avec un processeur AMD Athlon 64 X2 Dual Core 4000+ de 2.1 GHz et nous utilisons le solveur GLPK (GNU linear Programming Kit) ([Mahkorin, 2010]) pour la résolution de tous les programmes linéaires.

12.5.1/ COMPARAISON DU MODÈLE DE FLOT MAXIMAL AVEC NOTRE MODÈLE

Nous comparons le nombre d'ensembles couvrants obtenus, d'une part, en résolvant le problème de flot maximal (12.1) proposé par [Cardei and Du, 2005], et d'autre part en résolvant le programme IP (12.2) que nous avons établi. Pour l'ensemble des scénarios testés, nous obtenons les mêmes valeurs. Ce qui différencie les deux modèles est essentiellement le temps de résolution. Le tableau 12.4 donne le rapport R des temps de calcul entre les deux modèles.

$$R = \frac{\text{temps de résolution du modèle (12.1)}}{\text{temps de résolution du modèle (12.2)}}$$

Ces résultats étaient attendus. En effet, les deux modèles sont des modèles IP et le nombre de variables et de contraintes du modèle (12.1) est bien supérieur à celui du modèle (12.2). Une analyse rapide montre que ce dernier contient $n_v = K + KS$ [1] variables et $n_c = KT + S$ contraintes. Comparativement, le graphe associé au problème de flot maximal (12.1) comporte un nombre d'arêtes E égal à $S + K + K \times [\sum_{j \in S} T(j) + 2T + 1] > n_v + K(2T+1)$ et un nombre de sommets V égal à $S + KS + KT + K + 3 > n_c + K(S+1) + 3$. Le nombre de variables et de contraintes du problème de flot maximal auquel on ajouté plusieurs autres contraintes (contraintes (3) et (4) du modèle (12.1)) est supérieur respectivement à E (nombre de variables) et $2E + V$ (nombre de contraintes).

1. Pour simplifier les notations, on utilise la notation simplifiée A à la place de $|A|$ pour désigner le cardinal d'un ensemble A

n	m	Rapport de temps R
50	30	17.00
	60	20.26
	90	34.92
	120	39.77
100	30	11.12
	60	17.86
	90	20.80
	120	24.13
150	30	7.54
	60	10.03
	90	10.97
	120	8.37
200	30	6.09
	60	7.47
	90	3.58
	120	9.45

TABLE 12.4 – Rapport des temps de résolution entre les deux modèles

12.5.2/ COMPARAISON DU NOMBRE ET DU TYPE D'ENSEMBLES COUVRANTS GÉNÉRÉS

Nous comparons le nombre d'ensembles couvrants complets et incomplets obtenus, d'une part, en résolvant le programme IP (12.3) proposé par [Pedraza et al., 2006], et d'autre part en résolvant le problème de flot à coût minimal (12.5) que nous avons établi. Nous sommes très vite confrontés à un problème d'instabilité numérique. Nous rencontrons cette difficulté pour la résolution du problème (12.3) dès que nous dépassons les 30 capteurs et les 30 cibles. Avec notre modèle, le problème d'instabilité apparaît fréquemment pour des topologies de réseau comprenant un nombre de capteurs et de cibles supérieurs respectivement à 70 et 120. Pour les instances qui ont pu être résolues sans problème d'instabilité numérique, les deux modèles fournissent le même nombre d'ensembles couvrants complets et incomplets et la même décroissance en terme de nombre de cibles couvertes dans chaque ensemble. Contrairement à d'autres solveurs, le solveur GLPK est un outil libre (GNU General Public License) mais souffre d'une trop grande instabilité numérique dès que la taille des problèmes augmente et reste trop peu performant en temps de calcul. Nous avons donc l'intention de revoir nos programmes afin d'y intégrer une résolution avec le solveur commercial CPLEX, reconnu pour son efficacité. Le tableau 12.5 présente les résultats obtenus jusqu'à présent pour les instances qui ont pu être traitées avec GLPK. Le tableau 12.5 indique pour chaque scénario (valeur moyenne sur 20 topologies de réseau), le nombre d'ensembles couvrants complets obtenus indifféremment avec la résolution de (12.1) ou (12.2) dans la colonne 6, le nombre d'ensembles couvrants complets (colonne 5) et le pourcentage P_γ supplémentaire d'ensembles couvrants incomplets obtenus avec la résolution de (12.5) (colonnes de 2 à 4). Si γ représente le pourcentage de cibles couvertes dans un ensemble couvrant incomplet, le pourcentage P_γ s'obtient avec la formule suivante :

$$P_\gamma = \frac{\text{Nombre d'ensembles couvrant plus de } \gamma \text{ cibles} - \text{Nombre d'ensembles couvrant toutes les cibles}}{\text{Nombre d'ensembles couvrant toutes les cibles}} \times 100$$

Nos résultats montrent que la durée de vie du réseau peut être largement amélioré en

acceptant une dégradation très faible de la couverture. En effet, le fait de tolérer une couverture partielle où seulement 95% des cibles seraient couvertes, permettrait de générer entre 5% et 25% d'ensembles couvrants supplémentaires. Pour une dégradation de 10% de la couverture, on peut espérer prolonger la durée de vie du réseau de plus de 50% (par exemple, pour les instances avec 50 capteurs et 60 cibles). Étrangement, les résultats obtenus (pourcentages P_γ) ne présentent pas de régularité par rapport au nombre de capteurs ou de cibles. Une analyse plus fine serait donc nécessaire. En particulier, nous pouvons nous interroger sur la pertinence des valeurs des poids $w_{u,i,k}$ et $w_{o,i,k}$ qui ont été choisies selon la formule proposée dans [Pedraza et al., 2006].

Nous nous sommes également intéressés à la composition des ensembles couvrants complets générés avec le modèle (12.1) et nous avons comparé le nombre moyen de capteurs par ensemble généré avec celui obtenu par résolution du modèle (12.5). Le tableau 12.6 présente les valeurs N_{ec} (en pourcentage) des rapports du nombre moyen de capteurs dans chaque ensemble couvrant complet généré selon les deux méthodes employées. Le rapport NS est défini de la façon suivante (N_{moy} représente un nombre de capteurs moyen par ensemble couvrant complet généré) :

$$NS = \frac{N_{moy}^{fcmin} \text{ avec le Modèle Flot Coût Min (12.5)} - N_{moy}^{fmax} \text{ avec le Modèle Flot Max (12.1)}}{N_{moy}^{fmax} \text{ avec le Modèle Flot Max (12.1)}} \times 100$$

Nous observons que les ensembles couvrants complets générés avec notre modèle de flot à coût minimum contiennent moins de capteurs que ceux générés avec le modèle de flot maximal, entre 1 à 13% en moins selon les instances. Encore une fois, nous ne remarquons pas de corrélation entre le nombre de capteurs et de cibles et les valeurs du rapport NS.

12.6/ CONCLUSION

Nos expérimentations montrent que notre modèle IP (12.2) pour générer des ensembles couvrants complets est plus efficace que le modèle de flot maximal proposé par [Cardei and Du, 2005] car il est possible d'obtenir de manière beaucoup plus rapide (entre 3 et 40 fois) les mêmes résultats.
Nous avons mis en évidence que le problème de minimisation simultanée de la sous et sur-couverture proposé par [Pedraza et al., 2006] peut s'interpréter comme un problème de flot à coût minimum dans un graphe semblable à celui utilisé par [Cardei and Du, 2005]. Cette formulation est intéressante. D'une part, elle permet de générer autant d'ensembles couvrants complets qu'avec les modèles (12.1) et (12.2). D'autre part, les ensembles couvrants complets générés comportent généralement moins de capteurs qu'avec les modèles (12.1) et (12.2), ce qui permet la construction d'ensembles couvrants supplémentaires, certes incomplets, mais assurant tout de même une couverture partielle admissible pour certaines applications, et allongeant du même fait la durée de vie du réseau. Néanmoins, les temps prohibitifs de résolution exacte des modèles IP proposés dans ce chapitre ne permettent pas de traiter des instances de grande taille.

		Pourcentage d'ensembles couvrants supplémentaires P_γ avec modèle (12.5)			Nombre d'ensembles couvrants complets	
n	m	$\gamma = 50\%$	$\gamma = 90\%$	$\gamma = 95\%$	modèle (12.5)	modèle (12.1)
10	30	25.00	25.00	8.33	2.4	2.4
	60	50.00	20.00	20.00	2.0	2.0
	90	66.66	22.22	11.11	1.8	1.8
	120	66.66	11.11	11.11	1.8	1.8
20	30	27.77	11.11	5.56	3.6	3.6
	60	25.00	12.50	12.50	3.2	3.2
	90	25.00	12.50	12.50	3.2	3.2
	120	20.00	10.00	10.00	3.0	3.0
30	30	50.00	25.00	5.00	4.0	4.0
	60	36.84	15.79	5.26	3.8	3.8
	90	26.32	10.52	5.42	3.8	3.8
	120	33.33	11.11	5.56	3.6	3.6
40	30	76.19	23.80	9.52	4.2	4.2
	60	50.00	25.00	5.00	4.0	4.0
	90	57.89	21.05	10.53	3.8	3.8
	120	47.36	15.79	10.52	3.8	3.8
50	30	104.76	52.38	23.81	3.8	3.8
	60	89.47	52.63	42.11	3.8	3.8
	90	63.15	52.63	21.05	3.8	3.8
	120	52.63	31.58	21.05	3.8	3.8
60	30	57.14	32.14	14.29	5.6	5.6
	60	32.00	16.00	16.00	5.0	5.0
	90	80.95	28.57	23.81	4.2	4.2
	120	90.00	25.00	25.00	4.0	4.0

TABLE 12.5 – Nombre d'ensembles couvrants complets et pourcentage d'ensembles couvrants supplémentaires avec une couverture partielle (de 50%, 90% et 95%)

n	m	Rapport NS
10	30	-3.05
	60	-7.56
	90	-9.09
	120	-13.04
20	30	-6.16
	60	-12.90
	90	-6.96
	120	-7.53
30	30	-7.07
	60	-3.61
	90	–2.83
	120	-2.94
40	30	-9.02
	60	-2.05
	90	-6.05
	120	-6.13
50	30	-5.16
	60	-2.73
	90	-3.39
	120	-1.42
60	30	-2.67
	60	-2.65
	90	-2.71
	120	-1.21

TABLE 12.6 – Rapport NS du nombre moyen de capteurs par ensemble couvrant complet

IV

CONCLUSION ET PERSPECTIVES

13

CONCLUSION ET PERSPECTIVES

13.1/ SYNTHÈSE DES CONTRIBUTIONS

Depuis ma thèse, je mène des travaux de recherche théoriques et applicatifs dans le domaine de la recherche opérationnelle. L'ensemble de mes travaux témoignent d'un esprit d'ouverture sur les disciplines autres que l'informatique (tarification, transport, santé, production, énergie) et aussi sur ma capacité d'intégrer les problématiques et approches informatiques (télécommunication, réseaux, compilation, logiciels) autres que celles de mon domaine de prédilection. Parmi l'ensemble des problèmes étudiés au cours de ma carrière d'enseignant-chercheur, ce mémoire d'HDR regroupe mes principales contributions pour la modélisation et la résolution de trois problèmes d'optimisation difficiles, dans les domaines respectifs, de la curiethérapie, de la compilation et des réseaux de capteurs.

Le problème d'optimisation de la dose en curiethérapie
Nous avons réalisé un état de l'art des modèles d'optimisation de la dose en curiethérapie LDR et HDR (chapitre 5). Nous avons proposé un nouveau modèle d'optimisation (formulation PL) fondé sur la minimisation des déviations de dose en tout point du volume à traiter, et conforme aux règles du Système de Paris, pour l'implantation des vecteurs et le calcul des temps/positions d'arrêt de la source radioactive à travers ces vecteurs. Le modèle proposé a été implémenté dans un logiciel qui permet de le paramétrer et de visualiser l'irradiation du volume cible. Le développement du logiciel Isodose 3D permet de démontrer l'efficacité et l'utilité du modèle. Les principales difficultés rencontrées au cours de ces travaux portaient sur la compréhension des outils, abréviations, indicateurs, employés par les thérapeutes.

Le problème d'optimisation de registres en compilation
Cette étude traite du problème d'optimisation du besoin en stockage dans les graphes de tâches périodiques. En pratique, notre problème tend à minimiser le besoin en registres dans les programmes embarqués, où les instructions d'une boucle sont représentées par un graphe de dépendances de données cyclique (GDD). Dans nos travaux, nous supposons une exécution parallèle des instructions sans aucun modèle de ressources - la parallélisme étant borné par les contraintes de stockage uniquement. Notre but a été d'analyser le compromis entre le besoin en registres et le parallélisme dans un problème d'ordonnancement périodique de tâche. Le problème d'ordonnancement périodique de tâches (instructions) cycliques avec minimisation du nombre de registres, dans sa forme la plus générale, a été formulé sous forme d'un programme linéaire en nombre entiers où les variables de décision sont les dates de début de chacune des tâches, des

variables binaires indiquant une réutilisation ou non de registres entre deux tâches (non nécessairement distinctes), et les distances de réutilisation (chapitre 6).
Ce problème étant NP-complet, une résolution exacte s'avère trop coûteuse en pratique. Ici, nous avons exploité le fait que le modèle en question fait apparaître un problème sous-jacent d'affectation pour lequel nous connaissons un algorithme en temps polynomial (méthode Hongroise) pour proposer une heuristique appropriée appelée SIRALINA (chapitre 8). Cette heuristique, en deux phases, nécessite dans un premier temps la résolution d'un programme linéaire en nombre entier avec une matrice des contraintes totalement unimodulaire, puis la résolution d'un problème d'affectation linéaire. Nous avons montré dans le chapitre 11 que le problème dual du problème d'ordonnancement correspond à un problème de flot de coût minimum.
Nous avons testé SIRALINA sur un ensemble de benchmarks, issus de la communauté d'optimisation de codes, et correspondants à des codes de calcul haute performance pour l'embarqué. Nos expériences montrent que SIRALINA produit des solutions très satisfaisantes en très peu de temps. Par conséquent, cette méthode a été incluse dans un compilateur pour l'embarqué chez STMicroelectronics.

Le problème de maximisation de la durée de vie d'un réseau de capteurs
Parmi les nombreux problèmes d'optimisation (localisation, couverture, minimisation consommation d'énergie,...) soulevés dans le déploiement et l'utilisation des réseaux de capteurs, nous nous sommes intéressés au problème de maximisation de la durée de vie (chapitre 7). Nous avons réalisé une synthèse des modèles (centralisés) permettant de prolonger la durée de vie du réseau tout en maintenant la couverture complète des cibles ou de la zone à surveiller. Nous avons proposé plusieurs nouveaux modèles de programmation linéaire :

- Modèle (7.3) pour la génération d'un nombre maximal d'ensembles couvrants disjoints
- Modèle (7.5) pour la génération d'un nombre maximal d'ensembles couvrants non disjoints
- Modèle (7.13) pour la génération d'un ensemble couvrant un maximum de points primaires (représentatifs de la zone à couvrir) et minimisant la sur-couverture.
- Modèle (7.15) pour la couverture du périmètre de chaque capteur induisant une couverture de la zone d'intérêt.

En parallèle de ces travaux, nous avons proposé deux heuristiques de résolution pour la génération d'ensembles couvrants disjoints (chapitre 9) et non disjoints (chapitre 10). La première heuristique fait appel à la résolution d'un problème d'affectation linéaire. La seconde utilise la technique de génération de colonnes.
Dans la dernier partie de ce mémoire (chapitre 12), nous avons étudié de quelle manière le problème de génération d'ensembles couvrants disjoints pouvait se transformer en un problème de flot en nous inspirant d'une étude de référence sur ce sujet. Nous avons montré que la minimisation simultanée de sous et sur-couverture pouvait se modéliser sous forme d'un problème de flot de coût minimum (modèle 12.5). Du point de vue théorique, cette nouvelle modélisation est prometteuse, elle est plus riche dans la mesure où elle permet de construire autant d'ensembles couvrants complets que dans l'approche de référence tout en utilisant généralement moins de capteurs. D'un point de vue pratique, cette nouvelle modélisation autorise la construction d'ensembles de couverture partielle, ce qui peut être utile pour certaines applications de réseaux de capteurs.

13.2/ PERSPECTIVES

La poursuite de nos activités de recherche devrait suivre deux directions : continuation des travaux déjà engagés sur les réseaux de capteurs et travaux sur des problématiques de gestion d'énergie.

Poursuite des travaux sur les réseaux de capteurs

— Premièrement nous allons continuer à travailler sur l'optimisation de la durée de vie d'un réseau de capteurs. Une thèse va démarrer sur ce problème dans la continuité des travaux initiés autour de la génération d'ensembles couvrants complets et incomplets. Il existe des applications pour lesquelles les besoins en couverture peuvent être réduits. On parle alors de α-couverture pour $\alpha \in]0, 1]$. La couverture complète, c'est-à-dire la couverture de toutes les cibles, correspond au cas particulier $\alpha = 1$. Suivant les besoins de l'application, différentes stratégies d'α-couverture sont envisageables. Une stratégie peut consister à construire un maximum d'ensembles couvrants complets, puis des ensembles couvrants incomplets couvrant un nombre de cibles supérieur ou égal à αm (m est le nombre cibles). Nous avons vu que cette stratégie peut être traitée par la résolution du modèle d'optimisation de Pedraza [Pedraza et al., 2006] (dans sa formulation classique (12.3) ou sa version Flot à coût minimum (12.5)). Parmi les ensembles couvrants incomplets obtenus, seuls seront conservés ceux couvrant au moins αm cibles. Une autre stratégie vise à construire un maximum d'ensembles couvrants incomplets avec un nombre de cibles couvertes supérieur ou égal à αm dans chaque ensemble incomplet. De plus, nous pouvons ajouter une contrainte de "régularité/homogénéité" de couverture entre les cibles. Cette contrainte peut imposer, par exemple, que chaque cible soit couverte dans $\beta\%$ des ensembles couvrants incomplets. Nous souhaitons proposer plusieurs modèles d'optimisation pour ces différentes stratégies.

— Les garanties de connectivité entre les noeuds capteur du réseau introduisent également de nouvelles contraintes à prendre en considération dans nos modèles, notamment en cas de couverture partielle.

— A notre connaissance, peu de modèles intègrent la notion d'énergie restante pour construire des ensembles de couverture, la plupart des modèles considèrent un réseau de capteurs homogènes (avec une seule unité de captage [Xing et al., 2010] et des durées de vie initiales égales [Cardei and Wu, 2006]). Nous chercherons à tenir compte de l'hétérogénéité des capteurs pour enrichir nos modèles.

— Nous avons l'intention de simuler toutes nos méthodes sur une plate-forme de simulation du type OMNET++ pour valider leurs performances sur différentes configurations de réseaux de capteurs.

— Le principal verrou scientifique à l'usage de méthodes centralisées dans les réseaux de capteurs concerne le passage à l'échelle. Nous nous intéresserons au développement d'heuristiques efficaces pour y palier.

Problématiques d'optimisation en gestion d'énergie
Nous avons l'intention de poursuivre notre travail sur l'optimisation énergétique dans un aéronef avec l'équipe du département ENERGIE. Pour le moment, nous nous sommes concentrés sur le problème du dimensionnement qui consiste à déterminer la composition des packs d'accumulateurs et de super-condensateurs à embarquer pour apporter des sources supplémentaires d'énergie en phase de décollage. Au delà de cette problématique d'optimisation offline, nous verrons comment gérer la répartition de la demande de puissance sur les différentes sources d'énergie en fonction des besoins du système au cours du vol.

Mes activités de recherche m'ont permis de me confronter, à la modélisation, et à la résolution de plusieurs problèmes d'optimisation issus de domaines d'application variés. En recherche opérationnelle, une méthode ou une astuce de modélisation développée pour un modèle donné peut être utilisée dans le cadre d'un autre problème. Ceci s'est vérifié au cours de mes travaux. Par exemple, je me suis inspirée du principe de minimisation des écarts de dose en curiethérapie pour proposer un modèle de couverture du périmètre des capteurs. Aussi, je souhaite pouvoir traiter de nouveaux problèmes d'optimisation dans d'autres domaines d'application que ceux explorés jusqu'à présent.

A

RAPPELS DE PROGRAMMATION LINÉAIRE

A.1/ FORMULATION D'UN PROBLÈME DE PROGRAMMATION LINÉAIRE

La programmation mathématique est le nom donné aux problèmes d'optimisation sous contraintes : il s'agit de rechercher l'optimum d'une fonction de variables, étant donné que celles-ci doivent vérifier un certain nombre d'équations et/ou d'inéquations appelés contraintes. Le problème le plus simple de la programmation mathématique est celui de la programmation linéaire (PL) : il s'agit de la situation où à la fois la fonction à optimiser et les contraintes à respecter sont linéaires (il serait plus général de dire "affines"), c'est-à-dire du premier degré en les variables. Le programme linéaire s'écrit dans sa forme la plus générale :

$$\begin{cases} opt \quad z = \sum_{j=1}^{j=n} c_j x_j & \\ \text{sous :} & \\ \sum_{j=1}^{j=n} a_{ij} x_j \geq d_i & i = 1, .., s \\ \sum_{j=1}^{j=n} a_{ij} x_j \leq d_i & i = s+1, .., p \\ \sum_{j=1}^{j=n} a_{ij} x_j = d_i & i = p+1, .., m \\ x_j \text{ de signe quelconque} & j = 1, .., n \end{cases} \tag{A.1}$$

— Il y a n variables notées x_j ($j = 1, .., n$),
— z représente la fonction à optimiser (fonction objectif) ; c_j est le cœfficient de la variable x_j dans cette fonction z.
— i ($i = 1, .., m$) est l'indice des m contraintes réelles (équations ou inéquations) ; a_{ij} et d_i représentent respectivement le cœfficient de la variable x_j dans la contrainte i et le terme indépendant de cette contrainte.

Cette forme générale peut être simplifiée, dans une forme dite ”canonique”. De manière évidente, il est toujours possible de ramener :

— l'optimisation à un minimisation,
— toutes les égalités à 2 inégalités (en dédoublant la contrainte d'égalité),
— toutes les inégalités à des inégalités de même type,
— toutes les variables à être non négatives.

En notation vectorielle, la forme canonique est donc :

$$\begin{cases} Min & z = cx \\ \text{sous :} & \\ & Ax \geq d \\ & x \geq 0 \end{cases} \tag{A.2}$$

où $x = (x_1, ..., x_n) \in \mathbb{R}^n$ est un vecteur colonne des variables, A est la matrice $(m \times n)$ des contraintes de cœfficient (a_{ij}) et d le vecteur colonne du second membre.
De nombreux problèmes réels peuvent être exprimés comme des programmes linéaires. Les programmes linéaires peuvent être résolus efficacement par certains algorithmes (par exemple, , algorithme de points intérieurs,). Il a été montré que l'algorithme du Simplexe [Dantzig, 1963] pouvait prendre un temps de calcul exponentiel. Néanmoins, il reste très efficace en pratique et il est implémenté dans tous les solveurs d'optimisation linéaire. Les algorithmes de points intérieurs ([Karmarkar, 1984], [Ye, 1991]) plus récents permettent de résoudre les problèmes en un temps polynomial.
Dans certains problèmes, on requiert en plus que les variables ne prennent que des valeurs entières (contraintes dites d'intégrité), voire que les valeurs 0 ou 1. On parle alors de programme linéaire en nombres entiers. Ces derniers problèmes sont beaucoup plus difficiles à résoudre que les problèmes d'optimisation linéaire à variables continues, à l'exception de certains problèmes dits classiques (par exemple, problème d'affectation linéaire, problème de transport,...).
On dit que $x \in \mathbb{R}^n$ est une solution **réalisable** du problème (A.2) si x satisfait aux contraintes, autrement dit si $Ax \geq d$. L'ensemble P de toutes les solutions réalisables d'un problème d'optimisation est appelé son **ensemble réalisable**. On dit que $x \in \mathbb{R}^n$ est une solution **optimale** du problème (A.2) si x est une solution réalisable du problème (A.2), et si de plus, quelle que soit la solution réalisable $y \in \mathbb{R}^n$ du problème A.2, on a nécessairement $cx \leq cy$. Autrement dit, une solution réalisable est optimale si elle minimise la fonction objectif sur l'ensemble réalisable.

A.2/ DUALITÉ EN PROGRAMMATION LINÉAIRE

La **dualité** associe à toute problème linéaire un autre problème linéaire qui est appelé problème **dual** du problème initial ; par opposition le problème initial est appelé problème **primal**. Reprenons tout d'abord un problème linéaire sous la forme générale :

$$\begin{cases} min & z = \sum_{j=1}^{j=n} c_j x_j & \\ \text{sous :} & & \\ & \sum_{j=1}^{j=n} a_{ij} x_j \geq d_i & i = 1, .., p \\ & \sum_{j=1}^{j=n} a_{ij} x_j = d_i & i = p+1, .., m \\ & x_j \geq 0 & j = 1, .., q \\ & x_j \text{ de signe quelconque} & j = q+1, .., n \end{cases} \tag{A.3}$$

Par convention, les contraintes d'inégalité d'un problème de minimisation (maximisation) seront considérées sous la forme "≥" ("≤"). Le problème dual est le suivant :

$$\begin{cases} max \quad z = \sum_{i=1}^{i=m} d_i y_i & \\ \text{sous :} & \\ y_i \geq 0 & i = 1, .., p \\ y_i \text{ de signe quelconque} & i = p+1, .., m \\ \sum_{i=1}^{i=m} a_{ij} y_i \leq c_j & j = 1, .., q \\ \sum_{i=1}^{i=m} a_{ij} y_i = c_j & j = q+1, .., n \end{cases} \tag{A.4}$$

Les variables $x_j(y_i)$ sont appelées variables primales (duales). Cette définition est caractérisée par les règles suivantes :

- à toute contrainte primale correspond une variable duale : si la contrainte est une inégalité, la variable duale est soumise à condition de non-négativité ; si la contrainte est une égalité, la variables duale est de signe quelconque ;
- à toute variable primale correspond une contrainte duale : si la variable est soumise à une condition de non-négativité, la contrainte duale est une inégalité ; si la variable est de signe quelconque, la contrainte duale est une égalité ;
- les cœfficient de la fonction objectif du primal deviennent les seconds membres des contraintes duales ; les seconds membres des contraintes primales deviennent les coefficients de la fonction objectif du dual ;
- le cœfficient de la variable x_j dans la contrainte i devient, dans le dual, le cœfficient de la variable y_i dans la contrainte j.

A chaque programme linéaire (primal) à minimiser, on associe un programme linéaire (dual) à maximiser. Le dual permet de définir une bonne borne inférieure de la fonction objectif du problème primal. Les valeurs des fonctions objectif du primal et du dual coïncident à l'optimum sous certaines conditions. D'autres propriétés et théorèmes [Vazirani, 2006] liés à la dualité ne sont pas présentés ici.

A.3/ MÉTHODE DE GÉNÉRATION DE COLONNES

A.3.1/ PRINCIPE DE BASE

La génération de colonnes [Guy Desaulniers, 2005] est utilisée dans la résolution de programmes linéaires avec peu de contraintes et un grand nombre de variables. La technique a été suggérée en 1958 par Ford et Fulkerson [L. R. Ford and Fulkerson, 1958] pour le problème de flot multiproduit, puis généralisée par Dantzig et Wolfe [Dantzig and Wolfe, 1960b].
Un programme linéaire avec beaucoup de variables (les variables se traduisent par des colonnes dans la matrice des contraintes) peut être difficile à résoudre rapidement. Cependant les solutions optimales peuvent être obtenues sans inclure toutes les colonnes dans la matrice des contraintes. En fait, seul un petit nombre de variables participent à la solution optimale et les autres peuvent être ignorées (variables hors base). C'est sur ce principe que repose la méthode de génération de colonnes.

A.3.2/ ALGORITHME DE GÉNÉRATION DE COLONNES

Prenons le programme linéaire A.5 écrit sous forme canonique :

$$\begin{cases} Min & z = cx \\ \text{sous :} & \\ & Ax \geq d \\ & x \geq 0 \end{cases} \tag{A.5}$$

Étape 0
Choisir un petit nombre de variables pour constituer le programme linéaire de départ. Soit J cet ensemble.
Étape 1

- Résoudre le programme linéaire restreint (appelé aussi Problème Maître restreint (PMR)) aux variables de l'ensemble J. Remarque : on utilise la notation vectorielle $x_{(J)}$ pour désigner le vecteur colonne des variables restreint à l'ensemble J.
- Obtenir une solution optimale $x^*_{(J)}$ du problème PMR telle que :

$$c_{(J)}x^*_{(J)} = Min\left\{c_{(J)}x_{(J)} \text{ sous } A_{(J)}x_{(J)} \geq d, x_{(J)} \geq 0, x_{(J)} \in \mathbb{R}^{(J)}\right\}$$

Étape 2
Déterminer les valeurs des variables duales $y^* \in \mathbb{R}^m$ associées au (PMR)
Étape 3
Rechercher de nouvelles variables au coût réduit négatif (problème auxiliaire (AUX)). Le coût réduit $\overline{c_j}$ d'une variable x_j ($x_j \notin J$) est donné par $\overline{c_j} = c_j - \sum_{i=1}^{i=m} a_{ij}y^*_i$.

- Si $\overline{c_j} \geq 0 \quad \forall x_j \notin J$, la solution du (PMR) est solution optimale du problème d'origine.
- S'il existe une ou plusieurs variables x_j au coût réduit négatif, ajouter la ou les variables/colonnes correspondantes.$J \longleftarrow J + \{j\}$. Construire la nouveau problème maître restreint et retourner à l'étape 1.

B

PROBLÈME D'AFFECTATION LINÉAIRE

B.1/ DÉFINITION

Le problème d'affectation linéaire est un problème classique de recherche opérationnelle. Une illustration courante de ce problème d'affectation est l'affectation de n agents à n tâches. Chaque agent ne peut être affecté qu'à une seule tâche et chaque tâche doit être exécutée par un seul agent. Une matrice $n \times n$ d'entiers positifs contient le coût d'affectation de chaque agent à chacune des tâches. Le but est de minimiser le coût total d'exécution des tâches. Plus formellement, on peut définir c_{ij} le coût d'affectation de l'agent i à la tâche j et les variables binaires x_{ij}. Le problème d'affectation linéaire peut alors se formuler sous forme d'un programme linéaire en nombres entiers.

$$\begin{cases} \min \sum_{i=1..n} \sum_{j=1..n} c_{ij} x_{ij} & \\ \text{sous :} & \\ \sum_{j=1..n} x_{ij} = 1 & \forall i = 1..n \\ \sum_{i=1..n} x_{ij} = 1 & \forall j = 1..n \\ x_{ij} \in \{0, 1\} & \end{cases} \tag{B.1}$$

Le premier groupe de contraintes indique qu'un agent ne peut réaliser qu'une et une seule tâche. Le deuxième groupe de contraintes garantit qu'une tâche est réalisée par un et un seul agent.
L'algorithme hongrois ou méthode hongroise (parfois appelé aussi algorithme de Kuhn-Munkres) est un algorithme d'optimisation combinatoire, qui résout le problème d'affectation linéaire en temps polynomial. Il a été proposé en 1955 par le mathématicien américain Harold Kuhn, qui l'a baptisé « méthode hongroise » parce qu'il s'appuyait sur des travaux antérieurs de deux mathématiciens hongrois. James Munkres a revu l'algorithme en 1957 [Munkres, 1957] et a prouvé qu'il s'exécutait en temps polynomial $0(n^4)$. Une amélioration proposée par Edmonds et Karp [Edmonds and Karp, 1972], (et Tomizawa [Tomizawa, 1971] indépendamment) permet de réduire le temps d'exécution à $0(n^3)$. La méthode hongroise consiste à réaliser des opérations sur les lignes et les colonnes de la matrice pour faire apparaître des zéros dits indépendants (c'est-à-dire un zéro sélectionné par ligne et par colonne dans la matrice des coûts).

B.2/ MÉTHODE HONGROISE

Phase 1 : Obtention initiale de zéros

1. Soustraire ligne par ligne, puis colonne par colonne (ou l'inverse) le plus petit élément de la ligne ou de la colonne.

Phase 2 : Recherche d'une solution de coût nul

1. Prendre la ligne contenant le moins de zéros,
2. Encadrer le premier zéro de cette ligne et barrer les autres zéros de la ligne et de la colonne du zéro encadré,
3. retour à 1 jusqu'à impossibilité d'encadrer un zéro.

Phase 3 : Recherche d'une solution optimale

1. Procédure de marquage des lignes et des colonnes
   - **(a)** Marquer d'une croix les lignes ne contenant aucun 0 encadré (s'il n'y en a pas : FIN)
   - **(b)** Marquer d'une croix toute colonne qui a un zéro barré sur une ligne marquée,
   - **(c)** Marquer d'une croix toute ligne qui a un zéro encadré dans une colonne marquée,
   - **(d)** Retour à (b) et (c) jusqu'à ce qu'il n'y ait plus de ligne ou de colonne à marquer (si toutes les colonnes sont marquées : FIN)
2. Tracer un trait horizontal sur chaque ligne non marquée et un trait vertical sur chaque colonne marquée.
3. Choisir le plus petit élément du tableau non rayé, l'ajouter aux lignes rayées et le soustraire aux colonnes non rayées.
4. Retour à la phase 2

Phase finale (sortie en 3.1.(d)) : Obtention de la solution optimale
Remplacer alternativement les zéros barrées (resp. encadrés) ayant servi au marquage de cette colonne par des zéros encadrés (resp. barrés).

B.3/ EXEMPLE

Prenons l'exemple d'une problème d'affectation de taille $n = 5$ avec la matrice de coût suivante :

15	40	5	20	20
22	33	9	16	20
40	6	28	0	26
8	0	7	25	60
10	10	60	15	5

Phase 1

15	40	5	20	20
22	33	9	16	20
40	6	28	0	26
8	0	7	25	60
10	10	60	15	5
−8	0	−5	0	−5

7	40	0	20	15	0
14	33	4	16	15	−4
32	6	23	0	21	0
0	0	2	25	55	0
2	10	55	15	0	0

Phase 2

7	40	∅	20	15
10	29	[0]	12	11
32	6	23	[0]	21
0	[0]	2	25	55
2	10	55	15	[0]

Phase 3, étape 1

		$*(b)$			
7	40	∅	20	15	$*(a)$
10	29	[0]	12	11	$*(c)$
32	6	23	[0]	21	
0	[0]	2	25	55	
2	10	55	15	[0]	

Phase 3, étape 2

		$*(b)$			
7	40	∅	20	15	$*(a)$
10	29	[0]	12	11	$*(c)$
32	6	23	[0]	21	
0	[0]	2	25	55	
2	10	55	15	[0]	

Phase 3, étape 3

		$*(b)$			
7	40	∅	20	15	$*(a)$
10	29	0	12	11	$*(c)$
32	6	23	0	21	+7
0	0	2	25	55	+7
2	10	55	15	0	+7
−7	−7	0	−7	−7	

Fin

0	33	∅	13	8
3	22	0	5	4
32	6	30	0	21
∅	0	9	25	55
2	10	62	15	0

L'affectation résultante donne les couples (1,1), (2,3),(3,4),(4,2) et (5,5) et un coût total d'affectation égal à 8+5+5+4-7-7-7+7+7+7+7=29.

C

RAPPELS DE THÉORIE DES GRAPHES

C.1/ GRAPHE ET DÉFINITIONS

Un **graphe orienté** $G = (V, E)$ est défini par la donnée de deux ensembles :

— un ensemble non vide fini V dont les éléments sont appelés sommets ou nœuds,

— un ensemble fini E de couples ordonnées de sommets (i, j) appelées arcs.

On note $e = (i, j)$ l'arc qui lie le sommet i au sommet j. Le sommet i est l'origine (ou **extrémité initiale**) et j l'extrémité (ou **extrémité terminale/finale**) de l'arc. On peut dessiner un graphe, en représentant un sommet i par un point ou par un cercle, et chaque arc (i, j) par une flèche reliant le sommet origine au sommet extrémité. Dans le graphe de la figure C.1, $V = \{1, 2, 3, 4\}$ et $E = \{(1, 2), (1, 4), (2, 1), (1, 3), (2, 3), (3, 4)\}$.

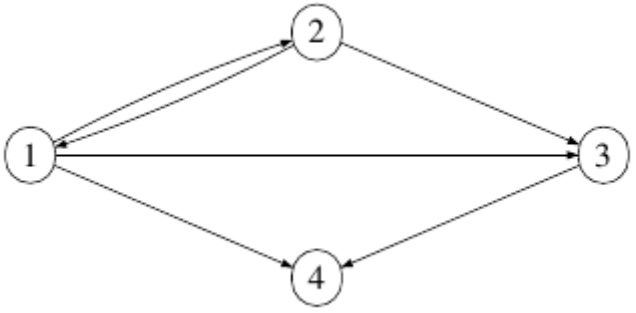


FIGURE C.1 – Graphe orienté

On appelle **ordre** d'un graphe le nombre de sommets de ce graphe, noté aussi $|V|$ (cardinal de l'ensemble V). La **taille** du graphe est le nombre d'arêtes du graphe, noté aussi $|E|$. Lorsque plusieurs arêtes relient deux sommets, on les appelle des arêtes **multiples**. Une **boucle** est une arête dont les deux extrémités sont identiques. Un graphe est **simple** s'il ne contient ni boucle ni arêtes multiples. Un **multigraphe** ou **graphe multiple** est un graphe qui n'est pas simple.

Les graphes sont souvent utilisés pour modéliser des problèmes associés à des parcours ou à des successions d'actions. Pour cela, les notions de **chemin**, et de **chaîne** sont introduites.

On appelle **chemin** d'origine x et d'extrémité y une séquence d'arcs telle que :

— le premier arc a pour origine x,

— l'origine de tous les autres coïncide avec l'extrémité de l'arc qui la précède dans la séquence,
— le dernier arc a pour extrémité y.

Par exemple, dans la figure C.1, la séquence $((1,2);(2,3);(3,4))$ est un chemin d'origine 1 et d'extrémité 4. Un **circuit** est un chemin dont le nœud initial et le nœud terminal coïncident. Si on ne tient pas compte de l'orientation des arcs, on parle de **cycle**. Un circuit est **élémentaire** s'il ne passe pas deux fois par le même sommet.

On appelle **chaîne** reliant x à y une séquence d'arcs telle que :
— le premier arc est adjacent à x par une de ses extrémités, et au deuxième arc de la séquence par son autre extrémité,
— le dernier arc est adjacent à y par une de ses extrémités, et à l'avant-dernier arc de la séquence par son autre extrémité,
— chaque arc intermédiaire est adjacent au précédent par une de ses extrémités et au suivant par l'autre.

Par exemple, dans la figure C.1, la séquence $((3,4);(1,4);(2,1))$ est une chaîne liant 3 au sommet 2 (on ne tient pas compte de l'orientation des arcs).

C.2/ RÉSEAU DE TRANSPORT ET FLOT

Un **réseau de transport** noté $\mathfrak{R} = (G = (V,E), s, t, c)$ est formé de :

— $G = (V,E)$ un graphe orienté,
— $s \in V$, sommet du graphe G qui n'a pas de prédécesseur, appelé sommet **source**,
— $t \in V$ sommet du graphe G, qui n' a pas de successeur, appelé sommet destination ou **puits**,
— $c : E \to \mathbb{N}$ application capacité (à chaque arc $(i,j) \in E$ est associée une capacité $c(i,j) \geq 0$).

C.2.1/ FLOT RÉALISABLE

Soit $\mathfrak{R} = (G = (V,E), s, t, c)$ un réseau. Un flot f dans $\mathfrak{R}$ est une application $f : E \to \mathbb{N}$. Un flot f est **réalisable** dans $\mathfrak{R}$ si les contraintes (inégalités) suivantes sont satisfaites :

— contraintes de capacité

$$0 \leq f(i,j) \leq c(i,j) \quad \forall (i,j) \in E$$

— contraintes de conservation de flot (Loi de Kirchhoff)

$$\sum_{i|(i,j)\in E} f(i,j) - \sum_{k|(j,k)\in E} f(j,k) = 0 \quad \forall j \in V \setminus \{s,t\}$$

(quantité qui entre dans j = quantité qui sort de j)

La figure C.2 représente un réseau $\mathfrak{R} = (G = (V,E), s, t, c)$ avec $V = \{s, v_1, v_2, t\}$ et $E = \{(s,v_1),(s,v_2),(v_1,v_2),(v_1,t),(v_2,t)\}$. Les valeurs de flot et de capacité (lisibles entre crochets) sont indiquées à côté de chaque arc.

Le flot f défini sur ce réseau est réalisable. Les contraintes des capacité sont vérifiées. Par exemple, pour l'arc (v_1,t), $f(v_1,t) = 3$, $c(v_1,t) = 4$ et $f(v_1,t) \leq c(v_1,t)$. Les contraintes

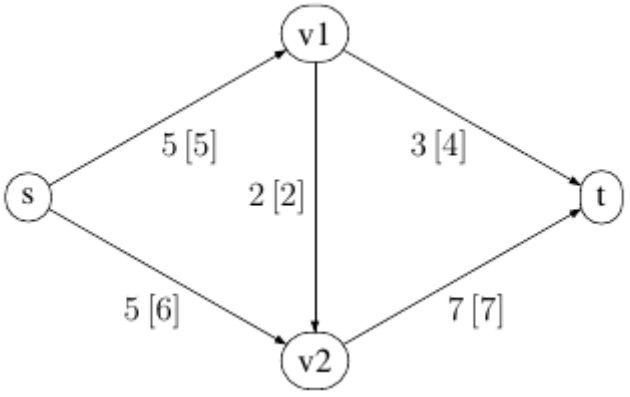


FIGURE C.2 – Réseau de transport (avec flot[capacité] sur chaque arête)

de conservation de flot sont satisfaites en v_1 et v_2. En v_1, la contrainte de conservation s'écrit :
$f(s, v_1) - f(v_1, v_2) - f(v_1, t) = 5 - 2 - 3 = 0$

C.2.2/ VALEUR D'UN FLOT

La définition de flot réalisable exige que les contraintes de conservation de flot soient vérifiées en chaque sommet $j \in V \setminus \{s, t\}$.
En additionnant ces contraintes pour tous les $j \in J \subseteq V \setminus \{s, t\}$, on voit clairement que le flot est conservé dans tous les ensembles $J \subseteq V \setminus \{s, t\}$:

(quantité qui entre dans J = quantité qui sort de J)

Cette propriété est vraie pour $J = V \setminus \{s, t\}$.

$$\sum_{j|(s,j)\in E} f(s, j) + \sum_{j|(t,j)\in E} f(t, j) = \sum_{j|(j,s)\in E} f(j, s) + \sum_{j|(j,t)\in E} f(j, t)$$

La **valeur d'un flot** f réalisable entre s et t est la quantité de flot envoyée de s à t. On la note F.

$$F = \sum_{j|(s,j)\in E} f(s, j) - \sum_{j|(j,s)\in E} f(j, s)$$

$$F = \sum_{j|(j,t)\in E} f(j, t) - \sum_{j|(t,j)\in E} f(t, j)$$

Dans l'exemple de la figure C.2, la valeur du flot est de 10 : $F = 5 + 5 = 3 + 7 = 10$.

C.3/ LE PROBLÈME DU FLOT MAXIMUM

C.3.1/ DÉFINITION

Soit un réseau $\mathfrak{R} = (G = (V, E), s, t, c)$. Le problème du **flot maximum** consiste à déterminer un flot réalisable entre s et t qui soit de valeur maximum.

C.3.2/ EXEMPLE

On remarque que le flot donné dans le réseau de la figure C.2 n'est pas maximum. En effet, on peut trouver un flot de valeur 11, comme le montre la figure C.3. Le nouveau flot est maximum. En effet, on remarque qu'on peut faire rentrer au maximum 11 unités de flot dans t à cause des limites de capacités (4 et 7) sur les arcs entrants (v_1, t) et (v_2, t).

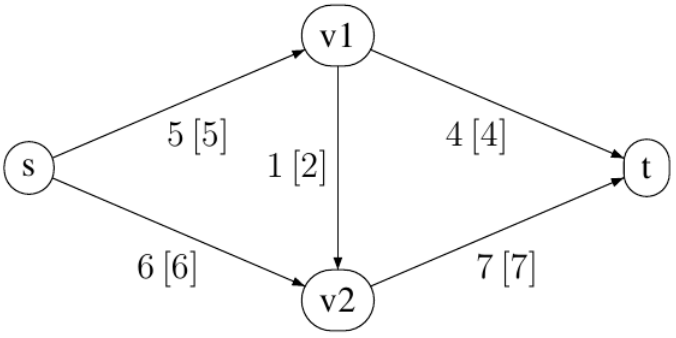


FIGURE C.3 – Réseau de transport et flot maximum

C.3.3/ FLOT MAXIMUM ET PROGRAMMATION LINÉAIRE

Soit un réseau $\mathfrak{R} = (G = (V, E), s, t, c)$. Soit $f(i, j)$ le flot transitant sur l'arc (i, j), $\forall (i, j) \in E$, et F la valeur du flot f. Le problème du flot maximum entre s et t peut se formuler sous la forme du programme linéaire suivant :

$$\begin{cases} \max F \\ \text{sous :} \\ \sum_{i|(i,j)\in E} f(i,j) - \sum_{k|(j,k)=\in E} f(j,k) = -F & \text{pour } j = s \\ \sum_{i|(i,j)\in E} f(i,j) - \sum_{k|(j,k)=\in E} f(j,k) = F & \text{pour } j = t \\ \sum_{i|(i,j)\in E} f(i,j) - \sum_{k|(j,k)=\in E} f(j,k) = 0 & \text{pour } j \neq s,t \\ 0 \leq f(i,j) \leq c(i,j), & \forall (i,j) \in E \end{cases} \tag{C.1}$$

Ce programme linéaire a $(|E| + 1)$ variables et $(2|E| + |V|)$ contraintes.

C.3.4/ ALGORITHMES DE RÉSOLUTION

Différents algorithmes ont été développés pour résoudre le problème du flot maximum. Parmi les plus connus figurent l'algorithme de Ford - Fulkerson [Ford and Fulkerson, 2010] et sa variante, l'algorithme d'Edmonds-Karp [Edmonds and Karp, 1972]. Sa complexité est en $O(|V||E|^2)$. L'algorithme itératif consiste à partir d'un flot réalisable, puis à l'augmenter en cherchant un chemin de la source au puits dans un graphe (dit graphe résiduel). Plutôt que de choisir un chemin augmentant au hasard, il faut systématiquement considérer le plus court chemin de s vers t au sens du nombre d'arcs dans le chemin. Cette modification garantit la terminaison de l'algorithme.

C.4/ LE PROBLÈME DE FLOT À COÛT MINIMUM

C.4.1/ DÉFINITION

Soit un réseau $\mathfrak{R} = (G = (V, E), s, t, c)$. De plus, chaque arc du réseau est muni d'un coût u_{ij} dit coût unitaire par unité de flot. Le sommet source s a une production de flot égale à d unités et le noeud puits a une demande de d unités. Le problème de flot à coût minimum consiste à trouver un flot f réalisable sur le réseau satisfaisant aux contraintes de production et de demande tel que le coût du flot soit minimal. Il permet de modéliser tout un ensemble de problèmes pratiques dans lesquels il s'agit de trouver une manière optimale d'acheminer une ressource (par exemple, un fluide, de l'électricité) d'une source à un puits (ou de plusieurs sources à plusieurs puits).
Le problème du flot de coût minimum est fondamental dans la mesure où la plupart des autres problèmes de flots (problème de flot maximum, problème de transport) peuvent en être vus comme des cas particuliers.

C.4.2/ FLOT À COÛT MINIMUM ET PROGRAMMATION LINÉAIRE

Soit un réseau $\mathfrak{R} = (G = (V, E), s, t, c)$. Soient $f(i, j)$ le flot transitant sur l'arc (i, j) et $u(i, j)$ le coût de transport sur l'arc (i, j). L'offre au noeud source s est égale à d et la demande au noeud puits t est égale à d. Le problème du flot à coût minimum entre s et t peut se formuler sous la forme du programme linéaire suivant :

$$\begin{cases} \max \sum_{(i,j)\in E} u(i,j) f(i,j) & \\ \text{sous :} & \\ \sum_{i|(i,j)\in E} f(i,j) - \sum_{k|(j,k)=\in E} f(j,k) = -d & \text{pour } j = s \\ \sum_{i|(i,j)\in E} f(i,j) - \sum_{k|(j,k)=\in E} f(j,k) = d & \text{pour } j = t \\ \sum_{i|(i,j)\in E} f(i,j) - \sum_{k|(j,k)=\in E} f(j,k) = 0 & \text{pour } j \neq s,t \\ 0 \leq f(i,j) \leq c(i,j), & \forall (i,j) \in E \end{cases} \tag{C.2}$$

C.4.3/ ALGORITHMES DE RÉSOLUTION

Le problème peut être résolu par programmation linéaire, dans la mesure où la fonction à minimiser, et les différentes contraintes sont linéaires. Plusieurs autres algorithmes existent, certains pouvant être considérés comme des généralisations de l'algorithme de Ford-Fulkerson, d'autres comme des généralisations de l'algorithme de poussage/réétiquetage, ou encore des variantes de l'algorithme du simplexe. Les algorithmes les plus connus sont entre autres :

- Cycle canceling [Klein, 1967] et Minimum mean cycle canceling [Goldberg and Tarjan, 1989] : algorithme fondée sur la recherche de cycle de coût négatif
- Successive shortest path and capacity scaling [Edmonds and Karp, 1972] : méthode duale vue comme une généralisation de l'algorithme de Ford-Fulkerson
- cost scaling [Goldberg and Tarjan, 1990] : approche primale-duale, vue comme une généralisation de l'algorithme de poussage/réétiquetage
- Algorithme du simplexe pour les réseaux [Orlin, 1997]

BIBLIOGRAPHIE

[Ahuja and ans James B. Orlin, 1993] Ahuja, R. K. and ans James B. Orlin, T. L. M. (1993). *Network Flows*. Prentice Hall. ISBN=0-13-61-617549-X.

[Alterovitz et al., 2006] Alterovitz, R., Lessard, E., Pouliot, J., Hsu, I.-C. J., O'Brien, J. F., and Goldberg, K. (2006). Optimization of hdr brachytherapy dose distributions using linear programming with penalty costs. *Medical Physics*, 33(11) :4012–4019.

[Barth et al., 2004a] Barth, D., Deschinkel, K., Diallo, M., and Echabbi, L. (2004a). Pricing, QoS and Utility models for the Internet . Research Report # 2004/60, Laboratoire Prism.

[Barth et al., 2004b] Barth, D., Deschinkel, K., Diallo, M., and L.Echabbi (2004b). Description of current cost and payment models for communication services and networks. Research Report, EURO-NGI Project, PRiSM (France) GET (France), INRIA (France), AUEB (Grèce), Université de Rome Tor Vergata (Italie), UC-Spain (Espagne).

[Berman et al., 2004a] Berman, P., Calinescu, G., Shah, C., and Zelikovsky, A. (2004a). Power efficient monitoring management in sensor networks. In *Wireless Communications and Networking Conference, 2004. WCNC. 2004 IEEE*, volume 4, pages 2329–2334 Vol.4.

[Berman et al., 2004b] Berman, P., Calinescu, G., Shah, C., and Zelikovsky, A. (2004b). Power efficient monitoring management in sensor networks. In *Wireless Communications and Networking Conference, WCNC. 2004*.

[Bettwy et al., 2014] Bettwy, M., Deschinkel, K., and Gomes, S. (2014). An optimization tool for process planning and scheduling. In *APMS 2014, Advances in Production Management Systems. Innovative and Knowledge-Based Production Management in a Global-Local World. IFIP WG 5.7 International Conference*, volume 438, pages 443–450, Ajaccio, France. Springer.

[Caprara et al., 1998] Caprara, A., Fischetti, M., and Toth, P. (1998). Algorithms for the set covering problem. *Annals of Operations Research*, 98 :2000.

[Cardei and Du, 2005] Cardei, M. and Du, D.-Z. (2005). Improving wireless sensor network lifetime through power aware organization. *Wirel. Netw.*, 11 :333–340.

[Cardei et al., 2005] Cardei, M., Thai, M., Li, Y., and Wu, W. (2005). Energy-efficient target coverage in wireless sensor networks. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 3, pages 1976–1984 vol. 3.

[Cardei and Wu, 2006] Cardei, M. and Wu, J. (2006). Energy-efficient coverage problems in wireless ad-hoc sensor networks. *Computer communications*, 29(4) :413–420.

[Carrabs et al., 2015a] Carrabs, F., Cerulli, R., DAmbrosio, C., Gentili, M., and Raiconi, A. (2015a). Maximizing lifetime in wireless sensor networks with multiple sensor families. *Computers Operations Research*, 60 :121 – 137.

[Carrabs et al., 2015b] Carrabs, F., Cerulli, R., DAmbrosio, C., and Raiconi, A. (2015b). A hybrid exact approach for maximizing lifetime in sensor networks with complete and partial coverage constraints. *Journal of Network and Computer Applications*, 58 :12 – 22.

[Castaño et al., 2014] Castaño, F., Rossi, A., Sevaux, M., and Velasco, N. (2014). A column generation approach to extend lifetime in wireless sensor networks with coverage and connectivity constraints. *Computers & Operations Research*, 52(B) :220–230.

[Cormen et al., 1990] Cormen, T., Leiserson, C. E., and Rivest, R. (1990). *Introduction to Algorithms*. MIT Press, McGraw-Hill, Cambridge, Massachusetts.

[Couchot et al., 2012] Couchot, J.-F., Deschinkel, K., and Salomon, M. (2012). Suitability of artificial neural network for mems-based flow control. In Bourgeois, J. and de Labachelerie, M., editors, *dMEMS 2012, Workshop on design, control and software implementation for distributed MEMS*, pages 1–6, Besançon, France.

[Couchot et al., 2013] Couchot, J.-F., Deschinkel, K., and Salomon, M. (2013). Active mems-based flow control using artificial neural network. *Mechatronics*, 23(7) :898 – 905.

[Crawley, 2007] Crawley, M. J. (2007). *The R Book*. Wiley. ISBN-13 : 978-0-470-51024-7.

[Dantzig and Wolfe, 1960a] Dantzig, G. and Wolfe, P. (1960a). Decomposition principle for linear programs. *Operations Research*, pages 101–111.

[Dantzig, 1963] Dantzig, G. B. (1963). *Linear Programming and Extensions*. Princeton University Press, Princeton.

[Dantzig and Wolfe, 1960b] Dantzig, G. B. and Wolfe, P. (1960b). Decomposition principle for linear programs. *Operations Research*, 8(1) :101–111.

[de Dinechin, 1996] de Dinechin, B. D. (1996). Parametric Computation of Margins and of Minimum Cumulative Register Lifetime Dates. In David C. Sehr and Utpal Banerjee and David Gelernter and Alexandru Nicolau and David A. Padua, editor, *LCPC*, volume 1239 of *Lecture Notes in Computer Science*, pages 231–245. Springer.

[Deffrenne et al., 2013] Deffrenne, C., Bettwy, M., Robert, A., Deschinkel, K., and Gomes, S. (2013). Parameter management in configuration for the design of products families. In *APMS 13, Advances in Production Management Systems. Sustainable Production and Service Supply Chains*, volume 415 of *IFIP AICT*, pages 501–508, State College, PA, United States.

[Demoly et al., 2011] Demoly, F., Yan, X.-T., Eynard, B., Rivest, L., and Gomes, S. (2011). An assembly oriented design framework for product structure engineering and assembly sequence planning. *Robot. Comput.-Integr. Manuf.*, 27(1) :33–46.

[Deng et al., 2012] Deng, X., Jiguo Yu, D. Y., and Chen, C. (2012). Transforming area coverage to target coverage to maintain coverage and connectivity for wireless sensor networks. *International Journal of Distributed Sensor Networks*, 2012, Article ID 254318 :1–12.

[Deschinkel, 1998] Deschinkel, K. (1998). Recherche avec tabous et génération de colonnes pour un problème de conception de réseaux. Research Report no 98-51, Centre de Recherche sur les transports.

[Deschinkel, 1999] Deschinkel, K. (1999). Optimisation dynamique des prix d'utilisation d'un réseau. Application à la régulation du trafic aérien . Research Report RD 1/7602.11 DCSD, ONERA.

[Deschinkel, 2000] Deschinkel, K. (2000). Optimisation dynamique des prix d'utilisation d'un réseau. Application à la régulation du trafic aérien . Research Report RD 2/03409.02F DCSD, ONERA.

[Deschinkel, 2003] Deschinkel, K. (2003). Régulation du trafic aérien par optimisation dynamique des prix d'utilisation du réseau. In *ROADEF'03, conférence de la Société Française de Recherche Opérationnelle et d'Aide á la Décision*, Avignon.

[Deschinkel, 2004] Deschinkel, K. (2004). An overview on utility functions for Internet networks. Research Report # 2004/55, Laboratoire Prism.

[Deschinkel, 2012] Deschinkel, K. (2012). A column generation based heuristic to extend lifetime in wireless sensor network. *Sensors and Transducers Journal*, Vol. 14-2 :242–253. 10.1007/s10878-010-9332-8.

[Deschinkel et al., 2004] Deschinkel, K., Baille, F., and Diallo, M. (2004). Pricing strategy for resource reservation on peak and off-peak periods . In *Applied Mathematical Programming and Modeling VII, APMOD*, Brunel, Royaume-Uni.

[Deschinkel et al., 2001a] Deschinkel, K., Delahaye, D., and Farges, J.-L. (2001a). Pricing Policies for Air Traffic Assignment. In *Air Transportation System Engineering*, volume 193 of *Progress in Astronautics and Aeronautics*. AIAA.

[Deschinkel and Echabbi, 2004] Deschinkel, K. and Echabbi, L. (2004). An offline routing model for MPLS networks. In *Applied Mathematical Programming and Modeling VII, APMOD*, Brunel, Royaume-Uni.

[Deschinkel et al., 2000a] Deschinkel, K., Farges, J.-L., and Delahaye, D. (2000a). Optimization of Prices for Air Traffic Control. In *9th IFAC Symposium on Control in Transportation Systems 2000*, Braunschweig, Allemagne.

[Deschinkel et al., 2000b] Deschinkel, K., Farges, J.-L., and Delahaye, D. (2000b). Pricing Policies for Air Traffic Assignment . In *3rd USA/Europe Air Traffic Management R & D Seminar*, Naples, Italie.

[Deschinkel et al., 2001b] Deschinkel, K., Farges, J.-L., and Delahaye, D. (2001b). Optimizing and Assigning Price Levels for Air Traffic Management. In *9th WCTR World Conference on Transport Research*, Séoul, Corée.

[Deschinkel et al., 2002a] Deschinkel, K., Farges, J.-L., and Delahaye, D. (2002a). Optimization of prices for Air Traffic Control. *Transportation Research, Part C*.

[Deschinkel et al., 2002b] Deschinkel, K., Farges, J.-L., and Delahaye, D. (2002b). Optimizing and assigning price levels for air traffic management. *Transportation Research Part E : Logistics and Transportation Review*, 38(3–4) :221 – 237.

[Deschinkel et al., 2008a] Deschinkel, K., Galea, F., Louat, C., and Roucairol, C. (2008a). Intégration de méthodes de coupes : Librairies Glop et Glock. In *ROADEF'08, conférence de la Société Française de Recherche Opérationnelle et d'Aide á la Décision*, Clermont-Ferrand.

[Deschinkel et al., 2005a] Deschinkel, K., Galea, F., and Roucairol, C. (2005a). A continuous tabu search method for catheter placement optimisation in HDR brachytherapy. In *Workshop on Optimization in Medicine*, Coimbra, Portugal.

[Deschinkel et al., 2005b] Deschinkel, K., Galea, F., and Roucairol, C. (2005b). Catheter placement and dwell time computations. In *Annual Meeting of INFORMS (Institut for Operations Research and the Management Sciences)*, San Fransisco, Etats-Unis.

[Deschinkel et al., 2005c] Deschinkel, K., Galea, F., and Roucairol, C. (2005c). Overview of optimization problems in HDR brachytherapy. In *Workshop on Optimization in Medicine*, Coimbra, Portugal.

[Deschinkel et al., 2006a] Deschinkel, K., Galea, F., and Roucairol, C. (2006a). Automated Optimization of Brachytherapy Treatment Plans : Software Isodose 3D. In *Applied Optimization and Metaheuristics Innovation*, Ukraine.

[Deschinkel et al., 2006b] Deschinkel, K., Galea, F., and Roucairol, C. (2006b). Optimization problems in treating cancer tumour by internal radiations : High Dose Rate Brachytherapy . In *Applied Mathematical Programming and Modeling APMOD*, Madrid, Espagne.

[Deschinkel et al., 2006c] Deschinkel, K., Galea, F., and Roucairol, C. (2006c). Problèmes d'optimisation en curiethérapie. In *Conférence Francophone de Modélisation et Simulation, Modélisation, Optimisation et Simulation*, Rabat, Maroc.

[Deschinkel et al., 2008b] Deschinkel, K., Galea, F., and Roucairol, C. (2008b). Basics for vector implantation schemes in hdr brachytherapy using a new linear programming model. *APJOR, Asia-Pacific Journal of Operational Research*.

[Deschinkel and Hakem, 2013] Deschinkel, K. and Hakem, M. (2013). A near optimal algorithm for lifetime optimization in wireless sensor networks. In van Sinderen, M., Postolache, O., and Benavente-Peces, C., editors, *SENSORNETS 2013, 2nd Int. Conf. on Sensor Networks*, pages 197 – 202, Barcelona, Spain. SciTePress.

[Deschinkel and Oudot, 2005] Deschinkel, K. and Oudot, A. (2005). Planification et tarification dans un réseau. In *ROADEF'05, conférence de la Société Française de Recherche Opérationnelle et d'Aide á la Décision*, Tours.

[Deschinkel and Touati, 2008a] Deschinkel, K. and Touati, S.-A.-A. (2008a). Efficient method for periodic task scheduling with storage requirement minimisation. In *COCOA'08, 2nd Int. Conf. on Combinatorial Optimization and Applications*, volume 5165 of *LNCS*, pages 438–447, St John, Canada.

[Deschinkel and Touati, 2008b] Deschinkel, K. and Touati, S.-A.-A. (2008b). A two-phases heuristic for the problem of periodic scheduling with storage minimisation. In *ECCO'08, European Chapter on Combinatorial Optimization*, pages ***–***, Dubrovnik, Croatie.

[Deschinkel and Touati, 2009] Deschinkel, K. and Touati, S.-A.-A. (2009). Une heuristique efficace pour l'ordonnancement périodique de tâches avec contraintes de stockage. In *ROADEF'09, 10e conférence de la Société Française de Recherche Opérationnelle et d'Aide á la Décision*, Nancy, France.

[Deschinkel and Touati, 2010] Deschinkel, K. and Touati, S.-A.-A. (2010). Elimination des circuits nuls dans les graphes cycliques pour l'ordonnancement périodique de tâches. In *ROADEF'10, 11e conférence de la Société Française de Recherche Opérationnelle et d'Aide á la Décision*, Toulouse, France.

[Deschinkel et al., 2011] Deschinkel, K., Touati, S.-A.-A., and Briais, S. (2011). Siralina : efficient two-steps heuristic for storage optimisation in single period task scheduling. *Journal of Combinatorial Optimization*, 22(4) :819 – 844.

[Edmonds and Karp, 1972] Edmonds, J. and Karp, R. M. (1972). Theoretical improvements in algorithmic efficiency for network flow problems. *J. ACM*, 19(2) :248–264.

[Ford and Fulkerson, 2010] Ford, D. R. and Fulkerson, D. R. (2010). *Flows in Networks*. Princeton University Press, Princeton, NJ, USA.

[Galea, 2006] Galea, F. (2006). *Problèmes d'optimisation en curiethérapie*. PhD thesis.

[Gallagher and Lee, 1997] Gallagher, R. and Lee, E. (1997). Mixed integer programming optimization models for brachytherapy treatment planning. *jounral of the American Medical Informatics Association*, pages 278–282.

[Garey and Johnson, 1990] Garey, M. and Johnson, D. (1990). *Computers and Intractability ; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA.

[Gentili and Raiconi, 2013] Gentili, M. and Raiconi, A. (2013). α-coverage to extend network lifetime on wireless sensor networks. *Optimization Letters*, 7(1) :157–172.

[Gerbaulet et al., 2002] Gerbaulet, T., Pötter, R., Mazeron, J., Meertens, H., and Limbergen, E. V. (2002). *The GEC-ESTRO Handbook of Brachytherapy*. ESTRO, Brussels, Belgium.

[Goldberg, 1992] Goldberg, A. V. (1992). An Efficient Implementation Of A Scaling Minimum-Cost Flow Algorithm. *Journal of Algorithms*, 22 :1–29.

[Goldberg and Tarjan, 1989] Goldberg, A. V. and Tarjan, R. E. (1989). Finding minimum-cost circulations by canceling negative cycles. *J. ACM*, 36(4) :873–886.

[Goldberg and Tarjan, 1990] Goldberg, A. V. and Tarjan, R. E. (1990). Finding minimum-cost circulations by successive approximation. *Math. Oper. Res.*, 15(3) :430–466.

[Guy Desaulniers, 2005] Guy Desaulniers, J. D. e. M. M. S. (2005). *Column Generation*. Springer-Verlag, New York.

[Hamacher et al., 2005] Hamacher, H., Pedersen, C., and Ruzika, S. (2005). Finding representative systems for discrete bicriteria optimization problems by box algorithms. Technical Report 94, University of Kaiserslautern, Department of Mathematics.

[Hanen and Munier, 1995] Hanen, C. and Munier, A. (1995). A Study of the Cyclic Scheduling Problem on Parallel Processors. *Discrete Applied Mathematics*, 57(2-3) :167–192.

[He et al., 2014] He, S., Gong, X., Zhang, J., Chen, J., and Sun, Y. (2014). Curve-based deployment for barrier coverage in wireless sensor networks. *IEEE Transactions on Wireless Communications*, 13(2) :724–735.

[Holm et al., 2016] Holm, ., Carlsson Tedgren, ., and Larsson, T. (2016). Heuristics for integrated optimization of catheter positioning and dwell time distribution in prostate hdr brachytherapy. *Annals of Operations Research*, 236(2) :319–339.

[Holm et al., 2013] Holm, , Larsson, T., and Tedgren, C. (2013). A linear programming model for optimizing hdr brachytherapy dose distributions with respect to mean dose in the dvh-tail. *Medical Physics*, 40(8) :081705–n/a. 081705.

[Huang and Tseng, 2005] Huang, C.-F. and Tseng, Y.-C. (2005). The coverage problem in a wireless sensor network. *Mobile Networks and Applications*, 10(4) :519–528.

[Hung and Lui, 2010] Hung, K.-S. and Lui, K.-S. (2010). Perimeter coverage scheduling in wireless sensor networks using sensors with a single continuous cover range. *EURASIP Journal on Wireless Communications and Networking*, 2010.

[Idrees, 2015] Idrees, A. K. (2015). *Techniques d'optimisation distribuées de la couverture pour améliorer la durée de vie des réseaux de capteurs sans fils*. PhD thesis, Université de Franche-comté.

[Idrees et al., 2014] Idrees, A. K., Deschinkel, K., Salomon, M., and Couturier, R. (2014). Coverage and lifetime optimization in heterogeneous energy wireless sensor networks. In *ICN 2014, 13-th Int. Conf. on Networks*, pages 49–54, Nice, France.

[Idrees et al., 2015a] Idrees, A. K., Deschinkel, K., Salomon, M., and Couturier, R. (2015a). Distributed lifetime coverage optimization protocol in wireless sensor networks. Technical Report RR-FEMTO-ST-2623, FEMTO-ST.

[Idrees et al., 2015b] Idrees, A. K., Deschinkel, K., Salomon, M., and Couturier, R. (2015b). Distributed lifetime coverage optimization protocol in wireless sensor networks. *The journal of Supercomputing*, 71(12) :4578–4593.

[Idrees et al., 2016] Idrees, A. K., Deschinkel, K., Salomon, M., and Couturier, R. (2016). Perimeter-based coverage optimization to improve lifetime in wireless sensor networks. *Engineering Optimization*, 48(11) :1951 – 1972.

[Jaggi and Abouzeid, 2006] Jaggi, N. and Abouzeid, A. (2006). Energy-efficient connected covereage in wireless sensor networks. In *Proceeding of 4th Asian International Mobile Computing Conference AMOC2006*, pages 77–86.

[Jain, 1991] Jain, R. (1991). *The Art of Computer Systems Performance Analysis : Techniques for Experimental Design, Measurement, Simulation, and Modeling*. John Wiley and Sons, Inc., New York.

[Karabis et al., 2009] Karabis, A., Belotti, P., and Baltas, D. (2009). *Optimization of Catheter Position and Dwell Time in Prostate HDR Brachytherapy using HIPO and Linear Programming*, pages 612–615. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Karmarkar, 1984] Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4) :373–395.

[Karouzakis et al., 2002] Karouzakis, K., Lahanas, M., Milickovic, N., Giannouli, S., Baltas, D., and Zamboglou, N. (2002). Brachytherapy dose-volume histogram computations using optimized stratified sampling methods. *Medical Physics*, 29 :424–432.

[Kim and Cobb, 2013] Kim, H. and Cobb, J. A. (2013). Maximum lifetime of reinforced barrier-coverage in wireless sensor networks. In *19th IEEE International Conference on Networks (ICON), 2013*, pages 1–6.

[Klein, 1967] Klein, M. (1967). A primal method for minimal cost flows with applications to the assignment and transportation problems. *Management Science*, 14(3) :205–220.

[Kuhn, 1955] Kuhn, H. W. (1955). The Hungarian Method for the assignment problem. *Naval Research Logistics Quarterly*, 2 :83–97.

[Kumagai, 2004] Kumagai, J. (2004). Life of birds [wireless sensor network for bird study]. *Spectrum, IEEE*, 41(4) :42–49.

[L. R. Ford and Fulkerson, 1958] L. R. Ford, J. and Fulkerson, D. R. (1958). A suggested computation for maximal multi-commodity network flows. *Management Science*, 5(1) :97–101.

[Lahanas et al., 2003] Lahanas, M., Baltas, D., and Zamboglou, N. (2003). A hybrid evolutionary algorithm for multiobjective anatomy based dose optimization in hdr brachytherapy. *Phys. Med. Biol.*, 48 :399–415.

[Lawler, 1972] Lawler, E. L. (1972). Optimal Cycles on Graphs and Minimal Cost-to-Time Ratio Problem. In Marzojlo, A., editor, *Periodic Optimization*, volume 1, pages 38–58. Springer-Verlag.

[Lee et al., 1999] Lee, E. K., Gallagher, R. J., Silvern, D., Wuu, C.-S., and Zaider, M. (1999). Treatment planning for brachytherapy : an integer programming model, two computational approaches and experiments with permanent prostate implant planning. *Physics in Medicine and Biology*, 44(1) :145.

[Lee and Zaider, 2003] Lee, E. K. and Zaider, M. (2003). Mixed integer programming approaches to treatment planning for brachytherapy – application to permanent prostate implants. *Annals of Operations Research*, 119(1) :147–163.

[Lessard and Pouliot, 2001] Lessard, E. and Pouliot, J. (2001). Inverse planning anatomy-based dose optimisation for hdr-brachytherapy of the prostate using fast simulated annealing and dedicated objective function. *Medical Physics*, 28 :773–779.

[Li and Vasilakos, 2013] Li, M. and Vasilakos, A. V. (2013). A survey on topology control in wireless sensor networks : Taxonomy, comparative study, and open issues. *Proceedings of the IEEE*, 101(12) :2538–2557.

[Li et al., 2011] Li, Y., Vu, C., Ai, C., Chen, G., and Zhao, Y. (2011). Transforming complete coverage algorithms to partial coverage algorithms for wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 22(4) :695–703.

[Ling and Znati, 2009] Ling, H. and Znati, T. (2009). Energy efficient adaptive sensing for dynamic coverage in wireless sensor networks. In *Wireless Communications and Networking Conference, 2009. WCNC 2009. IEEE*, pages 1–6.

[Louat et al., 2007] Louat, C., Deschinkel, K., Roucairol, C., and Gonzalez, R. (2007). Mixing Gomory mixed integer inequalities with others cutting plane. In *European Chapter on Combinatorial Optimization*, Chypre.

[Mahkorin, 2010] Mahkorin, A. (2010). *GNU Linear Programming Kit, Reference Manual*.

[Milickovic et al., 2002] Milickovic, M., Lahanas, M., Pagagiannopoulou, M., Zamboglou, N., and Baltas, D. (2002). Multiobjective anatomy-based optimization for hdr-brachytherapy with constrained free deterministic algorithms. *Phys. Med. Biol.*, 47 :2263–2280.

[Misra et al., 2011] Misra, S., Kumar, M. P., and Obaidat, M. S. (2011). Connectivity preserving localized coverage algorithm for area monitoring using wireless sensor networks. *Computer Communications*, 34(12) :1484–1496.

[Munkres, 1957] Munkres, J. (1957). On the assignment and transportation problems (abstract). *Naval Research Logistics Quarterly*, 4(1) :77–78.

[Orlin, 1997] Orlin, J. B. (1997). A polynomial time primal network simplex algorithm for minimum cost flows. *Mathematical Programming*, 78(2) :109–129.

[Padmavathy and Chitra, 2010] Padmavathy, T. and Chitra, M. (2010). Extending the network lifetime of wireless sensor networks using residual energy extraction—hybrid scheduling algorithm. *Int. J. of Communications, Network and System Sciences*, 3(1) :98–106.

[Pedraza et al., 2006] Pedraza, F., Medaglia, A. L., and Garcia, A. (2006). Efficient coverage algorithms for wireless sensor networks. In *Proceedings of the 2006 Systems and Information Engineering Design Symposium*, pages 78–83.

[Pujari, 2011] Pujari, A. K. (2011). High-energy-first (hef) heuristic for energy-efficient target coverage problem. *International Journal of Ad Hoc, Sensor & Ubiquitous Computing*, 2(1) :45–58.

[Qu and Georgakopoulos, 2013] Qu, Y. and Georgakopoulos, S. V. (2013). A distributed area coverage algorithm for maintenance of randomly distributed sensors with adjustable sensing range. In *IEEE Global Communications Conference (GLOBECOM), 2013*, pages 286–291.

[Robert, 2012] Robert, A. (2012). *Toward a methodology of structuring the interactions dynamic within the Multi-Domains and Multi-Views design model : Application to the design of modular product families*. Theses, Université de Technologie de Belfort-Montbeliard.

[Robert et al., 2011a] Robert, A., Deschinkel, K., Roth, S., Yan, X. T., and Gomes, S. (2011a). Approche dfa et conception fonctionnelle de produits modulaires : le modèle fard. In *CIGI 2011, 9ème congrès international de Génie Industriel*, page 8 pages, Québec, Canada.

[Robert et al., 2011b] Robert, A., Yan, X. T., Roth, S., Deschinkel, K., and Gomes, S. (2011b). A new approach to modularity in product development - utilising assembly sequence knowledge. In *ICED'11, 18th Int. Conf. on Engineering Design*, pages 236–248, Copenhague, Denmark.

[Robert et al., 2011c] Robert, A., Yan, X. T., Roth, S., Deschinkel, K., and Gomes, S. (2011c). Vers une nouvelle approche de conception »hautement productive» intégrant la démarche dfa. In *AIP PRIMECA 2011, 12e Colloque National*, pages ***–***, Le Mont-Dore, France.

[Saenger et al., 2015] Saenger, P., Deschinkel, K., Devillers, N., and Péra, M.-C. (2015). An optimal sizing of electrical energy storage system for an accurate energy management in an aircraft. In *IEEE Vehicle Power and Propulsion Conference (VPPC) 2015*, pages SS2–1 (6 pages). IEEE.

[Saenger et al., 2016] Saenger, P., Devillers, N., Deschinkel, K., Pera, M. C., Couturier, R., and Gustin, F. (2016). Optimization of electrical energy storage system sizing for an accurate energy management in an aircraft. *IEEE Transactions on Vehicular Technology*, PP(99) :1–1.

[Schrijver, 1987] Schrijver, A. (1987). *Theory of Linear and Integer Programming*. John Wiley and Sons, New York.

[Slijepcevic and Potkonjak, 2001] Slijepcevic, S. and Potkonjak, M. (2001). Power efficient organization of wireless sensor networks. In *IEEE International conference on Communications*, pages 472–476.

[Tomizawa, 1971] Tomizawa, N. (1971). On some techniques useful for solution of transportation network problems. *Networks*, 1(2) :173–194.

[Touati, 2002] Touati, S.-A.-A. (2002). *Register Pressure in Instruction Level Parallelisme*. PhD thesis, Université de Versailles, France. ftp.inria.fr/INRIA/Projects/a3/touati/thesis.

[Touati et al., 2013] Touati, S.-A.-A., Briais, S., and Deschinkel, K. (2013). How to eliminate non-positive circuits in periodic scheduling : a procative strategy based on shortest path equations. *RAIRO Operations Research*, 47(3) :223–249.

[Touati et al., 2011] Touati, S.-A.-A., Deschinkel, K., and Dupont-De-Dinechin, B. (2011). Efficient spilling reduction for software pipelined loops in presence of multiple register types in embedded vliw processors. *ACM Transactions on Embedded Computing Systems (TECS)*, 10(4) :47 :1–47 :21.

[Touati and Eisenbeis, 2004] Touati, S.-A.-A. and Eisenbeis, C. (2004). Early Periodic Register Allocation on ILP Processors. *Parallel Processing Letters*, 14(2). World Scientific.

[Vazirani, 2006] Vazirani, V. V. (2006). *Introduction à la dualité en programmation linéaire*, pages 103–119. Springer Paris, Paris.

[Vu et al., 2006] Vu, C., Gao, S., Deshmukh, W., and Li, Y. (2006). Distributed energy-efficient scheduling approach for k-coverage in wireless sensor networks. In *IEEE Military Communications Conference, 2006. MILCOM 2006*, pages 1–7.

[Vu, 2009] Vu, C. T. (2009). *Distributed energy-efficient solutions for area coverage problems in wireless sensor networks*. PhD thesis, Georgia State University.

[Wang, 2011] Wang, B. (2011). Coverage problems in sensor networks : A survey. *ACM Computing Surveys (CSUR)*, 43(4) :32–53.

[Xing et al., 2010] Xing, X., Li, J., and Wang, G. (2010). Integer programming scheme for target coverage in heterogeneous wireless sensor networks. In *Mobile Ad-hoc and Sensor Networks (MSN), 2010 Sixth International Conference on*, pages 79–84.

[Yan et al., 2008] Yan, T., Gu, Y., He, T., and Stankovic, J. A. (2008). Design and optimization of distributed sensing coverage in wireless sensor networks. *ACM Transactions on Embedded Computing Systems (TECS)*, 7(3) :33.

[Yang and Chin, 2014a] Yang, C. and Chin, K.-W. (2014a). Novel algorithms for complete targets coverage in energy harvesting wireless sensor networks. *IEEE Communications Letters*, 18(1) :118–121.

[Yang and Chin, 2014b] Yang, C. and Chin, K.-W. (2014b). A novel distributed algorithm for complete targets coverage in energy harvesting wireless sensor networks. In *IEEE ICC 2014- Ad-hoc and Sensor Networking Symposium*, pages 361–366.

[Yang and Liu, 2014] Yang, M. and Liu, J. (2014). A maximum lifetime coverage algorithm based on linear programming. *Journal of Information Hiding and Multimedia Signal Processing, Ubiquitous International*, 5(2) :296–301.

[Ye, 1991] Ye, Y. (1991). An o(n^3l)) potential reduction algorithm for linear programming. *Mathematical Programming*, 50 :239–258.

[Zhang and Hou, 2005] Zhang, H. and Hou, J. C. (2005). Maintaining sensing coverage and connectivity in large sensor networks. *Ad Hoc & Sensor Wireless Networks*, 1(1-2).

[Zhou et al., 2009] Zhou, Z., Das, S. R., and Gupta, H. (2009). Variable radii connected sensor cover in sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 5(1) :8.

[Zorbas et al., 2010] Zorbas, D., Glynos, D., Kotzanikolaou, P., and Douligeris, C. (2010). Solving coverage problems in wireless sensor networks using cover sets. *Ad Hoc Networks*, 8(4) :400–415.

TABLE DES FIGURES

LISTE DES TABLES

Résumé :

Ce document réunit une partie de mes contributions à la modélisation et la résolution de problèmes d'optimisation difficiles. Les problèmes traités dans ce mémoire sont issus du domaine de la santé, de la compilation, et des réseaux de capteurs. Ce mémoire se décompose en trois parties. La première partie présente l'état de l'art et la construction de nouveaux modèles de programmation linéaire pour l'optimisation de dose en curiethérapie, la minimisation du nombre de registres processeurs en compilation, et la maximisation de la durée de vie d'un réseau de capteurs.
La seconde partie de ce document expose nos heuristiques, dédiées à la résolution des problèmes présentés en première partie, et se montrant particulièrement efficaces. Celles-ci s'appuient sur des outils classiques de recherche opérationnelle : décomposition du problème en sous-problèmes connus, affectation linéaire et technique de génération de colonnes.
Dans la troisième partie, nous nous intéressons à la modélisation sous forme de problèmes de flots pour proposer de nouvelles approches de résolution, notamment pour la construction d'ensembles de couverture dans un réseau de capteurs.

Mots-clés : Optimisation combinatoire, Programmation linéaire, Graphe, Curiethérapie, Registres processeurs, Réseau de capteurs

Abstract:

This paper summarizes my contributions for modeling and solving difficult optimization problems. The problems dealt with in this document come from the domains of health, compilation, and sensor networks. This paper is divided into three parts. The first part presents the state of the art and the construction of new models of linear programming for dose optimization in brachytherapy, minimization of the number of processor registers in compilation, and lifetime maximization in wireless sensor networks (WSN) . The second part of this document presents our heuristics, dedicated to solving problems presented in the first part, and showing themselves to be particularly effective. These rely on classical operational research tools: decomposition of the problem into well-known subproblems, linear assignment and column generation method.
In the third part, we are interested in network flow models to propose new resolution approaches, especially for the construction of cover sets in WSN.

Keywords: Combinatorial Optimization, Linear Programming, Graph, Brachytherapy, Processor Registers, Sensor Network

SPIM

■ École doctorale SPIM 16 route de Gray F - 25030 Besançon cedex
■ tél. +33 (0)3 81 66 66 02 ■ ed-spim@univ-fcomte.fr ■ www.ed-spim.univ-fcomte.fr

UFC
UNIVERSITÉ DE FRANCHE-COMTÉ