UNIVERSITÉ PARIS 13
LABORATOIRE D'INFORMATIQUE DE PARIS NORD

HABILITATION À DIRIGER DES RECHERCHES

spécialité « Informatique »

par

Lucas Létocart

REFORMULATION, RELAXATION ET RÉOPTIMISATION

Soutenue le 8 septembre 2014 devant le jury composé de :

JACQUES DESROSIERS	HEC Montréal	(Rapporteur)
ANTONIO FRANGIONI	Università di Pisa	
RIDHA MAHJOUB	Université Paris Dauphine	
PHILIPPE MICHELON	Université d'Avignon et des Pays de Vaucluse	(Rapporteur)
CHRISTOPHE PICOULEAU	Conservatoire National des Arts et Métiers	(Rapporteur)
FRÉDÉRIC ROUPIN	Université Paris 13	
ROBERTO WOLFLER CALVO	Université Paris 13	

À Mathieu et Alice

REMERCIEMENTS

Je voudrais tout d'abord exprimer mes plus profonds remerciements à Jacques Desrosiers, Philippe Michelon et Christophe Picouleau qui m'on fait l'honneur d'accepter d'être rapporteurs de cette habilitation. Je suis fier de l'intérêt qu'ils témoignent ainsi à mon travail et je les remercie chaleureusement pour leur contribution.

Je remercie également Antonio Frangioni et Ridha Mahjoub pour avoir accepté sans hésitation de faire partie de mon jury et pour l'attention qu'ils ont accordé à la lecture de ce manuscrit.

Je tiens évidemment à remercier Frédéric Roupin et Roberto Wolfler Calvo pour leur contribution dans cette habilitation, pour leur soutien et pour leur présence dans ce jury.

Je voudrais également remercier l'ensemble de mes co-auteurs pour leurs contributions respectives, certains d'entre eux ayant été essentiels à mon épanouissement scientifique, je pense notamment à Marie-Christine Costa, Anass Nagih, Gérard Plateau, Frédéric Roupin, Céline Rouveirol et Roberto Wolfler Calvo, envers qui je suis très reconnaissant de m'avoir guidé vers et dans la carrière de chercheur ; je pense aussi à Hanane Allaoua, Nicolas Lermé, Karima Mouhoubi, Paolo Gianessi et Nora Touati Moungla que j'ai eu le plaisir de co-encadrer pendant leurs doctorats.

Mes remerciements vont également à Laurent Alfandari, Sylvie Borne, Alberto Ceselli, Fabio Furini, Monique Guignard, Sophie Toulouse, Emiliano Traversi et Angelika Wiegele pour leur implication et avec qui j'ai le plaisir de collaborer depuis plusieurs années et qui sont devenus des amis sincères.

Je remercie tous ceux grâce auxquels j'ai toujours travaillé dans une ambiance agréable à l'université Paris 13, les membres de l'équipe AOC et du LIPN en particulier.

Je remercie Colette, Marie et Sylvie pour la relecture attentive et les conseils qu'elles m'ont apportés pour la rédaction de ce mémoire.

Enfin, je remercie ma famille et ma belle-famille, ma mère, mon défunt père, mon frère, Gaby, Gérard, Agnès et Laurent pour leur soutien indéfectible. Je conclurai en remerciant de tout cœur Marie pour son soutien sans faille et sa patience, pour me supporter tous les jours et pour m'avoir donné mes deux plus belles réalisations, Mathieu et Alice, à qui je dédie ce mémoire.

TABLE DES MATIÈRES

LISTE DES FIGURES

PRÉFACE. . .

CE document présente une synthèse de mes travaux de recherche de ces dix dernières années en tant que maître de conférences à l'Université Paris 13. Ces travaux s'inscrivent dans le cadre de la résolution de problèmes d'optimisation combinatoire en nombres entiers, avec des aspects théoriques, algorithmiques et applicatifs.

En plus d'une introduction (chapitre 1) et d'une conclusion (chapitre 4), ce document comporte deux chapitres principaux.

Le chapitre 2 s'intéresse aux problèmes de (multi-)flots, à leur formulation, leur reformulation, leur résolution et les réductions que l'on peut apporter au graphe pour résoudre plus efficacement ces problèmes.

Le Chapitre 3 concerne le développement d'approches exactes et/ou approchées pour des problèmes d'optimisation en nombres entiers via des reformulations, des relaxations, des décompositions et la réoptimisation.

INTRODUCTION

1

Ce document présente une synthèse de mes travaux de recherche de ces dix dernières années. Le trait commun des travaux présentés ci-après concerne la reformulation de problèmes d'optimisation combinatoire de type (multi-) flots et de type sac à dos afin de développer des approches de résolution algorithmiques, mathématiques et combinatoires innovantes et efficaces pour ces problèmes. Les problèmes étudiés seront de type (multi-) flots (et problèmes connexes (chemins, (multi-)coupes, routage, tournées, ...), et de type sac à dos (et ses extensions : bin packing, affectation généralisée, quadratique, ...). Ces problèmes apparaissent par exemple comme sous-problèmes des problèmes de planification en transport, de traitement d'images et de recherche de motifs fréquents et les méthodes de résolution envisagées seront alors basées sur l'algorithmique de graphes et la programmation mathématique.

Nous nous intéresserons en particulier au cadre de la réoptimisation et aux méthodes intégrant des techniques de relaxation et de décomposition.

Ce document comporte deux chapitres principaux.

Le chapitre 2 s'intéresse aux problèmes de (multi-) flots, à leur formulation, leur reformulation, leur résolution et les réductions que l'on peut apporter au graphe pour les résoudre plus efficacement. Nous présenterons dans un premier temps des algorithmes polynomiaux pour la résolution des problèmes de multi-coupes minimales et de multiflots maximaux en nombres entiers dans les graphes en anneaux, préalablement réduits. Nous proposerons ensuite une nouvelle technique de programmation dynamique, la programmation dynamique par blocs, afin de réduire l'espace de recherche des solutions et nous l'appliquerons au problème de plus court chemin avec fenêtres de temps. Nous nous intéresserons ensuite au problème de segmentation d'images pour lequel nous utiliserons une technique de reformulation afin de se ramener à la résolution de problèmes de flots dans des graphes particuliers et nous proposerons des réductions de ces graphes afin de résoudre plus efficacement ce problème pour des images volumineuses. Enfin, nous reformulerons la recherche de motifs contraints en fouille de données comme une suite de problèmes de flots et nous proposerons ensuite une algorithmique dédiée innovante.

Le chapitre 3 concerne les aspects relatifs à la programmation mathématique afin d'élaborer des approches de résolution exactes et/ou approchées pour des problèmes encore mal résolus. Différentes formulations des problèmes étudiés seront considérées afin de construire des schémas de résolution complémentaires. Ceux-ci s'appuieront notamment sur des relaxations et décompositions des modélisations proposées, qui pourront ensuite être insérées dans des schémas de séparation et d'évaluation progressive. Nous introduirons également le concept de réoptimisation qui nous fournira un cadre de travail pour le développement d'approches innovantes dans différents contextes. Ce chapitre est composé de deux parties. La première partie (section 3.1) se concentrera principalement sur la méthode de décomposition de Dantzig-Wolfe et sur la résolution des problèmes induits par ce type de décomposition via la génération de colonnes. Nous introduirons notamment la notion de diversification et nous nous intéresserons aux techniques de réoptimisation lorsque l'on doit faire face à une modification partielle des données. Nous étudierons particulièrement les problèmes de type localisation et/ou routage (tournées de véhicules) dans cette partie. La seconde partie de ce chapitre (section 3.2) concerne l'introduction de techniques de réoptimisation et de reformulation dans le cadre de la programmation non linéaire, quadratique en particulier, pour des problèmes de type sac à dos.

Le dernier chapitre présente une conclusion générale des travaux présentés dans ce document. Il permet également d'aborder nos autres travaux et de donner un bref aperçu des différents travaux en cours. Ce chapitre se terminera par la présentation de plusieurs perspectives de recherche.

Pour compléter ce mémoire, un curriculum vitae est donné en annexe A afin notamment de donner une synthèse des différentes activités de recherche, d'administration et d'animation scientifique.

2 REFORMULATION, RÉDUCTION ET RÉSOLUTION DES PROBLÈMES DE (MULTI-)FLOTS

SOMMAIRE

DANS ce chapitre, nous nous intéressons aux problèmes de (multi)-flots, à leur formulation, leur reformulation, leur résolution et les réductions que l'on

peut apporter au graphe pour les résoudre plus efficacement. Nous nous intéressons ici aux méthodes issues de l'algorithmique de graphes afin d'élaborer des approches de résolution efficaces. Certains problèmes considérés seront a priori très différents (traitement d'images, fouille de données bio-informatiques) mais qui peuvent en fait être reformulés comme des problèmes de flots sur des graphes particuliers.

Nous présentons dans un premier temps des algorithmes polynomiaux pour la résolution des problèmes de multicoupes minimales et de multiflots maximaux en nombres entiers dans les graphes en anneaux, préalablement réduits. Nous proposons ensuite une nouvelle technique de programmation dynamique, la programmation dynamique par blocs, afin de réduire l'espace de recherche des solutions, et nous l'appliquons au problème de plus court chemin (cas particulier du problème de flots) avec fenêtres de temps. Nous nous intéressons ensuite au problème de segmentation d'images pour lequel nous utilisons une technique de reformulation afin de se ramener à la résolution de problèmes de flots dans des graphes particuliers. De plus, afin de résoudre plus efficacement (de manière exacte et approchée) ce problème pour des images volumineuses, nous proposons des réductions du graphe. Enfin, nous reformulons la recherche de motifs contraints en fouille de données comme une suite de problèmes de flots et nous proposons une algorithmique dédiée innovante pour résoudre ce problème difficile.

2.1 MULTICOUPES ET MULTIFLOTS DANS LES ANNEAUX

Dans cette section, nous montrons comment résoudre en temps polynomial les problèmes de multicoupes minimales et de multiflots maximaux en nombres entiers dans les graphes en anneaux. De plus, nous proposons une procédure en temps linéaire pour résoudre ces deux problèmes dans les anneaux avec capacité uniforme.

Soit $G = (V, E)$ un graphe non orienté avec une capacité (ou poids) positive et entière u_e sur chaque arête e de E et soit une liste $\mathfrak{L}$ de K paires de sommets $\{s_k, t_k\}$, $k = 1, \ldots, K$. On note $n = |V|$ et $m = |E|$. On associe un flot à chaque paire de sommets $\{s_k, t_k\}$. Le problème du multiflot maximal en nombres entiers, que nous appellerons dans la suite $IMFP$ (Integral Maximum Multicommodity Flow Problem), consiste à maximiser la somme des valeurs des flots entiers F_k routés de s_k à t_k. Le problème de la $s_k - t_k$ multicoupe minimale, que nous appellerons dans la suite $IMCP$ (Integral Minimum MultiCut Problem), consiste à trouver un ensemble d'arêtes de poids minimal dont le retrait coupe chaque chaîne menant de s_k à t_k, $k = 1, \ldots, K$. Ces deux problèmes sont connus pour être $\mathcal{NP}$-difficiles et Max SNP-difficiles pour $K \geq 3$ dans des graphes quelconques (Dalhaus et al. (1994) et Garg et al. (1997)).

Un anneau $\mathfrak{R}$ est un graphe connexe dans lequel tous les sommets sont de degré 2 et $|V| = |E|$, c'est-à-dire que $n = m$.

De nombreuses applications dans les réseaux de télécommunications utilisent cette structure d'anneau (Cosares et Saniee (1994), Myung (2006), Schrijver et al. (1998), Vachani et al. (1996)). On peut notamment citer les SONET (Synchronous Optical Network) dans lesquels les équipements de transmission en fibres optiques sont configurés en anneaux. Il peut être intéressant de considérer des anneaux orientés ou non orientés en fonction du problème que l'on souhaite résoudre.

Notons tout d'abord qu'il est facile de transformer une instance non orientée d'un problème de multiflot ou d'un problème de multicoupe en une instance orientée. En effet, chaque flot, correspondant à un couple $\{s_k, t_k\}$, ne peut passer que par deux chemins, soit dans le sens des aiguilles d'une montre, soit dans le sens inverse des aiguilles d'une montre. On choisit une orientation, par exemple dans le sens des aiguilles d'une montre et à chaque couple $\{s_k, t_k\}$ on associe deux couples : $\{s_k, t_k\}$ dont le flot F_k correspondant va de s_k à t_k et $\{s_{k+K}, t_{k+K}\}$ dont le flot F_{k+K} correspondant va de t_k à s_k de manière à respecter l'orientation.

Soit $\mathfrak{R} = (V, A)$ un anneau orienté. Soit p_k l'unique chemin de s_k à t_k, soit f_k le flot routé sur p_k, $k \in \{1, \ldots, K\}$, et soit c_a, $a \in A$, une variable binaire telle que $c_a = 1$ si l'arc a appartient à la coupe, et $c_a = 0$ sinon. IMCP et IMFP peuvent être formulés comme deux programmes linéaires en nombres entiers :

$$\text{(P-IMFP)} \quad \left| \begin{array}{lll} \max & \displaystyle\sum_{k=1}^{K} f_k & \\ \text{s. c.} & \displaystyle\sum_{k \text{ s.c. } a \in p_k} f_k \leq u_a & \forall a \in A \\ & f_k \in \mathbb{N} & \forall k \in \{1, \ldots, K\} \end{array} \right.$$

$$\text{(P-IMCP)} \quad \left| \begin{array}{lll} \min & \displaystyle\sum_{a \in A} u_a c_a & \\ \text{s. c.} & \displaystyle\sum_{a \in p_k} c_a \geq 1 & \forall k \in \{1, \ldots, K\} \\ & c_a \in \{0, 1\} & \forall a \in A \end{array} \right.$$

Notons que les relaxations continues de ces programmes linéaires sont duales ; c'est également le cas dans les graphes non orientés (Garg et al. (1996)).

En général, il existe un saut de dualité entre les valeurs optimales de IMCP et de IMFP, excepté dans certains cas particuliers, comme les arbres orientés (Costa et al. (2003)).

C'est également le cas dans les anneaux. Prenons par exemple l'anneau orienté avec trois sommets v_1, v_2, v_3, trois arcs de capacité/poids égal 5 et trois paires $\{s_k, t_k\}$ tels que $s_1 = t_2 = v_1$, $s_2 = t_3 = v_2$ et $s_3 = t_1 = v_3$. La valeur optimale de IMCP et IMFP pour cette instance sont respectivement de 10 et 7.

Dans la suite de cette section, nous montrons dans la section 2.1.1 comment simplifier et réduire l'anneau. Ensuite, dans la section 2.1.2, nous montrons que IMCP peut être résolu dans les anneaux en utilisant un algorithme polynomial pour les réseaux sous forme de chaînes (Hassin et Tamir (1991)), et en utilisant des idées de Bartholdi et al. (1980), nous construisons un algorithme polynomial pour résoudre IMFP dans les anneaux ; nous prouvons également que le saut d'intégrité pour ce problème est strictement plus petit que 1. Enfin, dans la section 2.1.3 nous proposons un algorithme en $O(n)$ pour résoudre IMFP et IMCP dans les anneaux avec capacité uniforme, i.e., dans les anneaux où toutes les capacités sont égales.

2.1.1 Simplifications et réductions

Nous pouvons effectuer quelques simplifications sur l'anneau orienté obtenu. Si un chemin p_k, d'une source s_k à un puits t_k est inclus dans un autre chemin p_l, d'une source s_l à un puits t_l, alors nous pouvons supprimer le couple $\{s_l, t_l\}$ de l'anneau. En effet, il est toujours plus intéressant de router le flot F_k que le flot F_l car tout ce qui peut circuler sur p_l peut circuler sur p_k et le nombre d'arcs de p_k est inférieur à celui de p_l, de plus si le chemin p_k est coupé, alors le chemin p_l le sera également, donc on peut ne garder que le couple $\{s_k, t_k\}$.

Une autre simplification du problème consiste à réduire n'importe quelle suite d'arcs (ou partie de chemin) qui ne comporte ni source ni puits en un seul arc dont le poids sera égal au poids minimal des arcs de ce chemin. En effet, pour couper un tel chemin, l'arc de poids minimal est toujours meilleur que les autres ; de plus, cet arc de poids minimal est celui qui limite tout flot routé sur ce chemin. Les autres arcs de ce chemin ne jouent donc aucun rôle que ce soit dans le flot maximal ou dans la coupe minimale : ils peuvent être retirés du graphe.

Nous pouvons également supprimer les arcs sur lesquels ne passe aucun flot.

Considérons deux arcs adjacents (u, v) et (v, w) tels que seule une source s_k se trouve en v. Si la capacité de (u, v) est plus grande ou égale que celle de (v, w) alors l'arc (u, v) peut être contracté en un seul sommet ($u = v$), puisque (v, w) pourra être sélectionné à la place de (u, v) dans toute multicoupe minimale et puisque (v, w) est plus contraignant pour le flot que (u, v). Nous plaçons ainsi s_k sur u. De manière analogue, s'il existe un puits sur v et si la capacité de (u, v) est plus petite ou égale que celle de (v, w) alors (v, w) peut être contracté en un seul sommet.

Ces simplifications et réductions doivent être itérées récursivement jusqu'à ce qu'aucune réduction ou simplification ne soit possible. Comme une paire ou un arc est supprimé à chaque réduction, l'anneau peut être réduit en $O(K + n)$.

Dans la suite nous considérerons des anneaux orientés (dans le sens des aiguilles d'une montre) $\mathfrak{R} = (V, A)$ sur lesquels chaque sommet possède une source et/ou un puits et la liste $\mathfrak{L}$ est un ensemble propre, i.e., qu'aucun chemin p_i n'est

un sous-chemin p_j pour $i \neq j$. Les sommets et les arcs sont numérotés de 1 à n et les sources et les puits sont numérotés de 1 à K, $K \leq n$, dans le sens des aiguilles d'une montre.

2.1.2 Cas général

Nous donnons ici un algorithme polynomial permettant de résoudre IMCP et IMFP.

Proposition 1. *IMCP peut être résolu en $O\left(n^2\right)$ dans les anneaux.*

Preuve 1. *Le chemin p_i de s_i à t_i doit contenir au moins un arc de la coupe, pour tout i. Nous avons ainsi à résoudre $O(n)$ IMCP instances : la j^{eme} instance est obtenue en supprimant le j^{eme} arc de p_k, qui est le plus court chemin parmi les p_i, et peut être résolue en $O(K+n)$ Hassin et Tamir (1991) ($K \leq n$ ici). Le coût de la solution pour la j^{eme} instance inclut le coût de la suppression du j^{eme} arc. Nous conservons la meilleure solution parmi les $O(n)$ solutions obtenues : la réduction optimale s'effectuant en $O(K+n)$, la complexité globale est donc $O(n^2)$.*

Theorème 1. *IMFP peut être résolu en temps polynomial dans les anneaux.*

Preuve 2. *Nous effectuons une recherche binaire sur la valeur de $\sum_{i=1}^{K} f_i \leq Ku_{max}$, où u_{max} est la capacité maximale des arcs de l'anneau. Une fois la valeur de cette somme fixée et égale à un entier F, nous pouvons ajouter la contrainte (Λ_0) : $\sum_{i=1}^{K} f_i = F$ à $(P - IMFP)$. Notons qu'il existe un multiflot entier de valeur F' quel que soit $F' \leq F$ si et seulement s'il existe un multiflot entier de valeur F. Notre problème est désormais un problème de décision : existe-t-il une solution réalisable ? La matrice des contraintes de $(P - IMFP)$ contient deux types de lignes : soit les 1 sont consécutifs soit ils ne le sont pas, et dans ce cas les 0 sont consécutifs (cette propriété est connue sous le nom de* propriété des 1 circulaires*). Pour chaque contrainte (Λ) impliquant des flots non consécutifs (i.e., où les 1 sont non consécutifs), nous définissons une nouvelle contrainte $(\Lambda') \leftarrow (\Lambda_0) - (\Lambda)$ et supprimons (Λ) : dans (Λ'), tous les 1 sont consécutifs. Nous obtenons ainsi un problème équivalent où la matrice des contraintes $0-1$ vérifie la propriété des 1 consécutifs (matrice d'intervalle) et est donc totalement unimodulaire. De plus, le vecteur des membres droits des contraintes est à valeurs entières, il existe donc une solution entière s'il existe une solution fractionnaire.*

Notons que nous n'avons pas besoin de transformer les lignes où les 1 sont déjà consécutifs, alors que dans Bartholdi et al. (1980) certaines de ces lignes sont modifiées. De plus, notre approche implique que :

Corollaire 1. *Dans les anneaux, le saut d'intégrité absolu entre la valeur d'un multiflot maximum fractionnaire et la valeur d'un multiflot maximum entier est strictement inférieur à 1.*

Preuve 3. *L'existence d'une solution fractionnaire de valeur F^* implique l'existence d'une solution fractionnaire de valeur $\lfloor F^* \rfloor$. Appliquons maintenant la transformation des contraintes proposée dans la preuve du théorème 1 avec (Λ_0) : $\sum_{i=1}^{K} f_i = \lfloor F^* \rfloor$. Le programme (équivalent) qui en résulte a une matrice des contraintes totalement unimodulaire avec un vecteur des membres droits à valeurs entières. Cela implique l'existence d'une solution entière de valeur $\lfloor F^* \rfloor$.*

Cette aproche permet également de prouver la résolution en temps polynomial de tout programme linéaire en nombres entiers tel que :

- la matrice des contraintes $0-1$ satisfait la propriété des 1 circulaires,
- tous les coefficients de la fonction objectif à maximiser sont égaux et non négatifs,
- et toutes les contraintes sont de type “packing“.

2.1.3 Cas uniforme

Nous proposons ici un algorithme en $O(n)$ pour résoudre IMCP et IMFP dans les anneaux (réduits) uniformes, i.e., où tous les arcs ont la même capacité, notée U. Notons que dans un anneau uniforme, nous pouvons supposer sans perte de généralité qu'il existe exactement une source et un puits en chaque sommet : cela provient des réductions de la section 2.1.1. En fait, si le sommet v ne possède qu'un seul terminal (source ou puits), alors un des deux arcs incidents à v est contracté. Le nombre de terminaux K est ainsi égal au nombre de sommets n, et tous les chemins p_k ont la même longueur, notée L. Les arcs et les terminaux sont numérotés comme précédemment et L flots successifs passent sur chaque arc, en supposant que f_1 suit f_n.

Remarque 1. *Les valeurs optimales des relaxations continues de $(P-IMFP)$ et $(P-IMCP)$ sont égales à $\frac{nU}{L}$ dans des anneaux uniformes de capacité U, puisque les solutions optimales de ces deux problèmes sont obtenues par $f_k = \frac{U}{L}$ pour tout $k \in \{1,\dots,n\}$ et $c_a = \frac{1}{L}$ pour tout $a \in A$, respectivement.*

Theorème 2. *IMCP peut être résolu en $O(n)$ dans les anneaux uniformes.*

Preuve 4. *En sommant les contraintes de coupe de $(P-IMCP)$ et en utilisant l'intégrité des variables c_a, nous obtenons $\sum_{a\in A} c_a \geq \lceil \frac{n}{L} \rceil$, et donc $\sum_{a\in A} u_a c_a \geq \lceil \frac{n}{L} \rceil U$. On peut alors définir une multicoupe de valeur $\lceil \frac{n}{L} \rceil U$ (et donc optimale). Pour $j \in \{1,\dots,n\}$, $c_{a_j} = 1$ si $j \in \{1+pL, p \in \{0,\dots,\lceil \frac{n}{L} \rceil - 1\}\}$, et $c_{a_j} = 0$ sinon. Cela fournit une solution réalisable pour la multicoupe avec $\lceil \frac{n}{L} \rceil$ arcs en $O(n)$.*

Considérons $\beta = nU - L\lfloor \frac{nU}{L} \rfloor$ et $\delta = \lfloor \frac{nU}{L} \rfloor - n\lfloor \frac{U}{L} \rfloor$. On peut noter que $0 \leq \beta < L$ et $0 \leq \delta < n$, car β est le reste de la division Euclidienne de nU par L, et δ est le reste de la division Euclidienne de $\lfloor \frac{nU}{L} \rfloor$ par n.

Algorithme 1 Multiflot_entier_maximal_anneaux_uniformes

Sortie : Un multiflot entier $\hat{f} = \left(\hat{f}_j, j \in \{1,\dots,n\}\right)$ tel que $\sum_{j=1}^{n} \hat{f}_j = \lfloor \frac{nU}{L} \rfloor$

for $j = 1$ to n **do**
    $\hat{f}_j := \lfloor \frac{U}{L} \rfloor$;
end for
$j := 1$;
for $i = 1$ to δ **do**
    $\hat{f}_j := \hat{f}_j + 1$;
    if $j + L < n + 1$ **then**
        $j := j + L$;
    else
        $j := j + L - n$;

end if
end for

Theorème 3. *IMFP peut être résolu en $O(n)$ dans les anneaux uniformes en appliquant l'algorithme 1.*

Preuve 5. *Montrons tout d'abord qu'aucune contrainte de capacité n'est violée par $\hat{f}$. Dans la seconde boucle de l'algorithme 1, nous ajoutons un flot unitaire à chaque arc au plus $\lceil \frac{\delta L}{n} \rceil$ fois (puisque nous tournons autour d'un anneau de n arcs en effectuant δ sauts de longueur L). Comme L flots passent sur chaque arc, la quantité de flots routée sur chaque arc est au plus de $\lfloor \frac{U}{L} \rfloor L + \lceil \frac{\delta L}{n} \rceil \leq \lfloor \frac{U}{L} \rfloor L + \left\lceil \frac{\delta L}{n} + \frac{\beta}{n} \right\rceil = \lfloor \frac{U}{L} \rfloor L + \lceil \frac{L}{n} \lfloor \frac{nU}{L} \rfloor - L \lfloor \frac{U}{L} \rfloor + U - \frac{L}{n} \lfloor \frac{nU}{L} \rfloor \rceil = \lfloor \frac{U}{L} \rfloor L + \lceil U - \lfloor \frac{U}{L} \rfloor L \rceil = \lfloor \frac{U}{L} \rfloor L + (U - \lfloor \frac{U}{L} \rfloor L)$, et ainsi au plus de U. Cette solution a comme valeur $\sum_i \hat{f}_i = n \lfloor \frac{U}{L} \rfloor + \delta = \lfloor \frac{nU}{L} \rfloor$, et d'après la remarque 1, est optimale. Enfin, la complexité de l'algorithme 1 est $O(n)$, puisque $0 \leq \delta < n$.*

2.1.4 Conclusion

Nous avons montré que les problèmes de multicoupes minimales et de multiflots maximaux en nombres entiers dans les anneaux pouvaient être résolus en temps polynomial ; et nous avons proposé une procédure en temps linéaire pour résoudre ces deux problèmes dans les anneaux avec capacité uniforme.

2.2 Réduction de graphes pour la segmentation d'images

Dans cette section, nous nous intéressons à la segmentation d'images via la résolution de problèmes de flots maximaux et coupes minimales. Les graph cuts, terme employé en traitement d'images pour désigner les approches à base de flot maximal / coupe minimale, sont des méthodes d'optimisation qui sont utilisées pour résoudre de nombreux problèmes en traitement d'images et vision par ordinateur. L'approche par minimisation d'énergie et l'optimisation discrète sont devenues des domaines importants en vision par ordinateur. Cela est notamment dû à la capacité de pouvoir calculer efficacement le Maximum a Posteriori (MAP) de Champs de Markov aléatoires (CMA) issus de nombreux problèmes à l'aide de méthodes comme les graph cuts. Les premiers travaux Greig et al. (1989) dans ce domaine datent de la fin des années 80, mais ces méthodes ont connu récemment un développement rapide avec l'arrivée d'algorithmes rapides et adaptés aux problèmes de traitement d'images (Boykov et Kolmogorov (2004)). Parallèlement, les avancées technologiques dans le domaine de l'acquisition des images ont augmenté de manière importante à la fois le volume et la diversité des données à traiter. Les graphes induits par des images volumineuses peuvent contenir jusqu'à des milliards de sommets et ne peuvent donc pas être stockés en mémoire vive, ce qui limite donc fortement l'intérêt des approches globales comme les graph cuts.

Dans le contexte de la segmentation d'images, les méthodes d'optimisation telles que les graph cuts sont incapables de résoudre de tels volumes de données à cause des besoins mémoire requis. Pour pallier ce problème, certains auteurs ont proposé des méthodes exactes (Strandmark et Kahl (2010), Lempitsky et Boykov

(2007), Delong et Boykov (2008)) pour réduire ces graphes. Dans Delong et Boykov (2008), les auteurs présentent un algorithme de flot maximum parallèle avec une accélération quasi linéaire en fonction du nombre de processeurs. Cependant, l'algorithme de flot maximum reste moins efficace sur des graphes de petite taille. Dans Lempitsky et Boykov (2007), les auteurs font évoluer une bande autour de l'objet qui s'agrandit lorsque la coupe minimum rentre en contact avec le bord de la bande. Ce processus est itéré jusqu'à ce que la bande ne soit plus déformée. L'algorithme converge rapidement mais aucune borne sur la taille de la bande n'est donnée. Bien que les bandes générées soient relativement minces pour le problème considéré, rien ne garantit qu'elles le soient aussi dans le cas de la segmentation. Dans Strandmark et Kahl (2010), l'approche utilisée est différente : le problème est cette fois-ci décomposé en sous-problèmes résolus indépendemment en parallèle ou en distribué. L'optimalité sur la solution est garantie par décomposition, ou plus précisément, les solutions des sous-problèmes sont contraintes à être égales sur une bande de chevauchement commune.

Par ailleurs, des heuristiques basées par exemple sur des schémas multi-résolution (Sinop et Grady (2006), Kohli et al. (2010)) ont été proposées. Le principe est de construire un graphe dans une bande étroite contrainte par la segmentation issue d'une image sous-échantillonnée. L'algorithme réduit considérablement le temps de calcul et la consommation mémoire mais ne parvient généralement pas à conserver les détails. Bien que réduit dans Sinop et Grady (2006), cet inconvénient reste vrai pour des images peu contrastées. L'approche est raffinée dans Kohli et al. (2010) en introduisant une mesure de confiance sur chaque pixel. L'approche permet d'obtenir des bandes assez fines, tout en étant très proche de la solution optimale. Cependant, aucune expérience n'est présentée pour de grandes images.

Une autre approche s'appuie sur les graphes d'adjacence (Li et al. (2004), Cigla et Alatan (2008)). Le principe est de pré-segmenter l'image à partir d'un outil de segmentation bas niveau (par exemple, ligne de partage des eaux (Li et al. (2004)) ou mean shift (Cigla et Alatan (2008)), de construire un graphe d'adjacence où chaque noeud correspond à une région et de calculer un flot maximum dans ce graphe. Là encore, le temps de calcul ainsi que la consommation mémoire sont grandement diminués. Néanmoins, les résultats dépendent de la sensibilité au bruit de l'outil de segmentation bas niveau. De plus, les résultats se dégradent rapidement à mesure que la taille des régions de la pré-segmentation augmente.

Dans ce rapport, nous décrivons une nouvelle stratégie de réduction. Le graphe est progressivement construit en ajoutant les noeuds qui satisfont localement une condition donnée. À l'instar de Sinop et Grady (2006) et Kohli et al. (2010), les noeuds sont situés dans une bande étroite autour des contours de l'objet à segmenter. Empiriquement, les solutions obtenues après réduction sont identiques à celles obtenues sans réduction. Grâce à un algorithme rapide, la complexité initiale de la condition est calculée en temps constant, hors bords de l'image. Les expériences montrent que le temps requis par notre algorithme est parfois compensé par le temps qui serait nécessaire à l'allocation du graphe ainsi qu'au calcul du flot maximum sur le graphe réduit.

Le reste de cette section est organisé comme suit. En section 2.2.1, nous rappelons le cadre général des graph cuts. Après avoir introduit quelques définitions, nous détaillons notre stratégie de réduction en section 2.2.2 et présentons des expériences pour segmenter des images multidimensionnelles en niveaux de gris et couleur.

2.2.1 Principe des graph cuts

Soit $I : \mathcal{P} \subset \mathbb{Z}^d \rightarrow \mathbb{R}^m$ ($d > 0, m > 0$) une image où chaque point $p \in \mathcal{P}$ correspond à un pixel de valeur I_p. Habituellement, $\mathcal{P}$ correspond à un carré lorsque $d = 2$, un cube lorsque $d = 3$ et à un cube doté d'un intervalle de temps lorsque $d = 4$. On définit une segmentation binaire comme une application u affectant à chaque pixel $p \in \mathcal{P}$ la valeur 0 (fond) ou 1 (objet) et l'on écrit $u \in \{0,1\}^{\mathcal{P}}$. Le problème de segmentation peut alors être résolu efficacement en minimisant un CMA de la forme (Boykov et Jolly (2001)) :

$$E(u) = \beta \cdot \sum_{p \in \mathcal{P}} E_p(u_p) + \sum_{(p,q) \in \mathcal{N}} E_{p,q}(u_p, u_q), \tag{2.1}$$

avec $u \in \{0,1\}^{\mathcal{P}}$ et pour $\beta \in \mathbb{R}^+$. Le système de voisinage $\mathcal{N}$ contient toutes les paires de points voisins $(p,q) \in \mathcal{N}$. Par la suite, nous utiliserons les voisinages suivants :

$$\mathcal{N}_0 = \{(p,q) \mid p,q \in \mathcal{P}^2, \textstyle\sum_{i=1}^{d} |q_i - p_i| = 1\} \quad \text{ou,}$$
$$\mathcal{N}_1 = \{(p,q) \mid p,q \in \mathcal{P}^2, |q_i - p_i| \leq 1 \; \forall 1 \leq i \leq d\}$$

où p_i désigne la i$^{\text{ème}}$ coordonnée du point p et $|.|$ désigne la valeur absolue. Par la suite, les termes « connexité 0 » et « connexité 1 » indiqueront respectivement l'utilisation d'un voisinage de type $\mathcal{N}_0$ et $\mathcal{N}_1$. Dans (2.1), le terme de région $E_p(.)$ favorise l'appartenance de chaque pixel au fond ou à l'objet. Ce terme est calculé en fonction de I ainsi que, si le modèle est interactif, des graines objet $\mathcal{O} \subset \mathcal{P}$ et fond $\mathcal{B} \subset \mathcal{P}$. Le terme de frontière $E_{p,q}(.)$ pénalise les pixels voisins p et q ayant des étiquettes différentes et favorise dans la segmentation les contours liés à un fort gradient. Notons que cette détection peut être améliorée en incluant d'autres caractéristiques telles que des informations sur la texture ou la direction du gradient.

Selon Kolmogorov et Zabih (2004), lorsque les termes $E_{p,q}(.)$ sont sous-modulaires, le minimiseur de (2.1) peut être obtenu en calculant une coupe minimum / flot maximum dans un graphe orienté et pondéré $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$ où $\mathcal{V} = \mathcal{P} \cup \{s,t\}$, $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ désigne l'ensemble des arcs et où chaque arc est muni d'une capacité $c : \mathcal{E} \rightarrow \mathbb{R}^+$. Les noeuds s et t correspondent respectivement aux terminaux « objet » et « fond ». L'ensemble des arcs $\mathcal{E}$ est divisé en deux ensembles disjoints $\mathcal{E}_n$ et $\mathcal{E}_t$ correspondant respectivement aux n-links (arcs reliant deux noeuds de $\mathcal{P}$) et aux t-links (arcs reliant un noeud à s ou t). Lorsque la coupe minimum est calculée, on fixe $u_p = 1$ si un noeud p est connecté à s et $u_p = 0$ si p est connecté à t.

2.2.2 Réduction

Définitions et notations

Comme nous l'avons évoqué précédemment, la consommation mémoire des graph cuts pour la segmentation d'images haute-résolution peut avoir un coût prohibitif. À titre d'exemple, l'algorithme de flot maximum de Boykov et Kolmogorov v2.2 alloue $24|\mathcal{P}| + 14|\mathcal{E}_n|$ octets (Boykov et Kolmogorov (2004)) (où $|.|$ désigne le cardinal d'un ensemble). Pour une quantité fixée de RAM fixée à 2 Go, on observe que la taille maximale d'une image décroît rapidement à mesure que la dimension d croît (voir Tableau 2.1).

Sur l'image de droite de la figure 2.1, nous représentons le flot passant sur les t-links. Les pixels les plus clairs (respectivement les plus foncés) indiquent qu'une

	Connexité 0	Connexité 1
2D	6426	4459
3D	319	219
4D	68	45

TABLE 2.1 – *Taille maximale d'une image carrée pour que le graphe associé puisse tenir en mémoire en fonction de d et de la connexité, pour une quantité de mémoire fixée.*

quantité de flot est routée de s à p (respectivement de p à t). Nous observons donc que la plupart des noeuds dans un graphe sont inutiles du point de vue du flot maximum car ils ne sont traversés par aucun flot.

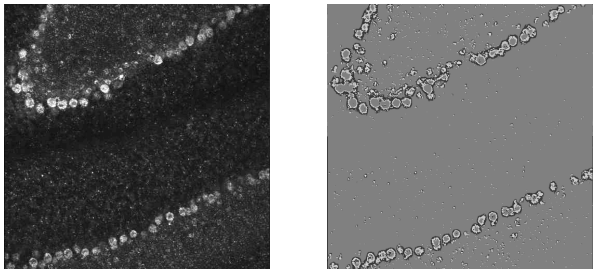


FIGURE 2.1 – *Illustration du flot passant par les t-links (droite) pour segmenter une image 2D (gauche).*

Partant de ce constat, on souhaiterait donc pouvoir extraire le plus petit sous-graphe $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', c)$ à partir de $\mathcal{G}$, tout en gardant une solution u' identique (ou très proche) de u. En d'autres termes, on souhaite maximiser le taux de réduction $\rho = 1 - \frac{\sharp\mathcal{V}'}{\sharp\mathcal{V}}$ sous la contrainte $u \simeq u'$. Cela correspond à un problème d'optimisation que nous n'essayerons pas de résoudre puisque la méthode pour déterminer $\mathcal{G}'$ doit ici rester rapide.

Avant de détailler notre stratégie pour construire $\mathcal{G}'$, nous avons besoin d'introduire quelques définitions. En accord avec Kolmogorov et Zabih (2004), on considère (sans perte de généralité) qu'un noeud n'est connecté qu'à au plus un terminal dans $\mathcal{G}$:

$$(s,p) \in \mathcal{E}_t \Rightarrow (p,t) \notin \mathcal{E}_t, \qquad p \in \mathcal{P}.$$

On contracte la notation des capacités sur les t-links connectés à un noeud $p \in \mathcal{P}$:

$$c(p) = c(s,p) - c(p,t).$$

Considérons maintenant une fenêtre carrée B de taille $(2r+1)$ $(r > 0)$ centrée à l'origine. On désigne par $\widetilde{B}_p$ la translation de B à une position $p \in \mathcal{P}$:

$$\widetilde{B}_p = \{b + p \mid b \in B\}.$$

Pour un ensemble $Z \subset \mathcal{P}$, on désigne aussi par $\widetilde{Z}_B$ la dilatation de Z par l'élément structurant B :

$$\widetilde{Z}_B = \{p + b \mid p \in Z,\ b \in B\} = \bigcup_{p \in Z} \widetilde{B}_p.$$

L'idée pour construire $\mathcal{G}'$ est alors la suivante : supprimer les noeuds d'un ensemble $Z \subset \mathcal{P}$ dont tous les noeuds sont reliés à s (respectivement à t) tel que la quantité maximale de flot qui peut entrer (respectivement sortir) de $\widetilde{Z}_B \setminus Z$ par les t-links est plus grande que la quantité de flot qui peut sortir (respectivement entrer) de $\widetilde{Z}_B$ par les n-links (voir Figure 2.2). La construction d'un tel ensemble Z est réalisée

en testant chaque noeud $p \in \mathcal{P}$. Par conséquent, les noeuds se trouvent localisés autour des contours de l'objet à segmenter. En prenant pour hypothèse que les capacités sur les n-links sont inférieures ou égales à un (ce qui reste vrai pour la plupart des modèles d'énergie en segmentation), nous proposons d'utiliser la condition suivante (légèrement plus restrictive) pour tester chaque pixel $p \in Z$:

$$\begin{cases} \forall q \in \widetilde{B}_p,\ c(q) \geq +\delta & \text{ou} \\ \forall q \in \widetilde{B}_p,\ c(q) \leq -\delta, \end{cases} \tag{2.2}$$

où $\delta = \frac{P(B)}{(2r+1)^2-1}$ et P(B) définit le périmètre de B :

$$P(B) = \max(\sharp\{(p,q)\ :\ p \in B, q \notin B \text{ et } (p,q) \in \mathcal{N}\}, \\ \sharp\{(p,q)\ :\ p \in B, q \notin B \text{ et } (p,q) \in \mathcal{N}\}).$$

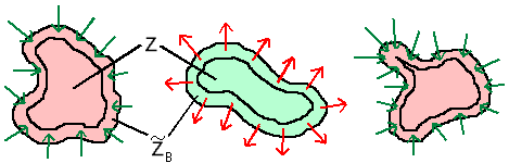


FIGURE 2.2 – *Principe de la réduction : les zones et flèches rouges (resp. vertes) indiquent le flot qui peut rentrer (resp. sortir) de $\widetilde{Z}_B$. Les nœuds de Z sont supprimés puisque chaque nœud $p \in \mathcal{P}$ vérifie la condition* (2.2)*. Les nœuds restants sont typiquement localisés dans la bande étroite $\widetilde{Z}_B \setminus Z$.*

Clairement, pour un noeud $p \in \mathcal{P}$ satisfaisant (2.2), p est seulement relié à s (respectivement à t) et le flot qui peut rentrer (respectivement sortir) à travers les t-links dans $\widetilde{B}_p \setminus \{p\}$ suffit à saturer les n-links sortant (respectivement entrant) de $\widetilde{B}_p$. Le noeud p devient inutile et peut alors être supprimé de $\mathcal{G}$. La condition (2.2) permet d'exhiber un algorithme simple pour construire $\mathcal{G}'$ en évaluant (2.2) de manière incrémentale pour atteindre une complexité $O(\sharp\mathcal{P})$ (hors bords de l'image), indépendante du rayon de la fenêtre.

Les expériences présentées dans Lermé et al. (2010) confirment la dépendance intuitive entre le taux de réduction et les paramètres du modèle. Par exemple, puisque les capacités contractées sont multipliées par β qui est présent dans (2.1), il est facile de voir que la condition (2.2) est satisfaite plus difficilement lorsque β décroît (forte régularisation). Ainsi, l'algorithme a besoin d'un rayon de fenêtre plus grand pour faire décroître δ et génère alors des bandes plus larges. Inversement, lorsque β augmente, le graphe réduit est constitué de bandes plus étroites.

Expériences

Dans cette section, nous comparons les performances des graph cuts et de notre algorithme en terme de temps d'exécution et d'utilisation mémoire pour segmenter des images à l'aide d'un modèle Boykov/Jolly Boykov et Jolly (2001) en connexité 1 (voir Tableau 2.2 et Figure 2.3). Les temps d'exécution sont mesurés sur une moyenne de dix lancements et incluent le temps de construction du graphe, le calcul de la coupe minimum et la construction de la solution. La machine utilisée pour ces expériences est un Athlon Dual Core 6000+ cadencé à 3 GHz avec 2 Go de RAM.

Avec un taux de réduction moyen du nombre de noeuds de 79.42%, les résultats présentés dans le Tableau 2.2 sont encourageants. Dans le cas de

l'image « cellules-souris », le taux de réduction est plus faible car les données sont très bruitées. Cela implique donc une valeur de β plus faible pour limiter l'attache aux données dans (2.1). Pour toutes les images, il est important de noter que la différence de flot maximum sur le graphe non réduit et le graphe réduit est toujours égale à zéro. Il en est de même pour la différence concernant le nombre de pixels appartenant à la partie objet dans la segmentation. Le Tableau 2.2 montre également que le rayon optimal qui maximise le taux de réduction est toujours égal à un pour des images contrastées comme c'est le cas ici.

Excepté pour une instance, les graph cuts standards sont incapables de segmenter les volumes 2D+t et 3D. Notre approche permet de les segmenter et ce, en moins d'une minute. Pour certaines images, notre méthode est même parfois plus rapide mais devient moins compétitive en couleur. En effet, bien que la condition (2.2) soit calculée en $O(1)$, le temps passé sur les bords croît progressivement avec r. Cet effet est accentué lorsque le nombre de canaux augmente.

	Nom de l'image	Taille	Graph cuts standards		Réduction		ρ^* (%)	r^*
			Temps	Mémoire	Temps	Mémoire		
2D	étoile-mer-c	481 ×321	0.22	27.59	0.27	8.31	76.76	1
	livre	3012 ×2048	9.19	1105.92	8.94	94.68	91.45	1
	symbole-viking	660 ×740	0.71	87.41	0.70	4.01	94.62	1
2D+t	interview-c	320 ×240 ×203	**PM**	8079.36	38.27	950.25	88.74	1
	avion-c	492 ×276 ×180	**PM**	12677.12	55.95	1198.08	90.28	1
	discours-c	370 ×276 ×190	**PM**	10065.92	49.85	1423.36	85.23	1
	cellule-fluo-c	478 ×396 ×121	**PM**	11868.16	61.68	1546.24	85.94	1
3D	ct-thorax	245 ×245 ×151	**PM**	4689.92	22.95	950.25	81.54	1
	cellules-souris	230 ×230 ×57	10.90	1546.24	16.04	1423.36	20.88	1
	cerveau+bruit-3%	181 ×217 ×181	**PM**	3676.16	19.30	950.25	78.81	1

TABLE 2.2 – *Comparaison temps de calcul (s) / mémoire (Mo) entre notre méthode et les graph cuts standards pour la segmentation d'images avec un modèle de Boykov/Jolly en connexité 1. Les images dont le nom est suffixé par « c » sont en couleur tandis que les autres sont en niveaux de gris. Le message* **PM** *(« Problème Mémoire ») indique que le graphe n'a pas pu être alloué. r^* correspond au rayon de la fenêtre pour lequel ρ est maximum (ici noté ρ^*).*

2.2.3 Conclusion

Dans cette section, nous avons présenté une nouvelle méthode pour réduire exactement des graphes dans le contexte de la segmentation d'images. Les expériences montrent des résultats prometteurs avec un taux de réduction moyen entre 66 et 79%, selon le modèle d'énergie utilisé. La réduction a été montrée exacte théoriquement, mais pour une condition légèrement plus forte que celle mentionnée. Nous investiguons actuellement des idées similaires dans un contexte multi-étiquettes et nous avons également proposé une adaptation de cette approche pour effectuer un débruitage de l'image en même temps que la réduction du graphe.

2.3 EXTRACTION DE MOTIFS BRUITÉS VIA LA RECHERCHE DE FLOTS MAXIMAUX

Nous abordons dans cette section le problème difficile de l'extraction de motifs contraints dans des données booléennes bruitées. La fouille de motifs ensemblistes contraints dans des matrices binaires consiste à rechercher des rectangles de 1 dans une matrice de données à valeurs dans {0,1} qui satisfont un ensemble de

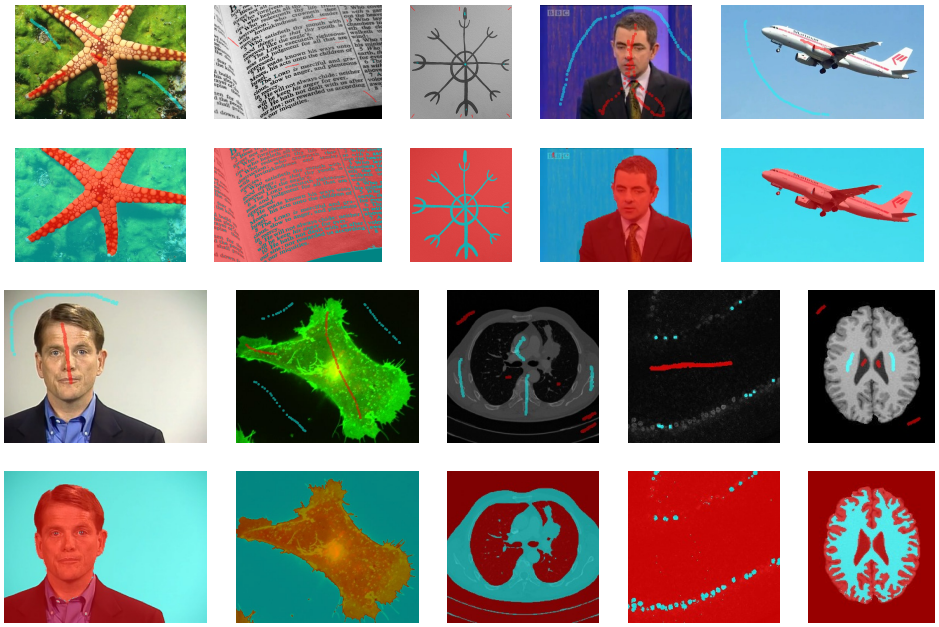


FIGURE 2.3 – *Résultats obtenus avec un modèle Boykov/Jolly en connexité 1. Les segmentations (lignes 2 et 4) ainsi que les graines (lignes 1 et 3) sont superposées par transparence aux images originales.*

contraintes (fréquence, aire, etc). Cependant, dans des applications réelles les données sont souvent bruitées. Un des effets du bruit est de "pulvériser" un motif pertinent en un ensemble de sous-motifs recouvrants et peu pertinents, entraînant une explosion du nombre de résultats. Nous proposons dans cette section des approches heuristiques originales qui combinent des algorithmes de fouille de données et des algorithmes de graphes de type flot maximal/coupe minimale pour rechercher des sous graphes denses maximaux qui peuvent se recouvrir dans un graphe biparti pondéré et augmenté associé à la matrice des données.

2.3.1 Formulation du problème

Soient A un ensemble fini d'attributs, O un ensemble fini d'observations ou d'objets, et $R \subseteq O \times A$ une relation binaire de O vers A. On appelle contexte formel D le triplet (O, A, R). Notre but dans ce travail consiste à rechercher des régions denses maximales $r = (O_1, A_1)$ qui peuvent se recouvrir dans des contextes booléens $D = (O, A, R)$ (applications en biologie (Pizzuti et al. (2012), Hu et al. (2008), Borgelt et al. (2011)), extraction de communautés (Leon-Suematsu et Yuta (2010), Traud et al. (2008), Newman (2004), Radicchi et al. (2004))...).

Notre principal objectif sera de rechercher des motifs résultats qui vérifient une contrainte de support minimal relatif σ et qui limitent la proportion d'exceptions en ajoutant une contrainte de densité minimale δ.

Nous représentons chaque contexte booléen $D = (O, A, R)$ par un graphe biparti $G = (V_1, V_2, E)$. Chaque observation $o_i \in O$ est représentée par un sommet $v_i \in V_1$ et chaque attribut $a_j \in A$ est représenté par un sommet $v_j \in V_2$ (voir figure 2.4). Une arête est ajoutée entre deux sommets dans le graphe si l'intersection entre la ligne et la colonne correspondantes aux sommets est égale à 1 dans la matrice des données. Comme les attributs et les observations sont deux ensembles disjoints, le graphe obtenu est biparti.

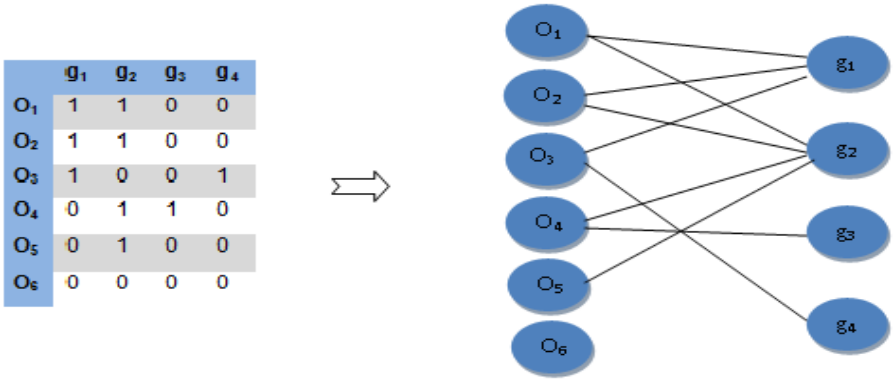


FIGURE 2.4 – *Représentation d'un jeu de données par un graphe*

L'extraction des régions denses dans un jeu de données revient à extraire des sous graphes denses dans le graphe correspondant à la matrice des données.

2.3.2 Principe

Notre objectif est de rechercher efficacement des régions denses en 1 maximales dans des contextes booléens susceptibles de contenir du bruit. Nous proposons ici une méthode itérative basée sur deux étapes. Une première étape qui consiste en une extraction d'un motif graine composé d'attributs et de densité 1 et une deuxième étape qui construit une région maximale qui inclut le motif graine et vérifiant des contraintes de densité et de support minimal. Pour le calcul des motifs graines, nous introduisons un contexte supplémentaire D_{mg}, utilisé spécifiquement pour le calcul des motifs graines. D_{mg} est initialement égal à la matrice des données initiale D.

Le schéma itératif de notre approche est le suivant :

1. Extraire dans D_{mg} un motif graine m_{graine} composé d'un ensemble d'attributs puis
2. Rechercher dans D une région r dense et maximale qui inclut le motif graine m_{graine} et mettre à jour D_{mg}
3. Si $D_{mg} \neq \emptyset$, aller en 1.

2.3.3 Extraction des sous-graphes ou régions denses

Construction des graphes et calcul des poids

L'approche que nous proposons est basée sur des algorithmes de flot maximal et coupe minimale. La construction des graphes se fait de façon à ce que les sous-graphes résultats vérifient les contraintes de densité et de support minimal exprimées sous forme de capacités affectées aux arcs dans les graphes.

À partir d'une matrice discrète à deux valeurs, décrite précédemment, nous extrayons un motif graine m_{graine} composé d'un sous ensemble d'attributs. Dans l'optique de rechercher une région dense et maximale qui l'inclut, nous construisons dans un premier temps un graphe valué et augmenté d'un sommet source s et d'un sommet destination t correspondant à m_{graine} (Algorithme 2) ; puis une coupe minimale est calculée dans le graphe.

Les capacités affectées aux arcs du graphe sont adaptées de manière à récupérer, après le calcul de la coupe minimale, un sous graphe dense qui comporte

en plus des sommets attributs de m_{graine} un ensemble d'observations O_0 qui sont fortement associées à ces attributs. Lors de la prochaine étape, nous construisons le graphe correspondant aux observations O_0 de manière à récupérer, après le calcul de la coupe minimale, un sous ensemble d'attributs ayant des densités supérieures à δ pour ces observations O_0. Comme illustré dans l'Algorithme 1, ce processus est répété jusqu'à ce que le sous graphe dense extrait à l'étape n soit identique à celui extrait à l'étape $n-1$, dans ce cas notre sous graphe ne peut plus être augmenté et le processus est arrêté.

Algorithme 1: L'algorithme de recherche des régions denses

Entrées : D : matrice des données, m_{graine} : motif graine, δ : densité minimale de chaque colonne (attribut)
Sorties : SGD : l'ensemble des sous graphes denses

début
- $O_{pre}=\emptyset$; /*initialisation*/
- $A_{pre}=\emptyset$;
- i=0
- G_i=CONSTRUIRE_GRAPHE(m_{graine}, D);
- (O_i,A_i)= ST_COUPE-MINIMALE(G_i);
- **tant que** *($O_{pre} \neq O_i$ ou $A_{pre} \neq A_i$)* **faire**
  - SGD=SGD $\cup$ (O_i,A_i) ;
  - $O_{pre}=O_i$;
  - $A_{pre}=A_i$;
  - i++;
  - **si** *(i est impair)* **alors**
    - G_i=CONSTRUIRE_GRAPHE_OBSERVATIONS(O_{pre}, D);
  - **sinon**
    - G_i=CONSTRUIRE_GRAPHE_ATTRIBUTS(A_{pre}, D, δ);
  - (O_i,A_i)= ST_COUPE-MINIMALE(G_i) ;

La construction d'un graphe lors de l'appel avec un ensemble d'observations diffère de celle avec des attributs. En effet, les contraintes sur les attributs et sur les observations sont différentes. Cette différence est exprimée dans l'affectation des poids (capacités) lors de la construction d'un graphe correspondant à des observations ou des attributs.

Calcul et affectation des poids

L'idée est de construire notre graphe pondéré de manière à ce que tous les sommets qui forment un sous graphe dense soient coupés des autres sommets du graphe. Pour cela, nous ajoutons dans un premier temps les sommets source s et destination t. Soit $E_{sommets}$ un ensemble de sommets attributs ou observations pour lequel nous construisons le graphe correspondant. Nous relions la source à tous les sommets de l'ensemble $E_{sommets}$ et nous affectons le poids $+\infty$ à chacun des arcs qui les relient à s. En fonction des sommets (attributs ou observations), nous calculons les poids des arcs qui relient les attributs aux observations (resp. les observations aux attributs) de sorte que si ces observations (resp. attributs) vérifient la contrainte de forte association (resp. de densité minimale) ; les poids des arcs qui les relient aux attributs (resp. observations) soient plus grands que ceux qui les

Algorithme 2: CONSTRUIRE_GRAPHE_ATTRIBUTS

Entrées : $E_{sommets}$: ensemble des sommets attributs, D : matrice des données

Sorties : $G(V, E)$: le graphe construit

début

$V = E_{sommets} \cup \{s, t\}$;

pour tous les $e_i \in E_{sommets}$ **faire**

$E = E \cup (s, e_i)$; poids$(s, e_i) = +\infty$;

pour tous les $a_i \in E_{sommets}$ **faire**

pour tous les $D[o_j][a_i] == 1$ **faire**

$V = V \cup o_j$;

$E = E \cup (a_i, o_j)$;

$$\text{poids}(a_i, o_j) = \left(\frac{d^+(a_i)}{max_{a_k \in E_{sommets}}(d^+(a_k))} + \frac{d^-(o_j)}{max_{o_k \in O_j}(d^-(o_k))}\right) \times \frac{100}{|E_{sommets}|} ;$$

pour tous les $o_j \in V \setminus (E_{sommets} \cup \{s, t\})$ **faire**

$E = E \cup (o_j, t)$;

poids $(o_j, t) = max_{o_k \in O_j}(d^-(o_k)) \times \frac{200}{|E_{sommets}|}$ - $\sum_{a_i \in E_{sommets}} poids(a_i, o_j)$

Algorithme 3: CONSTRUIRE_GRAPHE_OBSERVATIONS

Entrées : $E_{sommets}$: ensemble des sommets observations, D : matrice des données, δ : paramètre de densité

Sorties : $G(V, E)$: le graphe construit

début

$V = E_{sommets} \cup \{s, t\}$;

pour tous les $e_i \in E_{sommets}$ **faire**

$E = E \cup (s, e_i)$; poids$(s, e_i) = +\infty$;

pour tous les $o_i \in E_{sommets}$ **faire**

pour tous les $D[o_i][a_j] == 1$ **faire**

$V = V \cup a_j$;

$E = E \cup (o_i, a_j)$;

$$\text{poids}(o_i, a_j) = \frac{100}{|E_{sommets}|} ;$$

pour tous les $a_j \in V \setminus (E_{sommets} \cup \{s, t\})$ **faire**

$E = E \cup (a_j, t)$;

poids $(a_j, t) = 2 \times (100 \times \delta)$ - $\sum_{o_i \in E_{sommets}} poids(o_i, a_j)$;

relient à la destination t et forcer ainsi à couper les arcs qui relient ces observations (resp. attributs) à la destination.

2.3.4 Stratégies de calcul des motifs graines

L'idée de se baser sur un motif graine comme point de départ dans un problème de biclustering a été utilisée dans différents travaux (Bergmann et al. (2003), Zhao et al. (2011), An et al. (2012)).

Le choix des motifs graines utilisés comme points de départ pour la recherche de régions denses maximales est une étape difficile. En effet, la qualité de nos résultats est dépendante de ces graines qui forment les solutions initiales et les points de départ des régions extraites.

Nous avons dans un premier temps étudié les caractéristiques d'un bon motif graine. L'intuition est que ces motifs graines doivent vérifier les propriétés suivantes :

1. faible recouvrement pour une couverture maximale du contexte de données
2. spécifique pour être associé à une région dense non-redondante
3. petite taille pour faciliter les calculs et optimiser le temps de calcul

Les propriétés 1 et 2 sont des propriétés *globales* des motifs (Crémilleux et Soulet (2008)) et peuvent donc être impossibles à vérifier de manière efficace. Par ailleurs, aucune caractérisation directe des graines satisfaisant la propriété 3 n'est valable. Par contre, nous pouvons expérimenter les diverses caractérisations des motifs graines qui satisfont les propriétés 1 et 3 et *a posteriori* évaluer la propriété 2.

Nous avons proposé et évalué différentes stratégies heuristiques d'extraction de motifs graines. La première consistait en l'utilisation des maximaux fréquents (méthode $HERD$ (Heuristique d'Extraction de Régions Denses)), mais l'utilisation de tous les motifs maximaux comme motifs graines entraîne des résultats redondants ; de plus, quand la densité des données est élevée, le calcul de tous les motifs fréquents maximaux reste très coûteux en temps d'exécution. Nous avons ensuite proposé plusieurs statégies heuristiques d'extraction de motifs graines à faible recouvrement.

Nous introduisons un contexte supplémentaire D_{mg}, utilisé spécifiquement pour le calcul des motifs graines. D_{mg} est initialement égal à D la matrice des données et est mis à jour itérativement au cours du processus d'extraction. Après qu'une région dense (O_j, A_j) soit calculée, chaque élément de cette région est mis à zéro dans D_{mg} (le contexte d'extraction des régions denses D reste inchangé). Afin de minimiser la perte d'information, le support minimal σ' pour le calcul des motifs graines dans le contexte D_{mg} est calculé de manière adaptative.

La stratégie retenue dans la méthode $HANCIM$ (Hybrid Approach for Noisy Contexts Itemset Mining) consiste à calculer les graines avec le plus petit taux de recouvrement possible. Le motif graine m_{graine} est initialisé à a, un des attributs ayant le support le plus élevé dans D_{mg}. En suivant le même principe, nous recherchons itérativement un attribut a qui forme avec les attributs de m_{graine} un motif de support le plus élevé possible et tel que $support(D_{mg}, m_{graine} \cup a) \geq \sigma'$. Le processus est arrêté lorsqu'aucun autre attribut ne peut être ajouté (Algorithme 4).

Gestion des redondances et support adaptatif

La redondance des informations que comportent les motifs résultats est une limite commune aux différentes approches d'extraction de motifs. Pour remédier à cette limite, nous favorisons des motifs graines disjoints et spécifiques. Pour ce faire, nous distinguons deux contextes : un contexte pour l'extraction des motifs

Algorithme 4: MOTIF_GRAINE

Entrées : D_{mg} : matrice de calcul des motifs graines,
σ' : support minimal

Sorties : m_{graine} : un motif graine

début

$m_{graine} = \emptyset$;
/* a_j : un attribut de D_{mg} */
$a = argmax_{a_j}(SUPPORT(a_j, D_{mg}))$;
tant que $(SUPPORT((a \cup m_{graine}), D_{mg}) \geq \sigma')$ **faire**
$m_{graine} = m_{graine} \cup a$;
$a = argmax_{a_j \wedge a_j \notin m_{graine}} (SUPPORT((m_{graine} \cup a_j), D_{mg}))$;

initiaux et un autre pour les régions denses maximales initialement égaux à la matrice des données D. Les régions denses maximales extraites sont mises à zéro dans la matrice d'extraction des motifs graines, ce qui permet d'éviter l'extraction de nouvelles graines ayant un fort recouvrement avec cette région et susceptibles de nous mener vers une même région.

Avec cette méthode, le cas de résultats redondants ne se produit pas : la région dense extraite pour le premier motif initial sera mise à 0 dans la matrice d'extraction des motifs graines et donc le deuxième motif graine ne sera pas extrait. Nous rappelons que les sous-matrices denses maximales extraites ne sont mises à 0 que dans le contexte d'extraction des motifs graines D_{mg} et pas dans celui d'extraction des régions denses.

Afin d'assurer l'extraction du plus grand nombre de régions denses maximales et d'éviter la perte d'informations, nous avons défini un *support minimal adaptatif* σ*'*. Le support minimal d'extraction des motifs graines est mis à jour lors de chaque étape en fonction de la modification de la densité du contexte d'extraction des motifs graines. Comme les régions denses extraites sont mises à 0 dans le contexte d'extraction des motifs graines D_{mg}, la densité de ce dernier diminue au fur et à mesure. De ce fait, des motifs vérifiant le support minimal initial dans la matrice des données peuvent ne pas le vérifier après la mise à 0 des régions extraites. Pour éviter la perte de ces motifs, le support minimal σ est mis à jour à chaque itération en fonction du rapport entre la nouvelle densité, la densité initiale et le support initial.

2.3.5 Complexité et convergence

Soit $D = (O, A)$ un contexte formel tel que $|A| = n$ et $|O| = m$. L'extraction des régions denses maximales comporte deux principales étapes : calcul d'un motif graine vérifiant un support minimal σ', puis de la région dense qui l'inclut vérifiant une densité minimale δ et un support minimal σ. Nous allons évaluer dans un premier temps la complexité dans le pire cas de l'étape de calcul des motifs graines : les maximaux fréquents pour notre méthode $HERD$ et la stratégie 1 pour $HANCIM$.

Algorithme 5: Recherche des régions denses pour HANCIM

Entrées : D : matrice des données, δ : densité minimale de chaque attribut, σ : support minimal

Sorties : SGD : ensemble des sous graphes denses

début

- $D_{mg} = D$;
- $\sigma' = \sigma$; /* σ' désigne le support adaptatif */
- $densite$ = DENSITÉ(D_{mg});
- m_0 = MOTIF_GRAINE(D_{mg}, σ');
- **tant que** *($m_0 \neq \emptyset$)* **faire**
  - A_{pre}={ } ; O_{pre}={ } ; i=0
  - G_i=CONSTRUIRE_GRAPHE_ATTRIBUTS(m_i, D);
  - (O_i, A_i)= ST_COUPE-MINIMALE(G_i);
  - **tant que** *($A_{pre} \neq A_i$ ou $O_{pre} \neq O_i$)* **faire**
    - SGD=SGD $\cup$ (O_i, A_i) ;
    - A_{pre}=A_i ; O_{pre}=O_i ; i++;
    - **si** *(i est impair)* **alors**
      - G_i=CONSTRUIRE_GRAPHE_OBSERVATIONS(O_{pre}, D, δ);
    - **sinon**
      - G_i=CONSTRUIRE_GRAPHE_ATTRIBUTS(A_{pre}, D);
    - (O_i, A_i)= ST_COUPE-MINIMALE(G_i);
  - METTRE_A_ZERO(D_{mg}, O_i, A_i);
  - $densite$' = DENSITÉ(D_{mg});
  - **si** *($\frac{\sigma \times densite'}{densite} \geq 0.1$)* **alors**
    - $\sigma' = \frac{\sigma \times densite'}{densite}$;
  - m_0 = MOTIF_GRAINE(D_{mg}, σ');

Complexité de la Stratégie heuristique d'extraction de motifs graines

Le calcul d'un motif graine est ici en $O(mn^2)$. Si le motif graine m_{graine} est de taille n, aucun attribut ne sera ajouté à m_{graine}. Dans ce cas, le calcul de a de plus grand support se fait en $O(mn)$.

Complexité du calcul d'une région dense

Le calcul d'une région dense qui inclut un motif graine se base sur le calcul de coupes minimales. Pour cela, nous avons utilisé l'implémentation FIFO de l'algorithme $push - relabel$ de complexité temporelle $O(|V|^3)$ tel que V est l'ensemble des sommets du graphe (Goldberg et Tarjan (1986)). Dans le pire des cas, un graphe va contenir tous les attributs ainsi que toutes les observations du contexte, ie $n+m$ sommets, ce qui donne une complexité de $O((n+m)^3)$. La méthode est itérative, et dans le pire des cas, à chaque étape un seul attribut ou observation est ajouté à la région. De ce fait, on obtient au maximum $n-2$ itérations lors des étapes impaires puisque la graine est composée d'au moins 2 attributs. On obtient d'un autre côté $m - \sigma'_{absolu}$ itérations lors des étapes paires car lors de la première étape, on récupère au minimum σ'_{absolu} observations. Le calcul de la densité d'une région ainsi que la mise à zéro d'une région se font dans le pire des cas en $O(nm)$. On obtient donc une complexité du calcul d'une région dense pour un motif graine égale à $O((n+m)^4)$ et le calcul d'un résultat en $O((n+m)^4)$. Comme un motif graine est composé au minimum de 2 attributs, le nombre maximal de motifs graines que l'on pourrait obtenir dans le pire des cas est égal à $\frac{n(n-1)}{2}$. On obtient ainsi une borne supérieure pour $HANCIM$ égale à $O(n^2(n+m)^4)$. Cependant, comme dans les jeux de données réels, le nombre d'attributs est largement plus grand que le nombre d'observations, on obtient une complexité bornée par $O(n^6)$.

Cependant, dans la pratique, la complexité de l'approche ne pourrait pas atteindre $O(n^6)$. En effet, nous avons considéré la complexité dans le pire cas pour les deux phases de la méthodes : le calcul des motifs graines et le calcul de la région dense. Néanmoins, ce cas ne pourra pas se produire :

- si la méthode atteint le pire cas lors de l'étape de calcul d'un motif graine (i.e. un motif graine composé de tous les attributs de la matrice), la deuxième phase nécessitera uniquement deux itérations au lieu de $n-2$ itérations dans le pire cas.
- de même, si l'on suppose que le pire cas se produit pour la deuxième phase de calcul de la région dense, la graine doit être composée de 2 attributs seulement. Dans ce cas, le calcul de ce motif graine se fait en $O(mn)$. De plus, aucun autre résultat ne sera calculé car tous les éléments de la matrice seront mis à zéro et la compléxité se reduit dans ce cas à $O((n+m)^4)$.

Étude de la convergence

Comme indiqué précédemment, $HANCIM$ est composé de deux phases : calcul d'un motif graine et calcul d'une région dense comportant le motif graine et vérifiant une contrainte de densité et de support minimal. Dans ce cas, l'algorithme converge si ces deux étapes convergent.

Lors du calcul d'une région dense, notre approche $HANCIM$ n'interdit pas qu'un élément (attribut ou observation) ajouté à la région résultat soit retiré lors des prochaines étapes, sauf pour les attributs du motif graine. Soit $m = a_1 a_2 ... a_j$

un motif graine de D, tel que $j < n$. Durant les étapes d'extraction de la région dense comportant m, aucun des attributs de m ne sera retiré du résultat. Cependant, supposons que lors d'une étape l_i un attribut a_k (ou une observation o_k) soit ajouté(e) à la région dense, durant les prochaines étapes les éléments (attributs a_k ou observations o_k) ajoutés lors des étapes précédentes peuvent être retirés puisque rien ne l'interdit. Des cycles d'ajouts et de retraits de certains éléments sont donc possibles, ce qui provoque, en théorie, la non convergence d'$HANCIM$.

De ce fait, nous nous sommes intéressés à la convergence d'une variante de $HANCIM$, $HANCIM_{affaibli}$ que nous exposons ci-après. Lors du calcul d'une région dense, nous interdisons dans cette variante $HANCIM_{affaibli}$ de retirer à une itération i un attribut (resp. une observation) ajouté à la région dense résultat lors des itérations précédentes.

Lemma 1. *Le nombre de motifs graines que l'on peut calculer avec $HANCIM$ à partir d'un contexte formel $D = (O, A)$ est fini.*

Preuve 6. *Soient $D = (O, A)$ un contexte formel tel que $|A| = n$ et m_{graine} un motif graine. m_{graine} est composé au minimum de deux attributs. Par conséquent, le nombre maximal de graines que peut contenir D est le nombre de combinaisons de 2 attributs parmi n. Le nombre de graines possible que peut calculer $HANCIM$ est donc égal à $\frac{n(n-1)}{2}$.*
Il nous reste à prouver que $HANCIM$ ne pourrait pas calculer un même motif graine plusieurs fois. Ce cas ne peut pas se produire car chaque motif graine nous mène vers une région dense qui vérifie les deux contraintes, de densité et support minimal, et cette région est mise à zéro ; y compris le motif graine, ce qui rend le calcul d'un même motif graine plus d'une fois impossible.
Nous concluons que le nombre de motifs graines que peut calculer HANCIM est fini.

Lors de l'étape de calcul d'une région dense qui inclut un motif graine, $HANCIM$ itère deux étapes : augmentation de la région dense résultat en attributs et en observations.

Lemma 2. *Le nombre d'itérations de calcul d'une région dense est fini.*

Preuve 7. *Une région dense peut être composée au maximum de tous les attributs et observations de D. Dans ce cas, le nombre maximum d'itérations est atteint lorsque à chaque itération la région est augmentée d'un seul élément (attribut ou observation) et on obtient ainsi un nombre fini d'itérations, borné par $(n-2) + (m - \sigma_{absolu})$.*

Comme $HANCIM_{affaibli}$ vérifie le lemme 1 ainsi que le lemme 2, nous concluons que cette variante converge.

Une étude comparative a été réalisée entre les résultats de $HANCIM$ et ceux obtenus par cette dernière variante de l'algorithme. Les résultats obtenus par $HANCIM$ se sont avérés globalement similaires à ceux obtenus par cette dernière variante ce qui est dû au fait que l'ajout d'éléments et leur retrait durant les itérations suivantes ne se produit que rarement. Cependant, les résultats d'$HANCIM$ se sont avérés légèrement meilleurs sur les données réelles d'expression de gènes.

2.3.6 Résultats expérimentaux

Nous avons réalisé les expérimentations sur un ordinateur équipé d'un microprocesseur intel(R) Pentium(R) 4 (3 GHz) et d'une mémoire vive de 2Go.

Nous avons évalué dans un premier temps les différentes approches de calcul de motifs graines combinés avec l'extraction de régions denses sur des données manuelles. L'évaluation a été effectuée sur deux types de données. Nous nous sommes par la suite intéressés à l'évaluation des résultats d'HANCIM sur des données synthétiques manuelles et générées de façon aléatoire. Nous avons également évalué notre méthode sur des données réelles d'expression de levure et humaines. Nous avons comparé notre approche avec différentes méthodes de l'état de l'art : une méthode de recherche de motifs fréquents bruités *Dense* (Seppanen et Mannila (2004)), une méthode de recherche de $quasi-biclique$ (Uno et Arimura (2008)) ainsi que des méthodes de Biclustering : $BiMax$ (Prelic et al. (2006)), $SScorr$ (Nepomuceno et al. (2011)) et $BaggedBiclustering$ (Hanczar et Nadif (2011)).

Évaluation sur des données synthétiques : Analyse supervisée

L'avantage de l'évaluation sur des données synthétiques est que les motifs résultats sont connus et permettent ainsi d'exploiter les mesures supervisées pour l'évaluation.

Nous présentons ci-dessous les résultats obtenus par $HANCIM$ ainsi que par les autres méthodes auxquelles nous nous sommes comparés. Nous avons comparé dans un premier temps $HERD$ et $HANCIM$ dans le but d'appréhender l'influence des motifs graines sur la nature et la pertinence des résultats. Pour les approches de l'état de l'art, nous avons choisi des méthodes de différent type :

- une méthode d'extraction de motifs fréquents bruités : *Dense* (Seppanen et Mannila (2004))
- une méthode de graphe : $Quasi-Biclique$ (Uno et Arimura (2008))
- une méthode de biclustering : $BiMax$ (Prelic et al. (2006)).

Les évaluations ont été réalisées sur des données synthétiques de deux types :

- le jeu de données défini par Gupta et al. (2008) ;
- des données générées de façon aléatoire.

Nous exposons dans le tableau 2.3 le nombre de résultats obtenus avec chaque approche pour un taux de bruit variant entre 4% et 20% sur le jeu de données de Gupta et al. (2008). Les résultats obtenus avec *Dense* ne sont pas tous maximaux, la dernière colonne du tableau 2.3 donne le nombre de résultats extraits par *Dense* ainsi que le nombre de résultats maximaux parmi ceux-ci. Notons que dans toutes les autres évaluations de *Dense* (couverture, couverture des faux positifs, pertinence ainsi que la redondance), nous n'avons considéré que les motifs maximaux.

	$HANCIM$	BiMax	Quasi-Bicliques	HERD	Dense (ts motifs/Maximaux)
4%	3	100	991	25	515/18
8%	3	106	103	65	725/41
12%	3	110	100	68	844/76
16%	4	441	97	143	1068/89
20%	7	826	91	425	1784/322

TABLE 2.3 – *Nombre de résultats extraits sur le jeu de Gupta Gupta et al. (2008)*

	$HANCIM$	BiMax	Quasi-Bicliques	HERD	Dense (Maximaux)
4%	4	5005	8388	265	552
8%	4	5319	3138	328	942
12%	3	5552	3138	474	3072
16%	2	4634	25806	4157	3250
20%	2	4113	14746	6232	26076

TABLE 2.4 – *Redondance des résultats extraits sur le jeu de Gupta Gupta et al. (2008)*

En analysant le nombre de résultats rapportés dans le tableau 2.3, nous remarquons que le nombre de résultats augmente quand le taux de bruit augmente dans le jeu de données pour toutes les approches sauf pour *Quasi-bicliques*. Même si on s'attend à ce que le nombre de résultats augmente avec le taux de bruit, il doit tout de même rester le plus proche possible du nombre de motifs initiaux. En effet, le nombre de régions extraites par $HANCIM$ reste le plus proche du nombre réel de motifs qui est égal à 3, tandis que le nombre de résultats des autres méthodes explose avec le bruit.

Il y a une relation entre le nombre de motifs et la redondance. En effet, en comparant les résultats des deux tableaux 2.3 et 2.4, les approches qui ont un nombre de résultats élevé ont une redondance importante et donc des résultats non pertinents. Contrairement à toutes les autres méthodes, $HANCIM$ limite la redondance dans les résultats en fusionnant les motifs avec un fort recouvrement. En se basant sur ces deux critères d'évaluation, notre méthode $HANCIM$ reste la meilleure.

D'autre part, nous avons généré des données synthétiques pour évaluer les performances de nos algorithmes sur une large gamme de jeux de données.

Dans le but d'étudier le comportement de notre approche $HANCIM$ sur des jeux de données dans lesquels le nombre d'attributs excède le nombre d'observations, comme c'est le cas des données d'expression de gènes, nous avons généré des jeux de données en fixant leur taille en nombre d'attributs à $|A| = 1000$ et leur taille en nombre d'observation à $|O| = 100$. La matrice des données ne comporte initialement que des valeurs nulles. Tant que la densité des données en 1 est inférieure à une densité δ fixée par l'utilisateur, des régions de densité 1 sont aléatoirement générées.

Pour pouvoir analyser le comportement de la méthode par rapport à la taille des régions, trois types de régions ont été générées : des régions de petite, moyenne et grande taille, relativement à la taille du jeu de données.

Nous avons conclu de nos expérimentations sur les données générées aléatoirement que $HANCIM$ procure de très bons résultats (nombre de régions retrouvées, couverture, etc) notamment dans le cas de données comportant des motifs de grande taille.

Évaluation sur des données réelles et validation biologique

Nous exposons dans cette section les différentes expérimentations et évaluations de notre approche sur des données réelles de bioinformatique. Nous allons tout d'abord présenter les données sur lesquelles nous avons travaillé.

Application aux données de levure Nous allons présenter dans un premier temps une étude quantitative puis qualitative des résultats de nos approches $HERD$ et $HANCIM$ sur deux jeux de données de levure de Spellman et al.

(1998) et Gasch et al. (2000).

Données de Spellman(Spellman et al. (1998))
Elles se composent d'une série de 69 puces à ADN mesurant l'expression d'une sélection de 407 gènes pendant le cycle cellulaire chez la levure. Ces données d'expression proviennent de cultures de la levure synchronisée par quatre méthodes indépendantes (Cho et al. (1998)).

Données de Gasch (Gasch et al. (2000))
Ces données sont une collection de 173 échantillons sous différentes conditions de stress qui mesure l'expression de 6152 gènes de levures. Dans Prelic et al. (2006), les auteurs se sont restreints à une sélection de 2993 gènes, utilisés dans plusieurs travaux par la suite (Nepomuceno et al. (2011), Hanczar et Nadif (2011), Deodhar et al. (2009)).

Pour la discrétisation, nous avons utilisé le modèle décrit dans Prelic et al. (2006), nous avons pris un seuil de discrétisation égal à $e_{min} + (e_{max} - e_{min})/2$ où e_{min} et e_{max} représentent respectivement les valeurs minimales et maximales d'expression des gènes dans la matrice de données.

	HANCIM				*HERD*			
δ	# résultats	densité moy	taille max	temps	# résultats	densité moy	taille max	temps
0.5	353	0.75	36	2s	5380	0.81	95	1m 32s
0.6	857	0.83	36	6s	5096	0.85	90	1m 30s
0.7	896	0.91	36	7s	3485	0.89	43	1m 23s
0.8	561	0.97	21	12s	1208	0.96	13	27s

TABLE 2.5 – *Résultats des expérimentations sur les données de Spellman Spellman et al. (1998)*

Les résultats du tableau 2.5 nous montrent que $HANCIM$ permet d'extraire des motifs de densité élevée, de grande taille et surtout en un temps d'exécution raisonnable par rapport à $HERD$.

Nous nous sommes également comparés aux approches de biclustering.

Dans le but de valider biologiquement nos résultats, nous les avons comparé avec ceux obtenus par des approches récentes et pertinentes de l'état de l'art : *BiMax* (Prelic et al. (2006)), $SScorr$ (Nepomuceno et al. (2011)) et $BaggedBiclustering$ (Hanczar et Nadif (2011)).

Nous avons lancé $HANCIM$ ainsi que $Bimax$ sur ces données de Gasch restreint (173 échantillons $\times$ 2993 gènes) (Prelic et al. (2006)) discrétisées, avec un support minimal à 20% et la densité à 80% pour $HANCIM$. Cependant, les calculs avec $Bimax$ ont été arrêtés après six jours et ont généré des fichiers résultats de l'ordre de 300Go. Contrairement à $Bimax$, notre méthode a obtenu 548 régions en seulement 12 minutes.

Nous nous sommes ensuite intéressés à l'étude de la pertinence biologique des résultats obtenus par nos approches ainsi que les autres méthodes auxquelles nous nous comparons. Notre but consiste à détecter des relations de co-expression entre les gènes à partir de données d'expression, en se basant sur le fait que des gènes co-exprimés sont censés avoir les mêmes fonctions biologiques.

Nous avons comparé dans un premier temps les résultats obtenus par *HANCIM* et *BiMax* sur les données de Gasch restreintes (2993 gènes $\times$ 173 échantillons).

Comme indiqué précédemment, *BiMax* génère des fichiers résultats très volumineux. Par conséquent, nous avons choisi de comparer nos résultats aux 100 biclusters extraits du même jeu de données par *BiMax* et publiés dans Prelic et al. (2006) (supplementary material).

Pour évaluer la pertinence biologique des motifs extraits par les deux approches sur ces données, nous calculons l'enrichissement biologique des biclusters extraits dans Gene Ontology (GO) (Cherry et al. (1998)), comme il est fait dans Birmele et al. (2008).

Nous présentons dans le tableau 2.6, le pourcentage des motifs annotés par HANCIM et Bimax selon leur pvalue.

Définition 1. *(Pvalue)*
La pvalue est le plus petit niveau pour lequel on rejette l'hypothèse nulle. Elle exprime la probabilité d'obtenir les mêmes résultats, ou des valeurs plus extrêmes, si l'hypothèse nulle était vraie.

En d'autres termes, la pvalue exprime la probabilité de commettre une erreur et donc d'obtenir un faux positif. De ce fait, les valeurs les plus basses pour la pvalue dénotent des associations plus significatives. Dans notre cas, la pvalue mesure la probabilité qu'une liste de gènes de même taille tirée aléatoirement présente la même proportion de gènes ayant une fonction GO donnée.

	$<e^{-2}$	$<e^{-3}$	$<e^{-4}$	$<e^{-5}$	$<e^{-10}$	$<e^{-20}$
HANCIM	94%	48%	28%	18%	7%	4%
BiMax	80%	9%	0%	0%	0%	0%

TABLE 2.6 – *Pourcentage des motifs annotés selon leur pvalue.*

Comme nous pouvons le remarquer à partir du tableau 2.6, les meilleurs résultats sont obtenus par notre méthode.

Nous avons également comparé *HANCIM* avec *BiMax* (Prelic et al. (2006)), $SScorr$ (Nepomuceno et al. (2011)) et $BaggedBiclustering$ (Hanczar et Nadif (2011)) sur le même jeu de données de Gasch restreint (2993 gènes $\times$ 173 observations) utilisé dans les travaux de Prelic et al. (2006), Nepomuceno et al. (2011) et Hanczar et Nadif (2011).

Dans Nepomuceno et al. (2011), les auteurs donnent leurs résultats pour les 100 meilleurs biclusters. Pour une comparaison équitable, nous présentons et évaluons les 100 meilleurs biclusters en terme de pvalue obtenus par *HANCIM*, ainsi que l'évaluation de 100 biclusters choisis aléatoirement parmi tous les résultats obtenus par *HANCIM*. Nous avons sélectionné aléatoirement 100 biclusters parmi tous les résultats obtenus et nous présentons la moyenne ainsi que l'écart-type sur 10 exécutions. Nous avons comparé également nos résultats aux meilleurs résultats obtenus par $BaggedPlaid$ (resp. $BaggedCC$) dans Hanczar et Nadif (2011), obtenus avec un nombre de biclusters égal à 5 (resp. 8).

Le tableau 2.7 donne pour chaque méthode le pourcentage de résultats enrichis avec les termes de l'ontologie GO, dans lesquels au moins un terme GO est surreprésenté pour les différents niveaux de pvalues. La deuxième ligne du tableau 2.7 comporte la moyenne ainsi que l'écart type pour les 10 exécutions des tirages aléatoires.

Comme nous pouvons le constater à partir du tableau 2.7, $HANCIM$ (100 meilleurs ou encore 100 aléatoires) a obtenu de meilleurs résultats par rapport aux approches auxquelles nous nous sommes comparés.

	$<e^{-2}$	$<e^{-3}$	$<e^{-4}$	$<e^{-5}$	$<e^{-10}$	Meilleure pvalue
Meilleurs *HANCIM*	100	100	100	100	60	$3.1e^{-56}$
$HANCIM_R$	79.9 (2.8)	51(2.6)	34.7(2.6)	26.5(2.5)	12 (2)	$1e^{-37}$
BiMax	80	9	0	0	0	$6e^{-04}$
SScorr	30	18	8	2	0	$1.4e^{-07}$
BaggedCC	8	3	0	0	0	$1.6e^{-04}$
BaggedPlaid	5	3	1	1	0	$1e^{-07}$

TABLE 2.7 – *Meilleurs* HANCIM *(100 meilleurs biclusters),* $HANCIM_R$ *(100 biclusters aléatoires),* BiMax *(Prelic et al. (2006) supplementary material) , SScorr (100 meilleurs), BaggedCC (8) et BaggedPlaid (5) sur les données de Gasch restreint, nombre de biclusters pour lesquels au moins un terme GO est sur-représenté*

Application aux données humaines Après une étude sur les données de levure, nous nous sommes intéressés à la qualité de notre méthode $HANCIM$ sur des données humaines : Lung (Bhattacharjee et al. (2001)), Stransky (Stransky et al. (2006)) et Lindgren (Lindgren et al. (2010)).

Les données de Lung sont des données réelles humaines, ces données de cancer du poumon comportent les valeurs d'expression de 1543 gènes humains pour 203 expériences.

Nous comparons à présent les résultats d'*HANCIM*, *BiMax* et *Bagged Biclustering*. Notons que pour ces données, *HANCIM* a extrait 100 résultats. Pour les mêmes raisons que précédemment, nous avons comparé nos résultats aux 100 premiers biclusters extraits par *BiMax*. Les meilleurs résultats pour *BaggedPlaid* et *BaggedCC* sont obtenus pour un nombre de biclusters égal à 10 ; c'est pourquoi nous nous comparons à ces résultats. Nous exposons dans le tableau 2.8 pour chaque méthode, la taille moyenne en nombre de gènes et d'expériences des biclusters obtenus et pour les différents niveaux de signification, le nombre de biclusters pour lesquels au moins un terme GO est sur-représenté. Sur ce jeu de données, *HANCIM* trouve un nombre raisonnable de motifs enrichis et avec des tailles plus importantes comparativement aux autres méthodes.

	Taille moyenne	$<e^{-2}$	$<e^{-3}$	$<e^{-4}$	$<e^{-5}$	$<e^{-6}$	Meilleure pvalue
HANCIM	(31×108)	94	44	21	10	5	$2.6e^{-10}$
BiMax	(31 × 40)	0	0	0	0	0	-
BaggedCC	(11× 8)	5	0	0	0	0	$1.1e^{-03}$
BaggedPlaid	(15× 35)	2	0	0	0	0	$4e^{-03}$

TABLE 2.8 – HANCIM, BiMax *(100 premiers), BaggedCC (10) et BaggedPlaid (10) résultats sur les données de Lung*

Nous avons ensuite effectué nos expérimentations sur deux types de données réelles du cancer de la vessie, des données d'expression de gènes et de la CGH. On s'intéresse à observer des liens entre gènes d'expression et CGH (Comparative Genomic Hybridization) au sein d'un même jeu Stransky/Lindgren, ou des liens entre un même type de données CGH/expression et pour des jeux différents. Nous avons choisi deux jeux de données de cancer de la vessie, les données Stransky, qui sont un sur-ensemble des données publiées dans Stransky et al. (2006) et le jeu de données Lindgren (Lindgren et al. (2010)). Ces deux ensembles de données, *transcriptome* et *variations génétiques du nombre de copies* sont disponibles et ont été mesurées en utilisant les technologies des biopuces [1].

1. La technologie des biopuces permet de caractériser le niveau d'expression d'un grand nombre de gènes d'un génome.

Les données d'expression Stransky mesurent les niveaux d'ARNm pour une série de 57 (déjà publié dans Stransky et al. (2006)) + 22 échantillons supplémentaires de tumeur de la vessie pour 8555 sondes sélectionnées, chacune correspond à des gènes uniques. Les données CGH de Stransky se composent de 2359 BAC (chromosomes artificiels de bactéries) clones qui mesurent les variations génétiques du nombre de copies (CNV (Copy Number Variations)) pour 84 échantillons (dont 57 également publiés dans Stransky et al. (2006)). Dans ces données, 57 échantillons ont été mesurés à la fois pour l'expression et la CNV.

Pour les données Lindgren, le jeu de données d'expression comporte les valeurs d'expression de 2506 gènes pour 156 échantillons (144 échantillons tumoraux + 12 échantillons normaux). Les données CGH sont constituées d'ADN à partir de 103 échantillons de 31946 puces.

Pour les deux jeux de données CGH, les ratios normalisés ont été discrétisés en des valeurs ternaires : 'gain' (1), 'normal' (0) ou 'perte' (-1), en se basant sur l'algorithme de segmentation GLAD (Hupé et al. (2004)).

De même, les deux jeux de données d'expression Stransky et Lindgren ont été discrétisés comme décrit dans Elati et al. (2007) pour convertir les niveaux d'expression en des valeurs ternaires : -1 (sous-exprimé), 0 (neutre) ou 1 (surexprimé). Cette discrétisation est plus précise qu'une discrétisation booléenne : elle permet de représenter les deux niveaux sur- et sous-expression.

Étant donné que notre approche, ainsi que *BiMax* à laquelle nous nous comparons, ne fonctionnent que sur des données binaires, nous ne considérons qu'un cas à la fois :

- sur-expression/gain : seules les valeurs +1 sont retenues, toutes les autres valeurs sont mises à 0 ;
- sous-expression/perte : seules les valeurs -1 sont retenues, toutes les autres valeurs sont mises à 0.

Nous avons lancé *HANCIM* et *BiMax* avec les mêmes paramètres sur les données de Stransky et al. (2006) et de Lindgren et al. (2010) discrétisées pour extraire les biclusters sur- et sous-exprimés dans les données d'expression, d'une part, et les régions de gain et de perte dans les données CGH d'autre part.

Nous rapportons dans le tableau 2.9 le nombre de résultats obtenus, la taille moyenne des biclusters en nombre de lignes (gènes pour les données d'expression / sondes pour les données CGH) et colonnes (échantillons) et le temps de calcul (en minutes) mis par *HANCIM* pour chaque jeu de données.

	Sur-expression/Gain			Sous-expression/Perte		
Données	#biclusters	Taille moy.	Temps	#biclusters	Taille moy.	Temps
Stransky expression	782	(175 × 18)	11	900	(254 ×14)	14
Stransky CGH	83	(25 ×11)	<1	74	(22 × 11)	<1
Lindgren expression	426	(4 × 24)	2	429	(4 × 25)	2
Lindgren CGH	808	(124 × 16)	119	786	(158 × 19)	141

TABLE 2.9 – *Nombre de résultats, la taille moyenne des biclusters en termes de nombre de lignes et de colonnes et les temps de calcul d'*HANCIM *en minutes sur les données Stransky et Lindgren CGH et expression*

Comme prévu, les temps de calcul sont assez petits, quelques minutes, sauf pour le jeu Lindgren CGH, qui est considérablement plus grand en nombre de sondes. Notons que la taille moyenne des biclusters obtenus par *BiMax* pour les données d'expression sont plus petits que ceux obtenus par *HANCIM*. Par exemple,

la taille moyenne des bicluters obtenus par *BiMax* sur les données Stransky expression est de 10 gènes et 14 échantillons.

De plus, en analysant les 100 premiers résultats extraits par *BiMax*, nous avons constaté que les biclusters se recouvrent fortement et sont donc similaires. Nous avons remarqué, sur les résultats obtenus pour les deux jeux de données Lindgren and Stransky, que plus de 10% des résultats varient uniquement de un ou deux attributs.

Comme pour les données de Lung et de Gasch, nous avons évalué la pertinence biologique des biclusters extraits dans les données d'expression en recherchant des enrichissements pour les gènes et des annotations pour les tumeurs. Idéalement, le biclustering devrait produire des régions qui relient des ensembles de tumeurs d'un type spécifique avec des ensembles de gènes spécialisés dans une fonction biologique donnée.

Les tableaux 2.10 et 2.11 donnent le pourcentage de résultats annotés pour :

1. tous les résultats obtenus par *HANCIM*,
2. 100 biclusters tirés aléatoirement parmi tous les résultats d'*HANCIM* ainsi que l'écart type pour 10 exécutions
3. *BiMax* en fonction de la pvalue pour respectivement les données d'expression Stransky et Lindgren.

	Sur-expression			Sous-expression		
	HANCIM	*HANCIM*$_R$	*BiMax*	*HANCIM*	*HANCIM*$_R$	*BiMax*
# biclusters	782	100	100	900	100	100
$<e^{-1}$	99.7%	100% (0.0)	96%	99.5%	99.5%(0.6)	78%
$<e^{-2}$	97.7%	97.9% (1.1)	3%	96.8%	97.4%(1.3)	44%
$<e^{-3}$	74%	74% (3.8)	0%	69.4%	69.8%(2.8)	0%
$<e^{-4}$	58.2%	57.7% (3.5)	0%	42.1%	43.1%(3.1)	0%
$<e^{-5}$	48.3%	47.6% (3.3)	0%	27.4%	27.8%(2.4)	0%
$<e^{-10}$	11%	10.3% (2.4)	0%	10%	10.5%(1.6)	0%
$<e^{-15}$	3.3%	2.8% (0.9)	0%	3.4%	3.2%(1.3)	0%
Meilleure pvalue	$7e^{-23}$	$1e^{-16}$	$2.8e^{-3}$	$1e^{-24}$	$1e^{-18}$	$5.3e^{-3}$

TABLE 2.10 – *Les résultats d'*HANCIM*,* HANCIM$_R$ *et* BiMax *sur les données d'expression Stransky*

	Sur-expression			Sous-expression		
	HANCIM	*HANCIM*$_R$	*BiMax*	*HANCIM*	*HANCIM*$_R$	*BiMax*
# biclusters	651	100	100	656	100	100
$<e^{-2}$	63%	62.7% (3.4)	61%	64.5%	66.2%(3.2)	55%
$<e^{-3}$	8.3%	8.6% (2.4)	10%	10%	9.4%(1.4)	10%
$<e^{-4}$	3%	3.2% (2.1)	0%	5%	4.6%(1.4)	0%
$<e^{-5}$	0.6%	0.7% (0.6)	0%	2.3%	2.3%(1.1)	0%
$<e^{-10}$	0.15%	0.2% (0.1)	0%	0.6%	0.6%(0.4)	0%
Meilleure pvalue	$6.7e^{-11}$	$2.4e^{-05}$	$5.7e^{-4}$	$4.4e^{-15}$	$2.3e^{-08}$	$5.2e^{-4}$

TABLE 2.11 – *Les résultats d'*HANCIM*,* HANCIM$_R$ *et* BiMax *sur les données d'expression Lindgren*

Les deux tableaux 2.10 et 2.11 montrent que les meilleurs pvalues sont obtenues par *HANCIM* par rapport à *BiMax* pour les cas sur-exprimés et sous-exprimés. En effet, l'enrichissement met en évidence des fonctions pour la quasi-totalité des biclusters extraits à la fois pour les données d'expression Stransky et Lindgren.

Les biclusters extraits par *BiMax* ont un fort recouvrement et sont donc très similaires. En conséquence, une partie importante des biclusters obtenus par *BiMax* partagent les mêmes annotations GO. Contrairement à *Bimax*, *HANCIM* fusionne

les biclusters qui ont un fort recouvrement et obtient ainsi des biclusters qui ont un faible recouvrement et sont donc annotés avec des termes GO plus divers et informatifs.

Nous nous sommes par la suite intéressés à la couverture des gènes et échantillons des jeux de données par les deux méthodes. En effet, tandis qu'*HANCIM* extrait un nombre raisonnable de motifs, la couverture des lignes (échantillons) et des colonnes (gènes ou sondes) est importante. Par exemple, pour les données d'expression Stransky (le cas sur-expression), les biclusters extraits par *HANCIM* couvrent 47.7% des gènes et, plus important encore, 97.5% des échantillons. De l'autre coté, les 100 premiers biclusters extraits par *BiMax* sur le même jeu de données couvrent uniquement 0.6% des gènes et 22% des échantillons.

HANCIM a pu trouver des clusters très pertinents par rapport à l'annotation des tumeurs également (des pvalues < 5%) dans les deux jeux de données. En analysant de façon combinée les annotations sur les gènes et les tumeurs, 27 biclusters dans les données de Stransky et 87 biclusters dans les données de Lindgren ont été significativement associés (pvalues < 5%) à la fois à un type de tumeurs et à une fonction de gènes. De plus, d'importants processus biologiques et modules de gènes de cancer liés à des mSigDB ont été associés à des types de tumeurs. Le tableau 2.12 comporte quelques exemples de biclusters pertinents.

ID Bicluster	Tumeur		mSigDB	Annotation GO
435	G_3	9.14e-04	5.74e-21	9.14e-04
498	superficial	0	1.03e-18	5.52e-07
675	G_2	1.51e-03	5.99e-11	1.24e-05
696	T_1	2.84e-3	4.82e-18	6.53e-08
733	invasive	5.41e-3	1.76e-8	1.49e-05

TABLE 2.12 – *Annotation des gènes et tumeurs de quelques biclusters résultats. Les tumeurs sont classées par leurs annotations cliniques. La colonne 'mSigDB' comporte les pvalues des enrichissements des modules de calcul définies à partir de la signature des données de base moléculaire. De la même manière, la colonne 'Annotation GO' est l'enrichissement dans les processus biologiques de Gene Ontology*

La flexibilité d'*HANCIM* nous a permis de trouver non seulement des motifs pertinents dans les données d'expression de gènes, mais aussi des informations sur la variation du nombre de copies (CNV) à partir des données CGH. *HANCIM* a été appliqué pour trouver des motifs CNV qui pourraient être pertinents par rapport aux annotations cliniques des tumeurs, comme pour les données d'expression. Sur les 157 et 1686 biclusters extraits respectivement à partir des données Stransky et Lindgren, pour à la fois les pertes et gains, 14 et 110 étaient significativement (pvalue < 5%) associés à la fois à un grade des tumeurs, un stade et un état de métastase.

Pour vérifier la qualité des résultats, nous avons vérifié qu'ils étaient reproductibles. Nous avons cherché si les clusters identiques pourraient être trouvés dans les deux jeux de données d'expression Stransky et Lindgren. Chaque cluster a été réduit aux gènes qui sont communs à toutes les plateformes utilisées dans Stransky et Lindgren. Sur cette base, nous avons cherché les clusters qui ont exactement les mêmes gènes lorsqu'ils sont calculés à partir des deux jeux de données. Nous avons trouvé que parmi les clusters extraits par *HANCIM* dans les deux jeux de données, 49 étaient identiques, tandis qu'aucun n'a été trouvé dans les clusters extraits par avec *Bimax*.

2.3.7 Approche de biclustering semi-supervisé contraint

Nous présentons dans cette section une nouvelle méthodologie semi-supervisée, nommée *COBIC* (COnstrained BI-Clustering), pour l'extraction de biclusters pertinents à partir de contextes bruités. Notre objectif dans ce travail consiste à exploiter des informations d'une classification de référence pour guider l'extraction des régions denses. Cet algorithme se base également sur les algorithmes de graphes, flot maximal / coupe minimale et exploite les informations de la classification pour adapter les poids des arêtes lors de la construction des graphes de manière à orienter le processus d'extraction.

Principe de $COBIC$

Notre objectif dans ce travail consiste à augmenter la pertinence des motifs extraits en exploitant les informations d'une base (voir par exemple les travaux de Ventos et Soldano (2005) pour la recherche de motifs exacts clos influencés (en fonction d'un degré α compris entre 0 et 1) par une classification a priori). En effet, il existe dans différents domaines des bases qui regroupent des informations spécifiques à chacun de ces domaines spécialisés. Ces informations peuvent, par exemple, rassembler l'expertise d'un domaine. L'idée dans ce travail consiste à exploiter des informations exprimées sous forme de *classifications* pour guider l'extraction des régions denses. L'approche offre la possibilité de guider l'extraction des biclusters pour une des dimensions ou bien les deux en même temps.

Cette orientation se fait en adaptant les poids des arcs lors de la construction des graphes, contrairement à $HERD$ et $HANCIM$ présentés précédemment dans lesquels les poids affectés aux arcs des graphes sont fixes. Ce guidage est souple car il est utilisé pour adapter les poids des arcs dans les graphes bipartis, adaptés de manière à assurer une cohérence par rapport aux informations de la classification.

En plus des deux contraintes de densité et de support minimal, $COBIC$ offre la possibilité d'assurer des contraintes de cohérence par rapport à des bases d'informations exprimées sous forme de classifications.

Définition 2. *Soit Y un ensemble fini d'éléments (attributs ou observations). Nous définissons une classification de $C_Y = \{C_i : C_i \subseteq Y, C_i \neq C_j, \forall j, i \neq j\}$.*

Aucune contrainte n'est imposée sur la structure de la classification. En effet, la classification peut être lacunaire : certains éléments de Y peuvent n'appartenir à aucune classe de C_Y. D'autres peuvent se recouvrir : soient deux classes C_i et C_j, $i \neq j$ tels que $C_i \not\subseteq C_j$ et $C_j \not\subseteq C_i$, on peut avoir le cas où $C_i \cap C_j \neq \emptyset$ et en conséquence permettre à un élément y_i d'appartenir à plusieurs classes (ce qui est le cas des gènes dans GO (Cherry et al. (1998)) et KEGG (Kanehisa et Goto (2000)).

Comme indiqué précédemment, afin de guider la recherche de biclusters vers des solutions pertinentes, nous exploitons des classifications (par exemple, GO ou KEGG ontologies fonctionnelles pour la dimension gènes dans les données d'expression).

Nous présentons dans l'algorithme 6 notre méthodologie pour le calcul de similarité moyenne entre un ensemble d'attributs (respectivement observations) et certaines classes d'attributs (respectivement observations).

Considérons Y un ensemble d'éléments (attributs ou observations), C_Y une classification de Y et Sub_Y un sous-ensemble de Y pour lequel on calcule la similarité. Pour calculer la similarité entre deux ensembles, nous utilisons la similarité

Algorithme 6: SIMILARITÉ

Entrées : Y : Un ensemble d'éléments (attributs ou observations),
C_Y : classification de Y, Sub_Y : sous ensemble de Y
Sorties : $Avrg\_Sim$: Similarité moyenne

début
  pour $(C_i \in C_Y)$ **faire**
    $Sim^{C_i}_{Sub_Y} = Sim\_JACCARD(Y, Sub_Y, C_i)$;
  $Avrg\_Sim = \frac{1}{k}\sum_{i=1}^{k} Sim^{C_i^{max}}_{Sub_Y}$;
    /*C_i^{max} est la classe ayant obtenu la i^{me} plus grande similarité*/

de Jaccard. Comme la similarité de Jaccard est définie pour être appliquée entre des partitions, nous calculons notre similarité entre $Sub_Y \subset Y$ et $C_i \in C_Y$ comme la similarité entre les partitions $(Sub_Y, Y \setminus Sub_Y)$ et $(C_i, Y \setminus C_i)$. Ainsi, nous définissons la similarité de Jaccard comme suit :
$Sim\_JACCARD(Y, Sub_Y, C_i) = \frac{N_{01} + N_{10}}{N_{01} + N_{01} + N_{11}}$
où $N_{11} = |Sub_Y \cap C_i|$, $N_{01} = |C_i \setminus Sub_Y|$, $N_{10} = |Sub_Y \setminus C_i|$ et $N_{00} = |(Y \setminus Sub_Y) \cap (Y \setminus C_i)|$.

L'algorithme exploite plusieurs classes de C_Y pour guider la recherche. Dans ce cas, nous utilisons la moyenne des k meilleures valeurs de similarités dans l'adaptation des poids.

Nous détaillons dans l'algorithme 7 la construction d'un graphe pondéré pour un ensemble d'éléments (attributs ou observations) noté X_l à la l^{eme} étape. L'objectif consiste à extraire un ensemble d'attributs satisfaisant la contrainte de densité minimale après le calcul de la coupe minimale. Lorsque $l > 2$, nous calculons une mesure de qualité pour les deux ensembles d'éléments Y_{l-1} et Y_{l-2} extraits lors des deux dernières itérations $l-1$ et $l-2$. La qualité d'un ensemble Y_i est évaluée en termes de sa similarité avec toutes les classes $C_i \in C_Y$ définies sur Y. Rappelons que, à chaque étape, notre objectif est de guider la recherche de biclusters afin de favoriser le calcul d'ensembles d'éléments cohérents avec une classification potentiellement peu dense C_Y, et non avec une seule classe de C_Y.

Compte tenu du comportement de notre algorithme lors des deux itérations précédentes, nous vérifions la similarité des ensembles Y_{l-2} et Y_{l-1} avec des classes de C_Y, notées $sim_{Y_{l-2}}$ et $sim_{Y_{l-1}}$. Lorsque la similarité de l'ensemble Y_{l-1} extrait lors de l'itération précédente $l-1$ est meilleure que la similarité de Y_{l-2}, l'algorithme se comporte bien et dans ce cas nous construisons le graphe correspondant à X_l comme nous le faisions dans *HANCIM*. Dans le cas contraire, nous utilisons les valeurs de similarité pour pondérer les capacités des arcs du graphe. En fait, si la similarité de Y_{l-1} est inférieure à celle de Y_{l-2}, cela signifie que notre solution s'éloigne de la classification C_Y.

Dans ce cas, à l'étape l, en pondérant les poids des arcs incidents aux sommets y_j appartenant à $Y_{l-1} \setminus Y_{l-2}$ par la valeur de la similarité $sim_{Y_{l-1}}$, on pénalise ces sommets y_j, et en pondérant les poids des arcs incidents aux sommets y_k appartenant à Y_{l-2} par la valeur de la similarité $sim_{Y_{l-2}} > sim_{Y_{l-1}}$, nous indiquons notre préférence pour ces sommets y_k comparés aux sommets y_j.

Pour ce faire, nous pondérons les poids des arcs (x_i, y_j) tel que $y_j \in Y_{l-1} \setminus Y_{l-2}$ par la valeur de la similarité $sim_{Y_{l-1}}$ et les arcs (x_i, y_j) tel que $y_k \in Y_{l-2}$ par la valeur de la similarité $sim_{Y_{l-2}}$. Sachant que les poids des arcs lors de la

construction d'un graphe associé à un ensemble d'attributs diffèrent de ceux lors de la construction d'un graphe associé à un ensemble d'observations, les poids originaux sont définis comme suit :

1. si $X_l \subseteq O$:
   - $W_{x_i y_j}$=$\frac{100}{|X_l|}$ et
   - $W_{y_j t}$=2 $\times$ (100 $\times$ δ) - $weight^-(y_j)$.
2. si $X_l \subseteq A$:
   - $W_{x_i y_j}$=($\frac{d^+(x_i)}{max_{x_k \in X_l}(d^+(x_k))}$ + $\frac{d^-(y_j)}{max_{y_k \in Y_j}(d^-(y_k))}$) x $\frac{100}{|X_l|}$ et
   - $W_{y_j t}$= $max_{y_k \in Y_j}(d^-(y_k)) \times \frac{200}{|X_l|}$ - $weight^-(y_j)$.

Comme $sim_{Y_{l-1}} < sim_{Y_{l-2}}$, en pondérant les poids des arcs (x_i, y_j) tels que $y_j \in Y_{l-1} \setminus Y_{l-2}$ par la valeur de la similarité $sim_{Y_{l-1}}$, les poids de ces arcs (x_i, y_j) sont considérablement réduits et donc la possibilité de couper ces arcs et ainsi de supprimer les sommets $y_j \in Y_{l-1} \setminus Y_{l-2}$ de Y_l est augmentée.

Expérimentations et évaluations

Comme pour HANCIM, nous avons analysé le comportement de $COBIC$ sur des données synthétiques. Pour cela, nous avons repris les mêmes jeux générés et utilisés dans la section précédente. La classification de référence exploitée pour l'étape d'apprentissage des poids était composée des motifs initiaux intégrés dans les jeux de données.

Nous avons conclu de ces tests que plus la classification de référence est partielle/incorrecte, plus il faut augmenter k.

Nous avons ensuite choisi d'évaluer les performances de *COBIC* sur les mêmes jeux de données utilisés pour l'évaluation d'HANCIM. Nous avons travaillé sur deux jeux de données d'expression de levure : les données de Gasch et al. (2000) et les données de Lee et al. (2004), ainsi que sur les données de Lung (Bhattacharjee et al. (2001)). Le choix de ces jeux de données est dû à la disponibilité de classifications de référence correctes, mais incomplètes, pour guider le processus de biclustering. Les données de *Gasch* utilisées sont les données complètes précédemment présentées. Les données de *Lee* comportent les valeurs d'expression de 5612 gènes de la levure pour 592 expériences.

Nous avons également évalué $COBIC$ sur les données de Lung (Bhattacharjee et al. (2001)), présentées et utilisées dans l'évaluation d'$HANCIM$.

Les données réelles manipulées sont des données d'expression de gènes. Dans ce cas, les classifications dont nous avons besoin doivent contenir des informations de référence génétiques sur les gènes de la levure pour les données de *Gasch* et *Lee* et sur les gènes humains pour $Lung$. Pour cela, nous avons exploité les informations des bases de données publiques Gene Ontology (GO) (Cherry et al. (1998)) et Kyoto Encyclopedia of Genes and Genomes (KEGG) (Kanehisa et Goto (2000)).

Les bases GO pour l'humain et la levure contiennent plus de clusters que KEGG, et par conséquent plus d'informations. En effet, pour la levure la base GO utilisée comporte 6357 gènes et contient 4227 clusters de taille moyenne 14 ($min = 2$ et $max = 254$), tandis que KEGG ne contient qu'un sous ensemble des gènes contenus dans la base GO. En effet, KEGG comporte 2026 gènes regroupés en 99 clusters de taille moyenne 37 ($min = 2$ et $max = 253$). Nous nous sommes

Algorithme 7: CONSTRUIRE_GRAPHE_COBIC

Entrées : $D = (O, A)$: matrice des données, X_l : Ensemble de sommets ($X_l \subseteq A$ ou $X_l \subseteq O$) et $l > 2$, δ : densité minimale, C_Y : Classification de Y (si $X_l \subseteq A$ alors $Y = O$, sinon $Y = A$), $Y_{l-1} \subseteq Y$: l'ensemble de sommets extraits à l'étape $l-1$, $Y_{l-2} \subseteq Y$: l'ensemble de sommets extraits à l'étape $l-2$

Sorties : $G(V, E)$: le graphe construit

```
début
    sim_{Y_{l-1}} = SIMILARITE(Y, C_Y, Y_{l-1});
    sim_{Y_{l-2}} = SIMILARITE(Y, C_Y, Y_{l-2});
    si (sim_{Y_{l-1}} < sim_{Y_{l-2}}) alors
        V= X_l ∪ {s, t} ;
        pour (x_i ∈ X_l) faire
            E=E ∪ (s, x_i);
            weight(s, x_i)= +∞ ;
        pour (x_i ∈ X_l) faire
            pour (y_j t.q. D[x_i][y_j] == 1) faire
                V=V ∪ y_j ;
                E=E ∪ (x_i, y_j) ;
                si ( y_j ∈ Y_{l-2}) alors
                    weight(x_i, y_j)= W_{x_i y_j} × sim_{Y_{l-2}} ;
                sinon
                    si ( y_j ∈ Y_{l-1}) alors
                        weight(x_i, y_j)= W_{x_i y_j} × sim_{Y_{l-1}} ;
                    sinon
                        weight(x_i, y_j)= W_{x_i y_j} ;
        pour (y_j ∈ V\(X_l ∪ {s, t})) faire
            E=E ∪ (y_j, t) ;
            weight (y_j, t)= W_{y_j t};
    sinon
        CONSTRUIRE_GRAPHE_HANCIM (cf Algorithmes 2 et 3 );
```

intéressés par la suite à l'intersection entre les classes des deux bases. Nous présentons dans le tableau 2.13 suivant les taux de recouvrement des classes de chaque base par rapport à l'autre. La première ligne correspond aux taux de recouvrement des classes. La deuxième ligne expose le pourcentage des classes de KEGG qui sont incluses dans les classes de GO. Par exemple, 5% des classes de KEGG ont des taux de recouvrement compris entre 30% et 39% avec celles de GO. On remarque que plus de 20% des classes de KEGG sont incluses à plus de 90% dans les classes GO. D'un autre côté, on trouve que peu de classes de GO sont incluses dans des classes de KEGG. On déduit de cela que les classes de KEGG sont des sous classes des classes de GO.

Pour l'humain, GO contient 14586 gènes et 6803 clusters de taille moyenne 12 ($min = 2$ et $max = 353$).

Nous avons choisi la base qui comporte le moins d'informations KEGG

	[0-29]%	[30-39]%	[40-49]%	[50-59]%	[60-69]%	[70-79]%	[80-89]	[90-99]%	100%
KEGG	0	2	5	16	24	16	15	14	8
GO	68	5	2	11	4	3	3	1	2

TABLE 2.13 – *Les taux de recouvrement des classifications GO et KEGG utilisées*

comme classification de référence pour la phase d'adaptation des poids dans *COBIC* et préserver celle qui contient plus d'informations *GO* pour l'évaluation des résultats pour toutes les approches.

Pour évaluer la pertinence biologique des biclusters extraits à partir des données réelles, nous nous sommes appuyés sur deux différentes mesures :

– l'enrichissement des biclusters extraits avec les termes GO, comme cela a été fait avec $HANCIM$, en fonction de la valeur de la pvalue.
– le pourcentage de gènes de la matrice des données qui sont classifiés.

Nous avons comparé dans un premier temps les performances de *COBIC* avec notre approche *HANCIM* ainsi qu'avec des algorithmes de biclustering performants de l'état de l'art, tels que *SScorr* (Nepomuceno et al. (2011)), *bagged biclustering* (Hanczar et Nadif (2011)) et *ROCC* (Deodhar et al. (2009)).

Les résultats que nous allons présenter dans ce qui suit, sur Gasch discrétisé, Lee et Lung, ont été obtenus pour un support minimal de 20% et une densité minimale de 80%. Nous exposons dans le tableau 2.14 les temps de calcul (en minutes) pris par *COBIC* pour chacun des jeux de données. Comme prévu, les temps de calcul sont assez faibles, de l'ordre de quelques minutes, sauf pour les données de Lee où *COBIC* a pris plus de temps.

Jeu de données (#gènes×#observations)	*COBIC*
Gasch (2993×173)	3
Gasch (6152×173)	35
Lee (5612×592)	280
Lung (1543×203)	1

TABLE 2.14 – *Les temps de calcul obtenus avec* COBIC *sur les différents jeux de données (en minutes)*

En général, nous avons constaté que le temps supplémentaire induit par la phase d'apprentissage des poids est limité à environ 20% du temps de calcul total pour tous les jeux de données testés.

Nous nous intéressons désormais à l'évaluation biologique des résultats extraits par *COBIC* sur les données de Gasch, Lee et Lung.

Nous avons comparé dans un premier temps *COBIC* avec *HANCIM* sur le jeu de données Gasch restreint (2993×173) utilisé précédemment pour l'évaluation d'*HANCIM*. Le tableau 2.15 suivant résume le pourcentage des biclusters résultats enrichis avec les termes de l'ontologie GO, dans lesquels au moins un terme GO est sur-représenté pour les différents niveaux de pvalues.

Comme on peut le constater à partir du tableau 2.15, *COBIC* a obtenu de meilleurs résultats par rapport à *HANCIM* ; on voit que le nombre de résultats est presque équivalent (6 résultats de plus avec *COBIC*). Cependant, la proportion de biclusters enrichis par au moins un terme GO pour *COBIC* est toujours supérieure à celle d'*HANCIM* pour tous les niveaux de pvalue.

Nous observons le même comportement si l'on compare *COBIC* avec les ré-

	HANCIM	*COBIC*	*COBIC* 200 meilleurs
# résultats	548	554	200
$<e^{-2}$	94%	97.3%	100%
$<e^{-3}$	48%	58%	100%
$<e^{-4}$	28%	33.7%	93.5%
$<e^{-5}$	18%	24%	66.5%
$<e^{-10}$	7%	10%	28%
$<e^{-20}$	4%	6.5%	18%
Meilleure pvalue	e^{-56}	e^{-64}	e^{-64}

TABLE 2.15 – *Résultats d'*HANCIM *et* COBIC *sur le jeu de données Gasch restreint*

sultats obtenus par *SScorr* dans Nepomuceno et al. (2011) ; de plus, on remarque que *COBIC* est considérablement meilleur que *SScorr*. Sur ces données de Gasch restreint, on trouve que sur les 100 meilleurs résultats de *SScorr*, le pourcentage de biclusters enrichis avec une pvalue inférieure à 0.01 (resp. 0.001) est inférieur à 30% (resp. 20%). Si nous analysons maintenant les 200 meilleurs biclusters obtenus avec *COBIC*, 100% de nos biclusters enrichis ont une pvalue inférieure à 0.001.

Nous présentons à présent les résultats obtenus sur les données de Gasch complet (6152×173). Nous exposons dans le tableau 2.16 le pourcentage de biclusters extraits par *COBIC* et enrichis avec les termes GO pour lesquels un ou plusieurs termes GO sont sur-représentés.

	COBIC	*COBIC* 200 meilleurs
# résultats	1075	200
$<e^{-2}$	96%	100%
$<e^{-3}$	60%	100%
$<e^{-4}$	41.5%	100%
$<e^{-5}$	35.6%	100%
$<e^{-10}$	25%	100%
$<e^{-20}$	15%	83%
Meilleure pvalue	e^{-98}	e^{-98}

TABLE 2.16 – *Les résultats de* COBIC *sur les données de Gasch complet*

Comme on peut le voir, les résultats obtenus par *COBIC* sur le jeu de données de Gasch complet sont meilleurs que ceux obtenus sur le jeu de Gasch restreint. Toutes les pvalues associées aux 200 meilleurs biclusters obtenus avec *COBIC* sont inférieures à e^{-10} et la meilleure pvalue est égale à e^{-98}.

Notons que le pourcentage de gènes de la matrice des données qui sont apparus dans les 200 meilleurs résultats extraits par *ROCC* est 0.9%, alors que nous avons regroupé avec *COBIC* 51.4% des gènes dans nos 200 meilleurs résultats.

Les données de Lee (5612×592) sont les autres données de levure sur lesquelles nous avons évalué *COBIC*. Le tableau 2.17 donne le pourcentage de biclusters résultats de *COBIC* enrichis avec les termes de l'ontologie GO, dans lesquels au moins un terme GO est sur-représenté pour les différents niveaux de pvalues.

Comme nous pouvons le voir, toutes les pvalues associées aux 200 meilleurs biclusters obtenus avec *COBIC* sont inférieures à e^{-03}.

Notons que le pourcentage de gènes (attributs) représentés dans au moins un

	COBIC	*COBIC* 200 meilleurs
# résultats	1969	200
$<e^{-2}$	86%	100%
$<e^{-3}$	34.5%	100%
$<e^{-4}$	11%	77%
$<e^{-5}$	6%	40%
$<e^{-10}$	1.5%	8.5%
$<e^{-20}$	0.2%	2%
Meilleure pvalue	e^{-43}	e^{-43}

TABLE 2.17 – *Résultats de* COBIC *sur les données de Lee*

bicluster parmi les 200 meilleurs résultats de *ROCC* est de 16.5% pour les données Lee, tandis 30% des gènes sont représentés dans les 200 meilleurs résultats de *COBIC*.

Hanczar et Nadif (2011) considèrent des biclusters dans lesquelles les gènes sont sur-représentés, par rapport aux processus KEGG, comme une identification d'information biologique pertinente ; ils considèrent également que les biclusters pertinents sont ceux dans lesquelles les gènes d'un processus donné sont sur-représentés avec une pvalue inférieure à 0.05. Le nombre maximal de biclusters trouvé par Hanczar et Nadif (2011), (resp. processus sur-représentés), est 30, (resp. 20). Nous avons obtenu avec *COBIC* 94 résultats, tous avec des processus significatifs et avec des pvalues comprises entre e^{-2} et e^{-5}.

2.3.8 Conclusion

Nous avons considéré ici l'extraction de motifs contraints dans des données booléennes bruitées. Nous avons proposé dans cette section des approches heuristiques originales qui combinent des algorithmes de fouille de données et des algorithmes de flot maximal/coupe minimale pour rechercher des sous graphes denses maximaux qui peuvent se recouvrir dans un graphe biparti pondéré et augmenté associé à la matrice des données. Nous avons également exploité les informations d'une classification des attributs et/ou des observations pour guider la recherche de ces motifs. Il pourrait également être intéressant d'étudier le comportement de la méthode en fonction du bruit sur la classification de référence.

2.4 PROGRAMMATION DYNAMIQUE PAR BLOCS

Dans cette section, nous nous intéressons aux possibilités d'amélioration de la méthode de programmation dynamique. La phase critique dans la programmation dynamique est l'opération de dominance, qui consiste à identifier pour chaque nouvelle étiquette de chaque sommet du graphe, si la solution partielle associée peut être complétée pour construire une solution optimale. Cela nécessite la comparaison de cette nouvelle étiquette par rapport à toutes les étiquettes calculées au sommet courant. Cette opération est très coûteuse en temps lorsque le nombre de solutions partielles est important. Notre objectif est de diminuer le temps de traitement de la procédure de dominance en réduisant le nombre d'étiquettes traitées et par conséquent, diminuer le temps de résolution global.

2.4.1 Programmation dynamique et problèmes de plus court chemin

Les problèmes de type plus court chemin sont des cas particuliers du problème de flot où les capacités sont unitaires et sont généralement résolus avec la programmation dynamique. Nous nous intéressons dans cette section à la Programmation Dynamique avec Correction d'Étiquettes (PDCE). Nous considérons un graphe $G = (H, A)$. Nous rappelons ci-dessous la formulation du problème de type Problème de Plus Court Chemin avec Fenêtres de Temps et Contraintes de Capacité (PPCCFTCC) :

$$(SP(\pi)) \begin{cases} \min & \sum_{(i,j)\in A}(c_{ij} - \pi_i)x_{ij} & \\ \text{s.c.} & \sum_{j\in H} x_{sj} = \sum_{j\in H} x_{jt} = 1 & \\ & \sum_{j\in H} x_{ij} - \sum_{j\in H} x_{ji} = 0 & i \in N \\ & \sum_{(i,j)\in A} d_i x_{ij} \leq Q & \\ & x_{ij} \in \{0,1\} & \forall (i,j) \in A \\ & x_{ij}(T_i + t_{ij} - T_j) \leq 0 & \forall (i,j) \in A \\ & T_j \in [a_j, b_j] & \forall j \in H \end{cases}$$

L'objectif est de trouver un chemin du sommet source s au sommet puits t, satisfaisant les contraintes de capacité et les contraintes temporelles et à moindre coût. Nous considérons le cas de graphes acycliques où les sommets sont topologiquement ordonnés de 1 à n (n étant le nombre de sommets du graphe). Le sommet source est de numéro 1 et le sommet destination est de numéro n.

Le PPCCFTCC est NP-difficile, Desrosiers et Soumis (1988) ont proposé un algorithme pseudo-polynomial pour le résoudre. Le principe de base pour sa résolution est d'associer à chaque chemin partiel une étiquette décrivant sa durée et son coût. Les chemins partiels dont l'extension ne permet pas de construire un chemin optimal sont éliminés en utilisant des règles de dominance.

Pour chaque chemin réalisable X_{sj} allant de la source s au sommet j du PPCCFTCC, on associe une étiquette, définie par un triplet (coût, temps, demande) correspondant respectivement au coût, à la durée et à la demande cumulée sur le chemin X_{sj}. Une étiquette est notée $E_j^k = (C_j^k, T_j^k, D_j^k)$ pour indiquer les caractéristiques du k-ème chemin de s à j, C_j^k représente le coût, T_j^k la durée et D_j^k la demande sur ce chemin. On dit que l'étiquette E_j^k est réalisable si et seulement si $T_j^k \leq b_j$ et $D_j^k \leq Q$.

Il existe deux types d'algorithmes de programmation dynamique :

- algorithmes à fixation d'étiquettes qui consistent à sélectionner itérativement un sommet et de le traiter définitivement (Lacomme (2003)). Ces algorithmes sont dérivés de l'algorithme de Dijkstra.
- algorithmes à correction d'étiquettes qui améliorent des étiquettes déjà calculées (Desrochers (1988)). Ces algorithmes dérivent de l'algorithme de Ford-Bellman.

Les algorithmes à fixation d'étiquettes ne permettent pas de considérer des arcs à coûts négatifs, alors que les algorithmes à correction d'étiquettes sont valables pour des graphes dont les arcs sont de longueurs quelconques.

2.4.2 Programmation dynamique par blocs

Dans ce travail, nous nous sommes fixés comme objectif la réduction de l'espace de recherche de nouvelles étiquettes efficaces au cours de la PDCE. Pour ce faire, nous allons utiliser chaque étiquette retenue E pour construire une zone qui

ne peut contenir que des étiquettes qui dominent l'étiquette E. À cet effet, aucune étiquette n'appartenant pas à cette zone ne domine E, elles peuvent donc être ignorées. Le domaine de recherche de nouvelles étiquettes efficaces se réduit donc aux zones construites suivant ce principe.

Notre objectif consiste également à appliquer la dominance uniquement sur les étiquettes dominées afin de ne pas faire de tests inutiles. Cela est possible en établissant un ordre sur les durées des étiquettes efficaces de chaque sommet. Nous détaillons ci-dessous le principe de cette méthode.

Le principe de base est de construire des blocs qui représentent des espaces susceptibles de contenir des étiquettes efficaces. Initialement ces espaces sont les domaines réalisables relativement aux fenêtres de temps. Lorsqu'une nouvelle étiquette efficace E est trouvée, de nouvelles informations sont ainsi rajoutées. Il existe un espace qui ne contient que des étiquettes dominées par l'étiquette E. Notre objectif est de caractériser ces domaines pour chaque nouvelle étiquette calculée afin de réduire l'espace de recherche de nouvelles étiquettes efficaces.

On associe à chaque sommet i, un ensemble de blocs designés par des triplets $B_i^l = (t^i_{l(inf)}, t^i_{l(sup)}, c^i_{l(sup)})$ dont les éléments correspondent respectivement à la borne inférieure de la durée, la borne supérieure de la durée et la borne supérieure du coût, l'indice l représente le numéro du bloc. On suppose que les blocs ne sont pas bornés inférieurement suivant le coût. Chaque bloc représente une zone qui ne peut contenir que des étiquettes réalisables et efficaces.

Initialement, on construit au sommet i un bloc $B_i^0 = (a_i, b_i, +\infty)$. Afin de calculer les étiquettes efficaces de ce sommet, on prolonge toutes les étiquettes efficaces des sommets prédécesseurs.

Soit $E_i^1 = (C_i^1, T_i^1)$ une nouvelle étiquette. Si E_i^1 n'appartient pas au bloc B_i^0, alors elle est non réalisable. Sinon, elle est réalisable et efficace. L'étiquette E_i^1 permet de créer un nouveau bloc (voir la figure 2.5). Le coût de E_i^1 constitue une borne supérieure sur le coût de toutes les étiquettes dont la durée appartient à l'intervalle $[T_i^1, b_i]$.

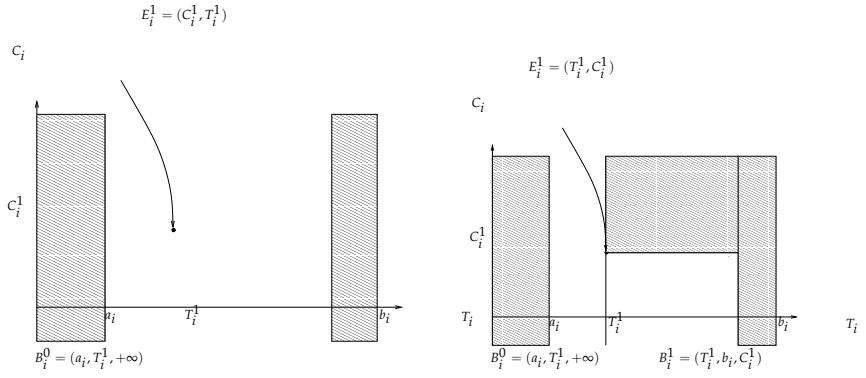


FIGURE 2.5 – *Prolongement de la première étiquette*

Le nouveau bloc $B_i^1 = (T_i^1, b_i, C_i^1)$ associé à l'étiquette E_i^1 est créé et le bloc B_i^0 est mis à jour comme suit : $B_i^0 = (a_i, T_i^1, +\infty)$ (figure 2.5).

Proposition 2. *Soit $E = (C, T)$ une nouvelle étiquette prolongée au sommet $i \in N$. Si l'étiquette E appartient à un bloc du sommet i, alors :*

1. *elle est réalisable.*
2. *elle est efficace.*

Preuve 8.

1. *Les blocs associés à un sommet $i \in N$ sont construits en fractionnant le bloc initial $B_i^0 = (a_i, b_i, +\infty)$ après chaque calcul d'une nouvelle étiquette efficace (figure 2.5), donc toute étiquette appartenant à un bloc est réalisable.*

2. *Soit $\mathcal{B}_i = \{B_i^0, ..., B_i^p, ..., B_i^q\}$ l'ensemble des blocs construits au sommet i, et ordonnés suivant l'ordre croissant des bornes inférieures des durées ($t^i_{l(inf)}$). On distingue deux types de blocs, les blocs étiquettes $B_i^1, ..., B_i^q$, construits via les étiquettes efficace $E_i^1 = (C_i^1, T_i^1), ..., E_i^q = (C_i^q, T_i^q)$ respectivement, et un bloc d'initialisation engendré par le bloc initial (le bloc B_i^0 après mise à jour). Nous distinguons les cas suivants :*

   (a) *L'étiquette $E = (C, T)$ appartient à un bloc étiquette B_i^p. Supposons que $E = (C, T)$ n'est pas efficace, i.e. est dominée par l'étiquette E_i^r, i.e. $E_i^r \leq E$ et $E_i^r \neq E$. Sachant que les blocs (les étiquettes) sont ordonnés suivant l'ordre croissant de la borne inférieure de la durée, l'étiquette E_i^r appartient à un bloc de l'ensemble $\{B_i^0, ..., B_i^{p-1}\}$. Nous avons $C_i^r \leq C$ et $C \leq C_i^p$, donc $C_i^r \leq C_i^p$, or sachant que les deux étiquettes E_i^r et E_i^p sont efficaces et que $T_i^r \leq T_i^p$, on aboutit à une contradiction.*

   (b) *Si l'étiquette $E = (C, T)$ appartient au bloc B_i^0. Supposons que $E = (C, T)$ n'est pas efficace. Sachant qu'il n'existe aucune étiquette dans le bloc B_i^0, toutes les étiquettes efficaces du sommet i sont de durées supérieures à T, donc l'étiquette E est efficace.*

Proposition 3. *Soit $E = (C, T)$ une nouvelle étiquette prolongée au sommet $i \in N$. Si l'étiquette E appartient au bloc B_i^p, alors les étiquettes dominées par E sont $E_i^{p+1}, ..., E_i^l$ qui satisfont $T > T_i^{p+1}, ..., T > T_i^l$ respectivement.*

Preuve 9.

1. *$E \in B_i^p$, alors $T \geq T_i^{p-1} > T_i^{p-2} ... > T_i^0$, il en découle qu'aucune étiquette dans l'ensemble $\{E_i^0, ..., E_i^{p-1}\}$ ne peut être dominée par E.*

2. *Les étiquettes susceptibles d'être dominées par l'étiquette E appartiennent à l'ensemble $\{E^{p+1}, ..., E^q\}$, car $T \leq T_i^{p+1} < ... < T_i^q$, il reste donc à comparer le critère coût pour identifier les étiquettes dominées par E.*
   Soient $E_i^{p+1}, ..., E_i^l$ les étiquettes qui vérifient $C \leq C_i^{p+1}, ..., C \leq C_i^l$ respectivement, ces étiquettes sont donc dominées par E. Sachant que $C > C_i^{l+1}$, l'étiquette E_i^{l+1} n'est pas dominée par E et donc les étiquettes $E_i^{l+2}, ..., E_i^q$ ne sont pas dominées par E non plus.

L'algorithme 8 présente les étapes principales de cette procédure appelée Programmation Dynamique par Blocs (PDBlocs) pour la résolution du problème de plus court chemin avec fenêtres de temps.

Le nombre d'opérations nécessaires pour la résolution du problème de plus court chemin avec fenêtres de temps est donné par Laval et al. (2006) :

Algorithme 8: Algorithme de programmation dynamique par blocs

pour $i = 1$ *à* n **faire**

1. Initialiser l'ensemble des blocs $B \leftarrow (a_i, b_i, +\infty)$
2. $Pred(i) \leftarrow$ ensemble des prédécesseurs du sommet i.
3. **pour** $i = 1$ *à* n **faire**
   (a) **pour *tout*** $j \in Pred(i)$ **faire**
      i. $EEE(j) \leftarrow$ ensemble des étiquettes efficaces.
      ii. **pour *tout*** $E = (C_j, T_j) \in EEE(j)$ **faire**
         A. *Prolongement :*
         $C \leftarrow C_j + C_{ij}$, $T \leftarrow T_j + t_{ij}$
         B. *Test de réalisabilité et de dominance :*
         si $(C, T) \in B$ **alors** , (C, T) est efficace,
         $EEE(i) \leftarrow EEE(i) \cup \{(C, T)\}$
         sinon (C, T) est non réalisable ou non efficace
         C. *Recherche d'étiquettes dominées :*
         $E_> \leftarrow \{E^p, ..., E^q\}$, tel que
         $\forall E^k = (C^k, T^k) \in E_>, T^k > T$
         parcours $\leftarrow$ vrai
         D. **pour** *((k = p à q) et (parcours = vrai))* **faire**
            si $C^k > T$ **alors** $EEE(i) \leftarrow EEE(i) \setminus E^k$
            sinon parcours $\leftarrow$ faux
         E. *Mise à jours des blocs*

$$\sum_{i=2}^{n}\sum_{j=1}^{i-1} 2S_j + \sum_{i=2}^{n} 2C_{S_i}^2$$

où S_i représente le nombre maximal d'étiquettes efficaces pour le sommet i, et est égal au nombre de toutes les étiquettes efficaces des sommets de plus petit indice, i.e. $S_i = \sum_{j=1}^{i-1} S_j$.

Le terme $\sum_{i=2}^{n}\sum_{j=1}^{i-1} 2S_j$ représente le nombre de comparaisons dans la phase de réalisabilité. Pour chaque sommet i, chaque étiquette d'un sommet prédécesseur j est comparée par rapport aux ressources temps et capacité. Le terme $\sum_{i=2}^{n} 2C_{S_i}^2$ représente le nombre de tests effectués dans la phase de dominance, dans notre cas, la dominance est appliquée suivant les critères coût et temps.

Le nombre d'opérations dans les tests de réalisabilité et d'efficacité dans la PDBlocs est équivalent à :

$$\sum_{i=2}^{n}\sum_{j=1}^{i-1} S_j log(S_i),$$

lorsque l'on suppose que la recherche d'un bloc auquel appartient une étiquette est effectuée avec une méthode dichotomique. Pour chaque étiquette prolongée à un sommet i, on parcourt au pire des cas toutes les étiquettes efficaces du sommet i. La recherche d'étiquettes dominées est effectuée avec $\sum_{i=2}^{n} C_{S_i}^2$ comparaisons au maximum.

Le nombre total d'opérations dans l'algorithme 8 pour la résolution du PPCCFTCC est donc donné par :

$$\sum_{i=2}^{n}\sum_{j=1}^{i-1} S_j log(S_i) + \sum_{i=2}^{n} C_{S_i}^2$$

D'après cette formule, lors du calcul d'une nouvelle étiquette efficace dans chaque sommet du graphe, on parcourt au pire des cas deux fois toutes les étiquettes efficaces du sommet traité. D'après cette étude, le nombre d'opérations de la PDBlocs est supérieur à celui de PDCE.

Remarque 2. *Soit E une nouvelle étiquette calculée dans un sommet i du graphe, qui domine toutes les étiquettes efficaces de ce sommet. L'étiquette E appartient donc au premier bloc de i (i.e. le bloc dont la borne inférieure du temps est égale à a_i). Le nombre d'opérations nécessaires dans le test de réalisabilité et d'efficacité se réduit donc à 1. Dans ce cas, le nombre d'opérations de la PDBlocs est inférieur à celui de PDCE.*

On déduit à travers cette étude que la complexité de la PDBlocs est supérieure à celle de PDCE, en théorie, mais peut être inférieure dans la pratique. Nous présentons dans la section suivante une étude comparative entre la PDCE et la PDBlocs sur le plan expérimental.

2.4.3 Expérimentations sur le problème de plus court chemin avec fenêtres de temps

Nous avons implémenté cette méthode dans la plateforme logicielle COGiTO (Column Generation in Transport Optimization). Afin d'étudier les performances de cette technique sur des instances du PPCCFTCC, nous l'appliquons sur des instances du problème de tournées de véhicules avec fenêtres de temps et capacité (CVRPTW) de la littérature et générées aléatoirement et reportons sur le tableau 2.18 les résultats associés à la résolution du sous-problème de PCCFTCC lors de la première itération de la génération de colonnes avec la PDCE d'un coté et la PDBlocs de l'autre. Nous observons le nombre d'étiquettes traitées (étiquettes sur lesquelles l'opération de dominance a été appliquée), le nombre total d'étiquettes comparées avec une nouvelle étiquette efficace dans la phase de dominance, et le temps de résolution global (tR).

La PDBlocs, voir tableau 2.18, diminue le nombre d'étiquettes traitées de 79% par rapport à la PDCE, cela est dû au test d'appartenance à un bloc qui permet de ne tester la dominance que sur des étiquettes efficaces. La PDBlocs permet aussi de diminuer considérablement le nombre d'étiquettes comparées dans la procédure de dominance (plus de 99%) par rapport à la PDCE. Notons qu'au gain obtenu par la PDBlocs, s'ajoute un surcoût de calcul et de mise à jours des blocs, mais le compromis entre ces deux critères est largement satisfaisant, le gain obtenu sur le temps de résolution de la PDCE est de l'ordre de 88%.

Nous étudions dans cette partie l'apport de la programmation dynamique par blocs sur les performances de la génération de colonnes. Nous comparons sur le tableau 2.19 les temps de résolution et le nombre total d'étiquettes traitées avec la Génération de Colonnes Intensifiée (GCI) (où l'on ajoute toutes les colonnes de coût réduit négatif à chaque itération) et la Génération de Colonnes utilisant la programmation dynamique par Blocs (GCBlocs).

Comme attendu, le nombre total d'étiquettes traitées avec la GCBlocs est très réduit par rapport à la GCI. Le gain obtenu sur le nombre d'étiquettes traitées par rapport à la GCI est de 84%. Ce pourcentage est plus important dans la résolution

Instances de	*C*101_25			*C*101_50		
Solomon	tR	nbEtiq	nbEtiqComp	tR	nbEtiq	nbEtiqComp
PDCE	0,1”	343	680	3,3”	1 673	2 695
PDBlocs	0.04”	79	4	0,2”	204	15
Instances de	*C*101			*C*1_2_1		
Solomon	tR	nbEtiq	nbEtiqComp	tR	nbEtiq	nbEtiqComp
PDCE	72,0”	5 941	10 510	2 526,0”	39 922	1 946 876
PDBlocs	3,2”	491	68	87,0”	1 036	11 425
Instances	*G*_100			*G*_120		
générées	tR	nbEtiq	nbEtiqComp	tR	nbEtiq	nbEtiqComp
PDCE	186,0”	200 586	21 700 018	426,0”	335 421	116 011 504
PDBlocs	18,9”	14 446	98 912	24,1”	92 325	123 413
Instances	*G*_140			*G*_160		
générées	tR	nbEtiq	nbEtiqComp	tR	nbEtiq	nbEtiqComp
PDCE	462,0”	454 729	21 220 354	138,0”	105 478	11 836 750
PDBlocs	27,0”	17 366	94 812	17,1”	57 758	115 377

tR : temps de résolution.
nbEtiq : nombre total d'étiquettes traitées.
nbEtiqComp : nombre total d'étiquettes comparées.

TABLE 2.18 – *Programmation dynamique par blocs*

Instances de	*C*101_25		*C*101_50		*C*101		*C*1_2_1	
Solomon	tGC	nbEtiq	tGC	nbEtiq	tGC	nbEtiq	tGC	nbLb
GCI	2,1”	9 112	31,7”	86 073	5,0’	584 810	150,9’	4 712 427
GCBlocs	0,1”	1 879	18,9”	11 653	3,2’	42 173	107,5’	180 389
Instances	*G*_100		*G*_120		*G*_140		*G*_160	
générées	tGC	nbEtiq	tGC	nbEtiq	tGC	nbEtiq	tGC	nbLb
GCI	12,5’	446 080	39,1’	741 413	35,5’	778 103	20,4’	206 290
GCBlocs	6,7’	40 761	15,2’	97 228	15,6’	78 116	13,7’	87 992

tGC : temps de résolution de la génération de colonnes.
nbLb : nombre total d'étiquettes traitées.

TABLE 2.19 – *Programmation dynamique par blocs dans la génération de colonnes*

du premier sous-problème (99%), car le nombre d'étiquettes calculées lors de la première itération est plus important que lors des autres itérations. Le temps de résolution global de la GCI est ainsi réduit de 48% en utilisant la PDBlocs pour résoudre les sous-problèmes.

2.4.4 Conclusion

Nous nous sommes intéressés dans cette section à la résolution du PPCCFTCC avec la programmation dynamique. Nous avons proposé d'ordonner les étiquettes selon l'ordre croissant de la durée. Cela permet de décomposer l'espace réalisable associé à chaque sommet du graphe en blocs, chacun ne pouvant contenir que des étiquettes réalisables et efficaces. Cette méthode a été utilisée dans la génération de colonnes pour la résolution des sous-problèmes du problème de tournées de véhicules avec fenêtres de temps. Le gain moyen obtenu sur le temps de résolution de la génération de colonnes lorsque les sous-problèmes sont résolus avec la programmation dynamique à correction d'étiquettes est de l'ordre de 48%.

CONCLUSION DU CHAPITRE

Nous nous sommes intéressés aux problèmes de (multi)-flots, à leur formulation, leur reformulation, leur résolution et aux réductions que l'on peut apporter au graphe pour résoudre plus efficacement ces problèmes. Nous avons présenté plusieurs travaux autour de cette problématique avec des résultats théoriques et expérimentaux, ainsi que des applications sur des domaines variés tels que le traitement d'images et la fouille de données biomédicales. Ces travaux permettent d'envisager de nombreuses perspectives, parmi lesquelles on peut citer par exemple :

- la recherche d'autres topologies de graphes pour lesquelles les problèmes de multiflots et/ou de multicoupes sont polynomiaux et donner des algorithmes dédiés efficaces pour les résoudre ;
- l'élaboration de nouvelles approches de réduction et de résolution des problèmes de traitement d'images, comme la segmentation, dans le cas multi-labels (segmentation de plusieurs objets) qui induisent alors des problèmes de type "multiway cut" ;
- l'adaptation et le développement d'approches similaires pour la recherche de communautés dans les réseaux sociaux.
- l'extension de la programmation dynamique par blocs aux cas à plus de deux ressources ;

Les problèmes de type (multi-)flots se retrouvent souvent comme sous-problèmes de problèmes plus complexes, comme par exemple les problèmes de tournées de véhicules et de routage en général, quand ces derniers sont résolus via des méthodes impliquant des relaxations et/ou des décompositions, étudiées dans le chapitre suivant.

3 REFORMULATION, RELAXATION ET RÉOPTIMISATION EN PROGRAMMATION MATHÉMATIQUE

SOMMAIRE

DANS ce chapitre, nous présentons des approches issues de la programmation mathématique afin d'élaborer des méthodes de résolution exactes et/ou approchées pour des problèmes encore mal résolus. Différentes formulations des problèmes étudiés seront considérées afin de construire des schémas de résolution complémentaires. Ceux-ci s'appuieront notamment sur des relaxations et décompositions des modélisations proposées, qui pourront ensuite être insérées dans des schémas de séparation et d'évaluation progressive. Nous introduirons également le concept de réoptimisation qui nous fournira un cadre de travail pour le développement d'approches innovantes dans différents contextes. Ce chapitre est composé de deux parties. La première partie (section 3.1) se concentrera principalement sur

la méthode de décomposition de Dantzig-Wolfe et sur la résolution des problèmes induits par ce type décomposition via la génération de colonnes. Nous introduirons notamment la notion de diversification et nous nous placerons également dans le cadre de la réoptimisation lorsque l'on doit faire face à des modifications de certaines données. Nous étudierons particulièrement les problèmes de type localisation et routage (tournées de véhicules) dans cette partie. La seconde partie de ce chapitre (section 3.2) concerne l'introduction de techniques de réoptimisation et de reformulation dans le cadre de la programmation quadratique pour des problèmes de type sac à dos.

3.1 DÉCOMPOSITION ET GÉNÉRATION DE COLONNES

Nous nous intéressons dans cette section à la décomposition de Dantzig-Wolfe et au schéma de génération de colonnes qui en résulte. Nous présentons en premier lieu les liens qui existent entre la décomposition Lagrangienne et la décomposition de Dantzig-Wolfe. Nous étudions ensuite, pour des problèmes de tournées de véhicules, la diversification au sein du schéma de génération de colonnes afin d'accélérer sa convergence ; ainsi que la réoptimisation au sein de ce processus itératif afin de diminuer le temps de résolution. Nous nous concentrons ensuite sur la résolution exacte et approchée de problème de localisation et routage et nous terminons cette section par l'étude du concept de réoptimisation locale dans lequel les problèmes sont résolus par génération de colonnes.

3.1.1 Décompositions Lagrangienne et de Dantzig-Wolfe

On peut écrire un programme linéaire en nombres entiers sous la forme suivante sans perte de généralité :

$$(P)\begin{cases} \max & c^t x & & \\ s.c. & Ax & = & a \\ & Bx & = & b \\ & x & \in & X \end{cases}$$

avec $c \in \mathbb{R}^n$, A une matrice $m \times n$, B une matrice $p \times n$, $a \in \mathbb{R}^m$, $b \in \mathbb{R}^p$ et $X \subseteq \mathbb{N}^n$.

Cett section rappelle les liens existants entre la Relaxation Lagrangienne (LR) et la Décomposition de Dantzig-Wolfe (DWD) (Dantzig et Wolfe (1960)) et établit une relation entre la Décomposition Lagrangienne (DL) et DWD pour en dériver un nouveau problème maître de type DW.

L'équivalence entre DWD et LR est bien connue (Lemaréchal (2003)). Résoudre un progralmme linéaire par la génération de colonnes (CG), en utilisant la DWD, est similaire à la résolution du dual Lagrangien par la méthode des plans coupants de Kelley (Kelley (1960)). Ce travail rappelle les résultats existants et les étend à la LD, qui peut être vue comme une DWD spécifique, afin de montrer la supériorité de la nouvelle borne obtenue.

La section suivante rappelle les principes de LR, LD et DWD. Nous montrons ensuite les relations entre LD et DWD, et donnons une nouvelle preuve de la dominance de la borne de LD sur LR. Nous présentons les deux modèles de DW pour le sac à dos 0-1 bi-dimensionnel (0-1_BKP) et pour l'affectation généralisée (GAP) avant de présenter les résultats expérimentaux sur ces deux problèmes.

Duaux Lagrangiens et décomposition de Dantzig-Wolfe

Nous rappelons brièvement le principe de la dualité Lagrangienne et ses liens avec la DWD et la CG.

Dual de la relaxation Lagrangienne La LR consiste à omettre quelques contraintes compliquantes ($Ax = a$) et à les incorporer dans la fonction objectif en utilisant un multiplicateur de Lagrange $\pi \in \mathbb{R}^m$. Nous obtenons la relaxation

suivante :

$$(LR(\pi)) \begin{cases} \max & c^t x + \pi^t (a - Ax) \\ \text{s. c.} & Bx = b \\ & x \in X. \end{cases}$$

Pour tout $\pi \in \mathbb{R}^m$, la valeur de $(LR(\pi))$ est un majorant de $v(P)$. Le meilleur est donné par le dual de la LR :

$$\begin{array}{rcl} (LRD) & \equiv & \min_{\pi \in \mathbb{R}^m} (LR(\pi)) \\ & \equiv & \min_{\pi \in \mathbb{R}^m} \max_{\{x \in X, Bx = b\}} c^t x + \pi^t (a - Ax). \end{array}$$

Soient $X_B = \{x \in X | Bx = b\}$ et $Conv(X_B)$ son enveloppe convexe, supposée bornée. On note par $x^{(k)} (k \in \{1, \ldots, K\})$ les points extrêmes de $Conv(X_B)$. Ainsi, (LRD) peut être reformulé de la manière suivante :

$$\begin{array}{rcl} (LRD) & \equiv & \min_{\pi \in \mathbb{R}^m} \ \max_{k=1,\ldots,K} \ c^t x^{(k)} + \pi^t (a - Ax^{(k)}) \\ & \equiv & \begin{cases} \min & z \\ s.c. & z + \pi^t (Ax^{(k)} - a) \geq c^t x^{(k)}, \ k = 1, \ldots, K \\ & \pi \in \mathbb{R}^m, \, z \in \mathbb{R}. \end{cases} \end{array}$$

Cette nouvelle formulation contient potentiellement un nombre exponentiel de contraintes, égal à K. La méthode des plans coupants de Kelley (Kelley (1960)) considère un ensemble réduit de ces contraintes qui composent le problème restreint. Les coupes (contraintes) sont ajoutées à chaque itération jusqu'à ce que l'optimum soit atteint.

Dual de la décomposition Lagrangienne Il est bien connu que l'efficacité d'un schéma de séparation et d'évaluation dépend fortement de la qualité des bornes. Afin d'améliorer celles issues de la relaxation Lagrangienne, Guignard et Kim (Guignard et Kim (1987a), Guignard et Kim (1987b)) ont proposé d'utiliser la décomposition Lagrangienne (LD). Dans cette approche, des contraintes de copie sont ajoutées à la formulation (P) pour aboutir à un problème équivalent :

$$\begin{cases} \max & c^t x & & \\ \text{s. c.} & Ay & = & a \\ & Bx & = & b \\ & x & = & y \\ & x \in X, \, y \in Y, & \text{avec } Y \supseteq X & \end{cases}$$

où les variables de copie permettent de séparer le problème initial en deux sous-problèmes indépendants après application de la LR sur les contraintes de copie $x = y$:

$$(LD(w)) \qquad \begin{cases} \max & c^t x + w^t (y - x) \\ \text{s. c.} & Ay = a \\ & Bx = b \\ & x \in X, \, y \in Y, \end{cases}$$

où $w \in \mathbb{R}^n$ sont les variables duales associées aux contraintes de copie. On obtient les deux sous-problèmes indépendants suivants :

$$(LD_y(w)) \begin{cases} \max & w^t y & & \\ \text{s. c.} & Ay & = & a \\ & y & \in & Y \end{cases} \qquad \text{et} \qquad (LD_x(w)) \begin{cases} \max & (c - w)^t x & & \\ \text{s. c.} & Bx & = & b \\ & x & \in & X \end{cases}$$

Le dual de la LD est donné par

$$(LDD) \min_{w \in \mathbb{R}^n} v(LD(w))$$

où

$$v(LD(w)) = \max\{w^t y \mid y \in Y_A\} + \max\{(c-w)^t x \mid x \in X_B\}$$

avec

$$Y_A = \{y \mid Ay = a,\ y \in Y\} \qquad X_B = \{x \mid Bx = b,\ x \in X\}.$$

Ce dual peut être réécrit comme :

$$(LDD) \quad \begin{cases} \min & \max(c-w)^t x & + & \max w^t y \\ w \in \mathbb{R}^n & x \in X_B & & y \in Y_A. \end{cases}$$

Nous supposons que les enveloppes convexes des ensembles Y_A et X_B sont bornées. Nous notons par $x^{(k)}, k \in \{1,\dots,K\}$ les points extrêmes de X_B et par $y^{(l)}, l \in \{1,\dots,L\}$ ceux de Y_A. Nous obtenons la formulation suivante :

$$(LDD) \quad \begin{cases} \min & \max(c-w)^t x^{(k)} & + & \max w^t y^{(l)} \\ w \in \mathbb{R}^n & k = 1,\dots,K & & l = 1,\dots,L \end{cases}$$

qui peut être exprimée sous la forme linéaire équivalente :

$$(LDD) \quad \begin{cases} \min & z_1 + z_2 & \\ & z_1 \geq (c-w)^t x^{(k)} & k = 1,\dots,K \\ & z_2 \geq w^t y^{(l)} & l = 1,\dots,L \\ & w \in \mathbb{R}^n & z_1, z_2 \in \mathbb{R}. \end{cases}$$

Le théorème suivant rappelle la relation de dominance entre (P), (LRD), (LDD) et (LP) qui est la relaxation continue de (P).

Theorème 4. *(Guignard et Kim (1987a), Guignard et Kim (1987b))* $v(P) \leq v(LDD) \leq v(LRD) \leq v(LP)$.

Décomposition de Dantzig-Wolfe et génération de colonnes L'idée fondatrice de la Décomposition de Dantzig-Wolfe (DWD) (Dantzig et Wolfe (1960)) est de reformuler le problème en substituant les variables originales par une combinaison convexe des points extrêmes du polyèdre correspondant à la sous-structure de la formulation.

On sait que

$$\forall x \in Conv(X_B), \qquad x = \sum_{k=1}^{K} \lambda_k x^{(k)}$$

avec $\sum_{k=1}^{K} \lambda_k = 1$, $\lambda_k \geq 0$, $\forall k \in 1,\dots,K$.

En substituant x dans (P) nous obtenons le problème maître de la DWD :

$$(MP) \begin{cases} \max & \sum_{k=1}^{K} (c^t x^{(k)}) \lambda_k & & \\ s.c. & \sum_{k=1}^{K} (Ax^{(k)}) \lambda_k & = & a \\ & \sum_{k=1}^{K} \lambda_k & = & 1 \\ & \lambda_k & \geq & 0, \qquad k = 1,\dots,K. \end{cases}$$

(MP) contient $m+1$ contraintes et (potentiellement) un très grand nombre de variables (i.e. le nombre K de points extrêmes de $Conv(X_B)$).

Remarque 3. *Puisque* (LRD) *est le dual de* (MP), $v(LRD) = v(MP)$ *(Lemaréchal (2003)).*

CG consiste à générer itérativement un sous-ensemble des points extrêmes de X_B pour déterminer une solution optimale de (MP) en résolvant alternativement :

– un *Problème Maître Restreint* de la DWD sur le sous-ensemble $\mathcal{K}$ de $\{1,\ldots,K\}$:

$$(RMP) \qquad \left\{ \begin{array}{lrcll} \max & \sum_{k\in\mathcal{K}}(c^t x^{(k)})\lambda_k & & & \\ s.c. & \sum_{k\in\mathcal{K}}(Ax^{(k)})\lambda_k & = & a & \\ & \sum_{k\in\mathcal{K}}\lambda_k & = & 1 & \\ & \lambda_k & \geq & 0\,, & \quad k\in\mathcal{K} \end{array} \right.$$

– un problème auxiliaire :

$$(SP) \left\{ \begin{array}{ll} \max & c^t x - \pi^t A x - \pi_0 \\ \text{s. c.} & Bx = b \\ & x \in X \end{array} \right.$$

où $(\pi, \pi_0) \in \mathbb{R}^m \times \mathbb{R}$ sont les variables duales fournies par la résolution de (RMP). La solution de (SP) est incorporée (comme une colonne) dans (RMP) si sa valeur est non négative.

Ce processus se termine quand il n'y a plus de variables dans $\{1,\ldots,K\}\backslash\mathcal{K}$ avec un coût réduit positif.

Décompositions Lagrangienne et de Dantzig-Wolfe

Cette section est dédiée à la décomposition Lagrangienne. Nous établissons la relation entre LD, DWD et CG. Nous considérons le problème maître de DW suivant :

$$(MPD) \quad \left\{ \begin{array}{ll} \max & \sum_{k=1}^{K}(cx^{(k)})\lambda_k \\ & \sum_{k=1}^{K} x^{(k)}\lambda_k - \sum_{l=1}^{L} y^{(l)}\gamma_l = 0 \\ & \sum_{k=1}^{K}\lambda_k = 1 \\ & \sum_{l=1}^{L}\gamma_l = 1 \\ & \lambda_k \geq 0, k = 1,\ldots,K,\ \gamma_l \geq 0, l = 1,\ldots,L \end{array} \right.$$

où $x^{(k)}, k \in \{1,\ldots,K\}$ sont les points extrêmes de X_B et $y^{(l)}, l \in \{1,\ldots,L\}$ sont ceux de Y_A.

Lemme 1. *La valeur de ce problème maître* (MPD) *fournit un meilleur majorant de* $v(P)$ *que la valeur de la DWD classique* (MP).

Preuve 10.

$$v(MPD) \quad = \quad \left\{ \begin{array}{ll} \max & \sum_{k=1}^{K}(cx^{(k)})\lambda_k \\ & \sum_{k=1}^{K} x^{(k)}\lambda_k - \sum_{l=1}^{L} y^{(l)}\gamma_l = 0 \\ & \sum_{k=1}^{K}\lambda_k = 1 \\ & \sum_{l=1}^{L}\gamma_l = 1 \\ & \lambda_k \geq 0, k = 1,\ldots,K,\ \gamma_l \geq 0, l = 1,\ldots,L. \end{array} \right.$$

Par dualité

$$v(MPD) \quad = \quad \begin{cases} \min & z_1 + z_2 \\ & z_1 + w^t x^{(k)} \geq cx^k,\ k = 1, \ldots, K \quad (1) \\ & z_2 - w^t y^{(l)} \geq 0,\ \ l = 1, \ldots, L \qquad (2) \\ & w \in \mathbb{R}^n,\ z_1, z_2 \in \mathbb{R} \end{cases}$$

Si nous ne considérons qu'un sous-ensemble des multiplicateurs $w \in \mathbb{R}^n$ tels que $w^t = \pi^t A$, avec π un vecteur de $\mathbb{R}^m$, et que l'on substitue dans (1) et (2), nous obtenons le problème suivant :

$$\begin{cases} \min & z_1 + z_2 \\ & z_1 + \pi^t A x^{(k)} \geq cx^k,\ k = 1, \ldots, K \\ & z_2 - \pi^t A y^{(l)} \geq 0,\ l = 1, \ldots, L \\ & w \in \mathbb{R}^n,\ z_1, z_2 \in \mathbb{R} \end{cases}$$

pour lequel le dual est :

$$\begin{cases} \max & \sum_{k=1}^{K} (cx^{(k)}) \lambda_k \\ & \sum_{k=1}^{K} A x^{(k)} \lambda_k - \sum_{l=1}^{L} A y^{(l)} \gamma_l = 0 \\ & \sum_{k=1}^{K} \lambda_k = 1 \\ & \sum_{l=1}^{L} \gamma_l = 1 \\ & \lambda_k \geq 0, k = 1, \ldots, K,\ \gamma_l \geq 0, l = 1, \ldots, L. \end{cases}$$

Comme $y^{(l)}, l \in \{1, \ldots, L\}$ sont les points extrêmes de Y_A, nous retrouvons $Ay^{(l)} = a$, et puisque $\sum_l \gamma_l = 1$, nous obtenons le problème (MP). Ainsi $v(MPD) \leq v(MP)$.

Remarque 4. *Si $n > m$, l'ensemble $\{\pi^t A, \pi \in \mathbb{R}^m\} \subsetneq \mathbb{R}^n$ et $v(MPD)$ est donc strictement meilleur que $v(MP)$.*

Remarque 5. *Comme (LDD) (resp. (LRD)) est le dual de (MPD) (resp. (MP)), on peut établir que*

$$v(MPD) = v(LDD) = \min_{w \in \mathbb{R}^n} v(LD(w)) \leq \min_{\pi^t \in \mathbb{R}^m} v(LD(\pi^t A))$$

et

$$\min_{\pi^t \in \mathbb{R}^m} v(LD(\pi^t A)) = \min_{\pi \in \mathbb{R}^m} v(LR(\pi)) = v(LRD) = v(MP).$$

Cette approche fournit une preuve alternative à la dominance de la LD sur la LR.

Modèles de décomposition

Cette section est dévolue à l'application de ce nouveau modèle de la DWD sur deux problèmes classiques d'optimisation combinatoire : le problème du sac à dos bidimensionnel en 0-1 et le problème d'affectation généralisée.

Le problème du sac à dos bidimensionnel en 0-1 Ce problème consiste à sélectionner un sous-ensemble d'objets tel que le profit des objets sélectionnés soit maximal en respectant deux contraintes de capacité. La formulation classique de ce problème est donnée par :

$$(0-1\_BKP)\begin{cases}\max & \sum_{i=1}^{n} c_i x_i \\ \text{s. c.} & \sum_{i=1}^{n} a_i x_i \leq A \\ & \sum_{i=1}^{n} b_i x_i \leq B \\ & x_i \in \{0,1\},\ \ i=1,\ldots,n\end{cases}$$

avec n le nombre d'objets, les coefficients a_i $(i=1,\ldots,n)$, b_i $(i=1,\ldots,n)$ et c_i $(i=1,\ldots,n)$ sont des entiers positifs et A et B sont des entiers tels que $max\{a_i : i=1,\ldots,n\} \leq A < \sum_{i=1,\ldots,n} a_i$ et $max\{b_i : i=1,\ldots,n\} \leq B < \sum_{i=1,\ldots,n} b_i$.

Le problème maître classique de Dantzig-Wolfe est donné par :

$$\begin{cases}\max & \sum_{k=1}^{K}(\sum_{i=1}^{n} c_i x_i^{(k)})\lambda_k & & \\ s.c. & \sum_{k=1}^{K}(\sum_{i=1}^{n} a_i x_i^{(k)})\lambda_k & \leq & A \\ & \sum_{k=1}^{K} \lambda_k & = & 1 \\ & \lambda_k & \geq & 0, \qquad k=1,\ldots,K\end{cases}$$

avec $x^{(k)}, k=1,\ldots,K$ les points extrêmes de $Conv(\{x_i \in \{0,1\} | \sum_{i=1}^{n} b_i x_i \leq B,\ \ i=1,\ldots,n\})$; et le problème auxiliaire est :

$$\begin{cases}\min & \sum_{i=1}^{n}(c_i - \pi a_i)x_i - \pi A \\ \text{s. c.} & \sum_{i=1}^{n} b_i x_i \leq B \\ & x_i \in \{0,1\}, i=1,\ldots,n.\end{cases}$$

Le problème maître associé à la décomposition Lagrangienne est donné par :

$$\begin{cases}\max & \sum_{k=1}^{K}(\sum_{i=1}^{n} c_i x_i^{(k)})\lambda_k \\ & \sum_{k=1}^{K}(\sum_{i=1}^{n} x_i^{(k)})\lambda_k - \sum_{l=1}^{L}(\sum_{i=1}^{n} y_i^{(l)})\gamma_l = 0 \\ & \sum_{k=1}^{K} \lambda_k = 1 \\ & \sum_{l=1}^{L} \gamma_l = 1 \\ & \lambda_k \geq 0, k=1,\ldots,K,\ \gamma_l \geq 0, l=1,\ldots,L\end{cases}$$

avec $x^{(k)}, k=1,\ldots,K$ (resp. $y^{(l)}, l=1,\ldots,L$) les points extrêmes de $Conv(\{x_i \in \{0,1\}, i=1,\ldots,n | \sum_{i=1}^{n} b_i x_i \leq B\})$ (resp. $Conv(\{y_i \in \{0,1\}, i=1,\ldots,n | \sum_{i=1}^{n} a_i y_i \leq A\})$) ;

et les problèmes auxiliaires sont :

$$\begin{cases}\min & \sum_{i=1}^{n} u_i y_i \\ \text{s. c.} & \sum_{i=1}^{n} a_i y_i \leq A \\ & y_i \in \{0,1\}, i=1,\ldots,n\end{cases}$$

et

$$\begin{cases}\min & \sum_{i=1}^{n}(c_i - u_i)x_i \\ \text{s. c.} & \sum_{i=1}^{n} b_i x_i \leq B \\ & x_i \in \{0,1\}, i=1,\ldots,n\end{cases}$$

avec $x_i,\ \ i=1,\ldots,n$ et $y_i,\ \ i=1,\ldots,n$ qui sont égaux à 1 si l'objet i est mis dans le sac.

Le problème d'affectation généralisée Il consiste à trouver le profit maximal tout en affectant T tâches à I agents tel que chaque tâche soit affectée à exactement un agent et chaque agent est contraint par une capacité (Martello et Toth (1992)). La formulation standard est la suivante :

$$\begin{cases} \max & \sum_i \sum_t c_{it} x_{it} \\ \text{s. c.} & \sum_i x_{it} = 1, \qquad t = 1, \ldots, T \\ & \sum_t r_{it} x_{it} \leq b_i, \qquad i = 1, \ldots, I \\ & x_{it} \in \{0,1\}, \qquad i = 1, \ldots, I, t = 1, \ldots, T \end{cases}$$

Deux décompositions classiques de Dantzig-Wolfe sont possibles, en relâchant soit les contraintes d'affectation, soit les contraintes de capacité.

Le premier problème maître classique de Dantzig-Wolfe est donné par :

$$\begin{cases} \max & \sum_{k=1}^{K} (\sum_i \sum_t c_{it} x_{it}^{(k)}) \lambda_k & & \\ s.c. & \sum_{k=1}^{K} (\sum_i x_{it}^{(k)}) \lambda_k & = & 1, \qquad t = 1, \ldots, T \\ & \sum_{k=1}^{K} \lambda_k & = & 1 \\ & \lambda_k & \geq & 0, \qquad k = 1, \ldots, K \end{cases}$$

où $x^{(k)}, k = 1, \ldots, K$, sont les points extrêmes de $Conv(\{x_{it} \in \{0,1\} | \sum_t r_{it} x_{it} \leq b_i, \ i = 1, \ldots, I\})$; et le problème auxiliaire associé est :

$$\begin{cases} \min & \sum_i \sum_t (c_{it} - \pi_t) x_{it} - \sum_t \pi_t \\ \text{s. c.} & \sum_t r_{it} x_{it} \leq b_i, \qquad i = 1, \ldots, I \\ & x_{it} \in \{0,1\}, \qquad i = 1, \ldots, I, t = 1, \ldots, T. \end{cases}$$

Le second problème maître classique de Dantzig-Wolfe est donné par :

$$\begin{cases} \max & \sum_{l=1}^{L} (\sum_i \sum_t c_{it} y_{it}^{(l)}) \gamma_l & & \\ s.c. & \sum_{l=1}^{L} (\sum_t r_{it} y_{it}^{(l)}) \gamma_l & \leq & b_i, \qquad i = 1, \ldots, I \\ & \sum_{l=1}^{L} \gamma_l & = & 1 \\ & \gamma_l & \geq & 0, \qquad l = 1, \ldots, L \end{cases}$$

où $y^{(l)}, l = 1, \ldots, L$ sont les points extrêmes de $Conv(\{y_{it} \in \{0,1\} | \sum_i y_{it} = 1, \ t = 1, \ldots, T\})$; et le problème auxiliaire associé est :

$$\begin{cases} \min & \sum_i \sum_t (c_{it} - \pi_i) y_{it} - \sum_i \pi_i \\ \text{s. c.} & \sum_i y_{it} = 1, \qquad t = 1, \ldots, T \\ & y_{it} \in \{0,1\}, \qquad i = 1, \ldots, I, t = 1, \ldots, T \end{cases}$$

Le problème maître associé à la décomposition Lagrangienne est donné par :

$$\begin{cases} \max & \sum_{k=1}^{K} (\sum_i \sum_t c_{it} x_{it}^{(k)}) \lambda_k \\ & \sum_{k=1}^{K} (\sum_i \sum_t x_{it}^{(k)}) \lambda_k - \sum_{l=1}^{L} (\sum_i \sum_t y_{it}^{(l)}) \gamma_l = 0 \\ & \sum_{k=1}^{K} \lambda_k = 1 \\ & \sum_{l=1}^{L} \gamma_l = 1 \\ & \lambda_k \geq 0, \qquad k = 1, \ldots, K \\ & \gamma_l \geq 0, \qquad l = 1, \ldots, L \end{cases}$$

où $x^{(k)}, k = 1, \ldots, K$ (resp. $y^{(l)}, l = 1, \ldots, L$), sont les points extrêmes de $Conv(\{x_{it} \in \{0,1\} | \sum_t r_{it} x_{it} \leq b_i, \ i = 1, \ldots, I\})$ (resp. $Conv(\{y_{it} \in \{0,1\} | \sum_i y_{it} = 1, \ t = 1, \ldots, T\})$) ;

et les problèmes auxiliaires sont :

$$\begin{cases} \min & \sum_i \sum_t u_{it} y_{it} \\ \text{s. c.} & \sum_i y_{it} = 1, \qquad t = 1, \ldots, T \\ & y_{it} \in \{0,1\}, \qquad i = 1, \ldots, I, t = 1, \ldots, T \end{cases}$$

et

$$\begin{cases} \min & \sum_i \sum_t (c_{it} - u_{it}) x_{it} \\ \text{s. c.} & \sum_t r_{it} x_{it} \leq b_i, \qquad i = 1, \ldots, I \\ & x_{it} \in \{0,1\}, \qquad i = 1, \ldots, I, t = 1, \ldots, T. \end{cases}$$

où $x_{it},\ i = 1, \ldots, I,\ t = 1, \ldots, T$ et $y_{it},\ i = 1, \ldots, I,\ t = 1, \ldots, T$ sont égaux à 1 si la tâche t est affectée à l'agent i.

Expériences numériques

Cette section est consacrée à une comparaison expérimentale entre la LD et la LR quand ils sont résolus par la génération de colonnes. Nous considérons les deux problèmes définis dans la section précédente : le problème du sac à dos bidimensionnel en 0-1 et le problème d'affectation généralisée.

Nous considérons 6 instances du problème de sac à dos bidimensionnel en 0-1 tirés de la OR-Library. Le tableau 3.1 présente une étude comparative entre la résolution par CG des formulations de la LD et de la LR (notées CG_LD et CG_LR respectivement). Les problèmes maître et auxiliaire sont résolus avec CPLEX11.2.

TABLE 3.1 – *Relaxation et décomposition Lagrangiennes pour le (0-1_BKP)*

	WEING1						WEING2					
	vR	%vE	Iter	tG	tSP	tMP	vR	%vE	Iter	tG	tSP	tMP
CG_LR	141 388.50	0.1	6	0.12	0.12	0.00	130 883.00	0.0	1	0.01	0.01	0.00
CG_LD	141 383.00	0.1	136	9.55	8.72	0.24	130 883.00	0.0	157	13.61	12.56	0.40
	WEING3						WEING4					
	vR	%vE	Iter	tG	tSP	tMP	vR	%vE	Iter	tG	tSP	tMP
CG_LR	97 613.92	2.0	5	0.13	0.11	0.00	122 321.58	2.5	7	0.08	0.06	0.01
CG_LD	95 677.00	0.0	142	11.42	10.64	0.25	119 337.00	0.0	156	12.68	11.54	0.33
	WEING5						WEING6					
	vR	%vE	Iter	tG	tSP	tMP	vR	%vE	Iter	tG	tSP	tMP
CG_LR	98 796.00	0.0	1	0.01	0.00	0.01	130 697.80	0.1	6	0.05	0.05	0.00
CG_LD	98 796.00	0.0	77	3.51	2.99	0.16	130 623.00	0.0	162	12.47	11.51	0.33

vR : la valeur de la relaxation.
%vE : le gap entre les valeurs de la relaxation et de l'optimum.
Iter : nombre d'itérations.
tG : le temps de résolution global (s).
tSP : le temps de résolution des problèmes auxiliaires (s).
tM : le temps de résolution cumulé des problèmes maîtres (s).

L'optimalité pour CG_LR et CG_LD est atteinte pour toutes les instances. Comme prévu, LD fournit un meilleur majorant que LR. En moyenne, sur les instances WEINGi, $i = 1, \ldots, 6$, le gap, entre les valeurs de la relaxation et de l'optimum, associé à LD (resp. LR) est de 0.02 (resp. 0.78). Cependant on observe que le temps de résolution moyen de CG_LR (0.07 s) est très faible comparé à celui de CG_LD (10.54 s). Cela s'explique par l'effort de calcul de chaque itération de CG_LD qui est plus important que celui de CG_LR et à la convergence lente de CG_LD comparé à CG_LR.

Nous considérons également 6 instances du problème d'affectation généralisée de la OR-Library. Toutes les instances gap$i$, $i = 1, \ldots, 6$ ont la même taille (5 agents et 15 tâches). Les problèmes maître et auxiliaire sont résolus avec CPLEX11.2. Le tableau 3.2 montre la comparaison entre les performances de la LR et de la LD quand nous utilisons la seconde décomposition classique de Dantzig-Wolfe de la LR, en relâchant les contraintes de capacité.

Comme précédemment, l'optimalité pour CG_LR et CG_LD est atteinte pour toutes les instances. LD donne de meilleurs majorants que LR. En moyenne, sur

TABLE 3.2 – *Relaxation et décomposition Lagrangiennes pour le (GAP)*

	gap1						gap2					
	vR	%vE	Iter	tG	tSP	tMP	vR	%vE	Iter	tG	tSP	tMP
CG_LR	343.59	2.3	33	0.27	0.16	0.03	339.38	3.8	26	0.22	0.17	0.00
CG_LD	337.00	0,3	1169	383.13	343.61	29.37	327.00	0,0	894	258.41	234.55	15.78
	gap3						gap4					
	vR	%vE	Iter	tG	tSP	tMP	vR	%vE	Iter	tG	tSP	tMP
CG_LR	349.68	3.2	33	0.22	0.14	0.01	350.40	2.8	31	0.25	0.17	0.00
CG_LD	339.50	0.1	945	273.18	245.89	19.01	341.00	0.0	878	282.25	258.89	15.74
	gap5						gap6					
	vR	%vE	Iter	tG	tSP	tMP	vR	%vE	Iter	tG	tSP	tMP
CG_LR	335.76	3.0	35	0.28	0.17	0.05	351.82	2.0	30	0.22	0.12	0.08
CG_LD	327.25	0.4	595	163.86	149.73	9.05	345.00	0.0	1115	334.65	301.99	23.93

vR : la valeur de la relaxation.
%vE : le gap entre les valeurs de la relaxation et de l'optimum.
Iter : nombre d'itérations.
tG : le temps de résolution global (s).
tSP : le temps de résolution des problèmes auxiliaires (s).
tM : le temps de résolution cumulé des problèmes maîtres (s).

les instances gap$i$, $i = 1, \ldots, 6$, le gap, entre les valeurs de la relaxation et de l'optimum, associé à LD (resp. RL) est de 0.13 (resp. 2.85), mais nous observons également que le temps moyen de résolution de CG_LR (0.24 s) est encore très faible comparé à celui de CG_LD (282.58 s).

La première décomposition classique de Dantzig-Wolfe de la LR, en relâchant les contraintes d'affectation (cf section 3.1.1), a aussi été testée sur les mêmes instances, les résultats montrent que les bornes sont plus serrées (mais toujours moins bonnes que celles obtenues par la LD) et que l'algorithme de génération de colonnes met plus d'itérations et de temps pour converger.

Conclusion

Nous avons proposé un nouveau problème maître issu de la décomposition de Dantzig-Wolfe pour la programmation en nombres entiers, qui permet de proposer une preuve alternative de la dominance de la borne de la Décomposition Lagrangienne sur la borne de la Relaxation Lagrangienne. Nous avons illustré ce résultat en l'appliquant à deux problèmes classiques de l'optimisation combinatoire, le sac à dos bidimensionnel en 0-1 et l'affectation généralisée. Les résultats expérimentaux obtenus montre la supériorité de la borne de la Décomposition Lagrangienne, mais le gain qualitatif impose un effort computationnel important. Grâce à cette étude comparative, nous pouvons conclure que la décomposition Lagrangienne résolue par la génération de colonnes peut être utile si l'on souhaite obtenir une borne de bonne qualité, comme par exemple au nœud racine d'un schéma de séparation et d'évaluation.

3.1.2 Diversification

Nous positionnons ce travail dans le cadre de la résolution de la relaxation continue de la formulation de Dantzig et Wolfe du Problème de Tournées de Véhicules avec Fenêtres de Temps (PTVFT) avec la génération de colonnes. Les sous-problèmes engendrés sont de type Problème de Plus Court Chemin avec Fenêtres de Temps et Contraintes de Capacité (PPCCFTCC) et sont résolus généralement avec la programmation dynamique.

Il est bien connu que, dans un schéma de Génération de Colonnes (GC), l'intensification, qui consiste à rajouter au Problème Maître (PM) un ensemble (voire toutes) de colonnes améliorantes lors de chaque itération, permet de diminuer le nombre d'itérations. Malheureusement, cette intensification rajoute des colonnes qui n'appartiendront jamais à la base optimale et agrandissent inutilement le problème maître alors que la base optimale ne contiendra qu'une très faible propor-

tion de l'ensemble des colonnes générées. À cet effet, nous nous sommes proposés d'étudier les caractéristiques des colonnes générées afin d'établir un critère de sélection de "bonnes" colonnes pour garder le bénéfice de l'intensification, à savoir réduire le nombre d'itérations de la génération de colonnes tout en gardant une taille raisonnable du problème maître. Nous validerons cette étude sur le Problème de Tournées de Véhicules avec Fenêtres de Temps (PTVFT).

Intensification des solutions dans un schéma de génération de colonnes

Afin d'accélérer la résolution par la GC, il est avantageux d'insérer lors de chaque itération plusieurs colonnes dans le Problème Maître (PM). Nous pouvons à cet effet exploiter des méthodes d'optimisation permettant de calculer plusieurs solutions réalisables, telles que la programmation dynamique (Bellman (1957)) et les heuristiques.

Nous exposons ci-dessous quelques résultats expérimentaux sur des instances du PTVFT, où les sous-problèmes sont résolus par la programmation dynamique avec correction d'étiquettes (section 2.4.2). Deux types d'instances sont considérées, des instances de Solomon de 25, 50, 100 et 200 clients et des instances générées aléatoirement de 100, 120, 140 et 160 clients. Les résultats présentés pour chaque classe d'instances générées aléatoirement (G_100, G_120, G_140 et G_160) représentent une moyenne sur 10 instances.

Nous appelons *colonne sous-optimale de base*, une solution sous-optimale du sous-problème et qui appartient à la base optimale. Le tableau 3.3 montre la contribution des colonnes sous-optimales dans la base optimale. Les instances test sont résolues avec la Génération de Colonnes Intensifiée (GCI), qui consiste à rajouter lors de chaque itération toutes les colonnes de coût réduit négatif.

Instance	C101_25	C101_50	C101	C1_2_1	G_100	G_120	G_140	G_160
NbCols	246	1 003	2 724	11 461	16 960	19 013	25 259	23 864
%CSBO	60	87	81	87	97	98	96	98
%CBCG	2,0	0,7	0,5	0,1	0,6	0,2	0,2	0,5

NbCols : nombre total de colonnes insérées dans le problème maître.
%CSBO : pourcentage de colonnes sous-optimales dans la base finale.
%CBCG : pourcentage des colonnes de la base finale dans l'ensemble des colonnes générées.

TABLE 3.3 – *Génération de colonnes sous-optimales*

Ces résultats montrent que le pourcentage de colonnes sous-optimales dans la base optimale est supérieur à 80%, excepté pour la plus petite instance de Solomon C101_25. Une colonne sous-optimale par rapport à un sous-problème est optimale par rapport au sous-problème résolu sur un domaine réduit. Elle représente donc soit un point intérieur, soit un point extrême de l'enveloppe convexe du domaine réalisable du sous-problème. Lorsque cette solution est un point extrême, alors elle est solution du sous-problème par rapport à une solution duale réalisable. Du point de vue dual, une coupe associée à une colonne sous-optimale peut être une facette de la fonction duale.

L'ajout prématuré de colonnes au PM permet d'accélérer le calcul d'une base optimale, cela permet de diminuer le nombre d'itérations. En moyenne, plus de 99% de l'ensemble des colonnes générées n'appartiennent pas à la base finale. La procédure appliquée ci-dessus, qui consiste à insérer dans le PM toutes les colonnes de coût négatif, engendre des problèmes maîtres denses. Parmi toutes les colonnes générées, peu d'entre elles appartiennent à la base optimale. On peut en conclure que parmi toutes les colonnes sous-optimales générées, il existe de

"bonnes" colonnes (qui vont contribuer à la base optimale) et des colonnes inutiles qui ralentissent la résolution.

Nous étudions à cet effet les caractéristiques des colonnes générées afin de proposer un critère de sélection de "bonnes" colonnes. Notre objectif est de diminuer le nombre de colonnes insérées dans le PM sans augmenter considérablement le nombre d'itérations afin de diminuer le temps global de résolution.

Génération de solutions de meilleurs coûts réduits Afin d'éviter une expansion rapide de la taille du PM, nous ne rajoutons généralement que les k meilleures solutions (k solutions de meilleurs coûts réduits) lors de chaque itération, k étant un paramètre à déterminer. Dans la suite, k représente le pourcentage de meilleures solutions à rajouter au PM, le cas précédent (GCI) correspond donc à $k = 100$. Dans la suite, nous appelons cette procédure la Génération de Colonnes_k%_Intensifiée (GC_k%_I).

Nous présentons dans le tableau 3.4 une étude comparative entre les performances de la GCI et de la GC_k%_I pour différentes valeurs de k. Nous rappelons que les résultats reportés pour les instances générées aléatoirement représentent des moyennes sur 10 instances.

Instances de Solomon	*C*101_25				*C*101_50			
	nbI	nbC	tG	tM	nbI	nbC	tG	tM
GC_10%_I	50	147	3,6"	0,06"	179	451	80,8"	0,9"
GC_20%_I	51	153	4,3"	0,06"	127	647	73,1"	0,7"
GC_50%_I	30	201	2,8"	0,05"	61	823	32,7"	0,4"
GCI	24	246	2,1"	0,04"	55	1 003	31,7"	0,4"
Instances de Solomon	*C*101				*C*1_2_1			
	nbI	nbC	tG	tM	nbI	nbC	tG	tM
GC_10%_I	286	1 819	26,1'	6,2"	402	6 549	468,2'	5,0'
GC_20%_I	170	2 008	15,2'	4,1"	286	10 768	438,6'	5,8'
GC_50%_I	105	2 154	9,6'	3,9"	250	11 020	197,1'	5,0'
GCI	63	2 724	5,0'	1,4"	163	11 460	150,9'	4,5'
Instances générées	*G*_100				*G*_120			
	nbI	nbC	tG	tM	nbI	nbC	tG	tM
GC_10%_I	101	3 828	11,4'	2,3'	158	5 141	37,7'	5,9'
GC_20%_I	69	6 263	10,3'	2,2'	99	7 864	32,3'	6,6'
GC_50%_I	36	11 406	9,8'	2,0'	48	13 663	31,2'	6,0'
GCI	27	16 960	12,5'	1,4'	26	19 013	39,1'	0,7'
Instances générées	*G*_140				*G*_160			
	nbI	nbC	tG	tM	nbI	nbC	tG	tM
GC_10%_I	151	3 738	42,0'	10,6'	101	6 715	24,6'	4,7'
GC_20%_I	97	10 169	43,5'	12,4'	72	10 409	18,0'	5,1'
GC_50%_I	49	18 800	40,1'	12,1'	42	17 980	20,0'	5,2'
GCI	25	25 259	35,5'	2,1'	27	23 864	20,4'	0,7'

nbI : nombre total d'itérations.
nbC : nombre total de colonnes générées.
tG : temps global de résolution.
tM : temps cumulé de résolution des problèmes maîtres.

TABLE 3.4 – *Génération de colonnes intensifiée et k% intensifiée*

Ces résultats montrent que le nombre total de colonnes rajoutées au PM augmente lorsque k augmente, en moyenne pour toutes les instances. En moyenne sur les instances de Solomon, le gain obtenu sur le nombre total de colonnes rajoutées au PM avec la GC_10%_I, GC_20%_I et GC_50%_I est supérieur à 52%, 26% et 15% respectivement par rapport à la GCI, et en moyenne sur les instances générées aléatoirement, ce gain dépasse 76%, 65% et 27% par rapport à la GCI respectivement. La base optimale peut donc être caractérisée en insérant moins de colonnes

dans le PM lors de chaque itération. Plus le nombre de colonnes insérées dans le PM est faible, plus le gain en nombre de colonnes générées par rapport à la GCI est grand. On s'attend ainsi à ce que le temps de résolution des PM diminue. Effectivement, le temps de résolution moyen des PM diminue avec la diminution de k pour toutes les instances, alors que le temps cumulé de résolution des problèmes maîtres augmente. Cela est dû à une forte augmentation du nombre d'itérations lorsque k décroît pour toutes les instances. Cela augmente au final le temps global de résolution en moyenne sur toutes les instances.

Nous constatons à travers ces expérimentations que l'ajout de colonnes de meilleurs coûts réduits augmente considérablement le nombre d'itérations. Cela provoque une augmentation du temps global de résolution malgré une diminution significative du nombre total de colonnes générées.

Dans le tableau 3.5, nous étudions la contribution des solutions sous-optimales générées, dans la base finale pour les procédures précédentes. Nous donnons le pourcentage moyen de colonnes sous-optimales dans la base finale pour les instances générées aléatoirement. Pour chaque classe d'instances, ce pourcentage augmente lorsque k augmente. Ainsi les solutions sous-optimales de bon coût réduit ont une plus faible contribution à la base optimale que des colonnes de coût de plus mauvaise qualité. On déduit à travers ces expérimentations que le critère de sélection de nouvelles colonnes à insérer dans le problème maître se basant sur le coût réduit n'améliore pas l'approximation du domaine réalisable du sous-problème et la fonction duale.

	G_{100}	G_{120}	G_{140}	G_{160}
GC_10%_I	88	84	87	90
GC_20%_I	89	92	92	96
GC_50%_I	95	96	95	97
GCI	96	98	96	99

TABLE 3.5 – *Pourcentage des colonnes sous-optimales dans la base optimale*

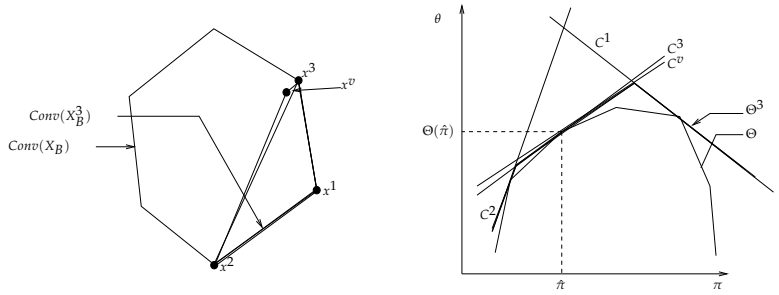
Nous avons constaté que, sur toutes les instances de test, les solutions de meilleurs coûts réduits ont des déviations faibles par rapport à la solution optimale. Nous étudierons dans la section suivante quelques unes de leurs caractéristiques.

Génération de solutions voisines

Définition 3. *Soient $x, x' \in \mathbb{N}^n$, $n \in \mathbb{N}$ et $L = \{i \mid x_i = x'_i, x_i \neq 0, i = 1, ..., n\}$. On dit que x est "voisin" par rapport à x' si et seulement si $|L|$ est "proche" de n.*

La figure 3.1 illustre quelques caractéristiques d'une solution voisine du point de vue primal et dual. Notons $Conv(X_B^3)$ (resp. Θ^3), l'approximation du domaine $Conv(X_B)$ (resp. de la fonction Θ), obtenue par les colonnes x^1, x^2 et x^3 (resp. C^1, C^2 et C^3), délimitée par les traits gras (Figure 3.1-(a) (resp. 3.1-(b))). Remarquons que l'ajout de x^v n'apporte pas d'informations pertinentes en plus de celles obtenues par x^3. Du point de vue dual, la coupe ajoutée C^v est redondante.

L'ajout de colonnes très voisines peut être sans intérêt car elles apportent des informations redondantes au modèle, on obtient ainsi des majorants de mauvaise qualité. Nous avons besoin de générer moins de colonnes mais de bonne qualité, afin d'améliorer significativement l'approximation du domaine réalisable du sous-problème (et de la fonction duale) et d'obtenir de meilleurs majorants dans le but d'accélérer la résolution. À cet effet, au lieu de sélectionner les colonnes à rajouter



(a) Espace primal (b) Espace dual

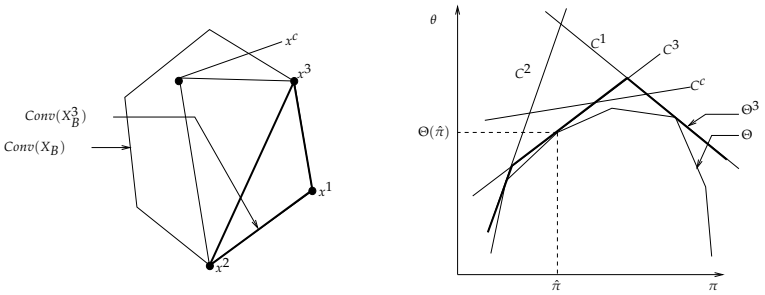
FIGURE 3.1 – *Ajout d'une colonne/coupe voisine au problème maître*

au PM en fonction de leurs coûts réduits, nous nous intéressons à la structure de ces colonnes.

Afin d'éviter de générer des solutions voisines, nous nous sommes proposés de diversifier les solutions insérées dans le PM. Nous étudions dans la suite l'impact sur les performances de la méthode de l'ajout de solutions complémentaires entre elles (au sens de la solution optimale du problème maître) au PM lors de chaque itération de la GC.

Génération de solutions complémentaires Nous présentons dans cette section quelques caractéristiques de solutions complémentaires des points de vue primal et dual.

Définition 4. *Soient $x, x' \in \mathbb{N}^n$, $n \in \mathbb{N}$, le vecteur x' est dit complémentaire par rapport à x si et seulement si $x_i \neq 0 \Rightarrow x'_i = 0$, $\forall i = 1, ..., n$.*



(a) Espace primal (b) Espace dual

FIGURE 3.2 – *Ajout d'une colonne/coupe complémentaire au problème maître*

La figure 3.2-(a) (resp. 3.2-(b)) illustre la même situation que la figure 3.1-(a) (resp. 3.1-(b)), mais au lieu d'ajouter une colonne voisine, nous rajoutons une solution complémentaire par rapport à x^3, notée x^c. La complémentarité (resp. déviation) de la solution x^c (resp. la coupe C^c) par rapport à x^3 (resp. C^3) permet d'améliorer l'approximation de $Conv(X)$ (resp. Θ).

Nous nous intéressons dans ce qui suit à une interprétation théorique de l'ajout de coupes sous-optimales au PM en étudiant les caractéristiques des sous-gradients.

Proposition 4. *Soient C une coupe optimale, g le sous-gradient associé, $C^1, ..., C^p$, p coupes sous-optimales et $g^1, ..., g^p$ les sous-gradients normalisés associés respectivement. La meilleure coupe associée à une solution sous-optimale C^q, $q \in \{1, ..., p\}$ (qui améliore l'approximation de la fonction duale), est associée au*

sous-gradient g^q qui minimise le produit scalaire avec le sous-gradient de la coupe optimale g.

Preuve 11. *On suppose dans ce qui suit que les vecteurs $g^i, \forall i \in \{1,...,p\}$ sont normalisés. Si $(\widehat{g,g^i}) > (\widehat{g,g^j}), \forall j = 1,...,p, j \neq i$, alors, le sous-gradient qui minimise le produit scalaire avec g est g^i.*

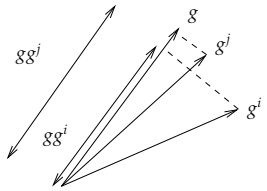


FIGURE 3.3 – *Caractéristiques des sous-gradients*

Comme les vecteurs $g^i, \forall i \in \{1,...,p\}$, sont normalisés, le sous-gradient g^q qui minimise le produit scalaire $g.g^i, \forall i \in \{1,...,p\}$, est celui qui minimise le cosinus de l'angle $(\widehat{g,g^i}), \forall i \in \{1,...,p\}$, et par conséquent celui qui améliore l'approximation de la fonction duale (figure 3.3). □

Dans ce qui suit, nous étudierons ce critère sur le PTVFT qui servira pour la validation expérimentale. Soit $\mathcal{R}$ l'ensemble des chemins allant de la source au puits et satisfaisant les contraintes temporelles, I_n est un vecteur identité de dimension n et $\delta_r = (\delta_{1r},...,\delta_{nr})$, où $\delta_{ir} = 1$ si le chemin r passe par le sommet i et est nul sinon.

Le sous-gradient associé à un chemin $r \in \mathcal{R}$ est donné comme suit :

$$g_r = I_n - \delta_r$$

Proposition 5. *Soit x la solution optimale obtenue par la résolution du sous-problème dans une itération de la GC, x^v (resp. x^c) une solution voisine (resp. complémentaire) par rapport à x, et g, g^v, g^c les sous-gradients associés à x, x^v, x^c respectivement. Si la longueur de la solution x^c est supérieure à la longueur de x^v à laquelle on retranche le nombre de sommets communs entre x et x^v, alors la solution complémentaire domine la solution voisine, i.e. :*

$$g.g^c < g.g^v$$

Preuve 12. *Notons n_t, n_{tv} et n_{tc} les longueurs des solutions x, x^v et x^c respectivement, et par n_v le nombre de colonnes en commun entre x et x^v.*

- *Entre g et g^v, il y a n_v composantes similaires et égales à 0, $n - (n_t + n_{tv} - n_v)$ composantes similaires et égales à 1 et $n_t + n_{tv} - 2n_v$ composantes complémentaires (0 ou 1). Le produit scalaire est donné par :*

  $$g.g^v = n - n_t - n_{tv} + n_v.$$

  Pour des solutions de même longueur, plus n_v (le nombre de composantes égales et non nulles entre x et x^v) est grande, plus le produit scalaire $g.g^v$ est grand.
- *Entre g et g^c, il y a $n - n_t - n_{tc}$ composantes similaires et égales à 1, et $n_t + n_{tc}$ composantes complémentaires. Le produit scalaire est donné par :*

  $$g.g^c = n - n_t - n_{tc}. \tag{3.1}$$

  Remarquons que plus la longueur de la solution complémentaire n_{tc} est grande, plus le produit scalaire $g.g^c$ est petit.

Nous établissons ci-dessous la relation entre les produits scalaires $g.g^c$ et $g.g^v$:

si $n_{tv} - n_v < n_{tc}$ alors $g.g^c < g.g^v$.

Autrement dit, si le nombre de composantes complémentaires entre x^c et x est supérieur à la différence entre la longueur de x^v et le nombre de composantes voisines entre x^v et x, alors la solution x^c est plus intéressante que x^v. □

Nous nous focalisons ci-dessous sur le cas où les solutions x^c et x^v sont de même longueur.

Proposition 6. *Si les solutions x^c et x^v sont de même longueur, alors la solution complémentaire domine la solution voisine, i.e. :*

$$g.g^c < g.g^v$$

Preuve 13. *Lorsque x^v et x^c couvrent le même nombre de sommets ($n_{tv} = n_{tc}$), alors $g.g^v - g.g^c = n_v$.*

Si $g.g^c = 0$, les solutions x et x^c couvrent tous les sommets du graphe $(n_t + n_{tc} = n)$. □

Nous avons étudié dans cette section l'impact de l'ajout de colonnes complémentaires au PM sur la qualité de l'approximation du domaine réalisable du sous-problème (et de la fonction duale). Nous nous intéressons dans la section suivante à l'impact de l'ajout de solutions complémentaires dans le PM sur les performances de la GC à travers des expérimentations sur le PTVFT.

Méthodes de diversification et expérimentations

Les méthodes de diversification consistent à insérer lors de chaque itération dans le PM un ensemble de colonnes complémentaires. L'algorithme 9 présente les principales étapes de la GC avec diversification des solutions.

Algorithme 9: Algorithme générique d'une itération de la GC avec diversification des solutions

1. Résoudre le sous-problème
2. Pool $\leftarrow \emptyset$, ensemble des colonnes insérées dans le PM
3. $X \leftarrow$ toutes les solutions de coût réduit négatif
4. **tant que** $X \neq \emptyset$ **faire**
   - Sélectionner dans X, une solution complémentaire par rapport à toutes les solutions dans Pool, de coût minimal : $\hat{x}$
   - Pool $\leftarrow$ Pool $\cup \hat{x}$

Nous distinguons dans ce qui suit deux manières différentes de calculer ces solutions particulières. Nous présentons pour chacune d'entre elles le principe ainsi que des résultats expérimentaux sur le PTVFT.

Diversification par sélection La Génération de Colonnes avec Diversification par Sélection (GCDS) consiste, lors de chaque itération de la GC, à sélectionner itérativement une solution complémentaire de meilleur coût réduit comparé

Algorithme 10: Algorithme générique d'une itération de la GC avec diversification par sélection

1. Résoudre le sous-problème
2. $X \leftarrow$ toutes les solutions de coût réduit négatif
3. **tant que** $X \neq \emptyset$ **faire**
   - Sélectionner dans X, une solution de coût minimal : $\hat{x}$
   - $S \leftarrow$ l'ensemble des sommets appartenant à $\hat{x}$
   - $X_s \leftarrow$ ensemble des solutions de X qui ont au moins un sommet dans S
   - $X \leftarrow X \setminus X_s$

à toutes les solutions sélectionnées antérieurement. L'algorithme 10 présente les principales étapes de cette procédure.

Nous avons implémenté cette procédure dans la plateforme logicielle COGiTO (Column Generation in Transport Optimization), et nous l'avons testée sur des instances de la littérature et sur des instances générées aléatoirement. Nous souhaitons observer la qualité de l'approximation du domaine réalisable du sous-problème obtenue par la GCDS comparé à la GC_k%_I (k=10, 20, 50, 100) en observant l'évolution de la valeur du PM au cours de la résolution.

Nous présentons dans le tableau 3.6 une étude comparative entre les méthodes citées ci-dessus. Nous observons le nombre d'itérations, le nombre total de colonnes insérées dans le PM, le temps cumulé de résolution des PM et le temps global de résolution.

La GCDS permet de :

- Réduire le nombre de colonnes générées par rapport à la GC_k%_I (k = 10, 20, 50, 100). Nous ne génèrons en moyenne que 30% (resp. 44% et 36%) du nombre de colonnes générées par GCI (resp. 20%_GCI et 50%_GCI) pour les instances de Solomon. De plus nous ne génèrons en moyenne que 3% (resp. 7% et 4%) du nombre de colonnes générées par GCI (resp. GC_20%_I et GC_50%_I) pour les instances générées aléatoirement. On manipule alors des PM plus petits dont la résolution est plus facile.
- Réduire significativement le temps cumulé de résolution des problèmes maîtres pour toutes les instances.
- Diminuer le temps global de résolution par rapport à au moins une méthode de la GC_k%_I (k = 10, 20, 50, 100) pour les instances de Solomon et par rapport à toutes ces méthodes pour les instances générées aléatoirement.

Cependant, nous remarquons à travers ces expérimentations que l'intensification des solutions dans la GCDS est faible (le nombre de colonnes complémentaires ajoutées dans le PM lors de chaque itération est faible). En moyenne sur toutes les instances test, 5 colonnes sont insérées dans le PM lors de chaque itération de la GCDS contre 32 pour la GC_10%_I, 72 pour la GC_20%_I, 212 pour GC_50%_I et 511 pour la GCI, et généralement les longueurs de ces colonnes sont relativement petites. Il serait plus avantageux de générer plus de colonnes complémentaires de longueurs supérieures.

Diversification par résolution Afin d'intensifier la génération de colonnes complémentaires de coût négatif lors de chaque itération de la GC, nous proposons la GC avec Diversification par Résolution (GCDR). Lors de chaque itération de la GC

Instances de Solomon	C101_25				C101_50			
	nbI	nbC	tG	tM	nbI	nbC	tG	tM
GCDS	69	107	5,7”	0,02”	135	315	72,9”	0,4”
GC_10%_I	50	147	3,6”	0,07”	179	451	80,8”	0,9”
GC_20%_I	51	153	4,3”	0,06”	127	647	73,1”	0,7”
GC_50%_I	30	201	2,8”	0,05”	61	823	32,7”	0,4”
GCI	24	246	2,1”	0,04”	55	1 003	31,7”	0,4”
Instances de Solomon	C101				C1_2_1			
	nbI	nbC	tG	tM	nbI	nbC	tG	tM
GCDS	206	768	19,4’	1,7”	320	2 263	430,2’	1,4’
GC_10%_I	286	1 819	26,1’	6,2”	402	6 549	468,2’	5,0’
GC_20%_I	170	2 008	15,2’	4,1”	286	10 768	438,6’	5,8’
GC_50%_I	105	2 154	9,6’	3,9”	250	11 020	197,1’	5,0’
GCI	63	2 724	5,0’	1,4”	163	11 460	150,9’	4,5’
Instances générées	G_100				G_120			
	nbI	nbC	tG	tM	nbI	nbC	tG	tM
GCDS	84	475	5,7’	0,9”	127	666	21,3’	3,3”
GC_10%_I	101	3 828	11,4’	2,3’	158	5 141	37,7’	5,9’
GC_20%_I	69	6 263	10,3’	2,2’	99	7 864	32,3’	6,6’
GC_50%_I	36	11 406	9,8’	2,0’	48	13 663	31,2’	6,0’
GCI	27	16 960	12,5’	1,4’	26	19 013	39,1’	0,7’
Instances générées	G_140				G_160			
	nbI	nbC	tG	tM	nbI	nbC	tG	tM
GCDS	152	841	20,3’	5,0”	108	647	9,3’	2,3”
GC_10%_I	151	3 738	42,0’	10,6’	101	6 715	24,6’	4,7’
GC_20%_I	97	10 169	43,5’	12,4’	72	10 409	18,0’	5,1’
GC_50%_I	49	18 800	40,1’	12,1’	42	17 980	20,0’	5,2’
GCI	25	25 259	35,5’	2,1’	27	23 864	20,4’	0,7’

nbI : nombre total d’itérations.
nbC : nombre total de colonnes insérées dans le PM.
tG : temps global de résolution.
tM : temps cumulé de résolution des problèmes maîtres.

TABLE 3.6 – *Sélection de colonnes complémentaires*

et de manière gloutonne, le sous-problème est résolu avec l’algorithme de programmation dynamique, puis, tous les sommets appartenant à la solution optimale sont supprimés. Cette procédure est réitérée jusqu’à ce qu’il n’existe plus de chemins de la source au puits. Alors que la diversification par sélection consiste à choisir un sous-ensemble de colonnes complémentaires parmi l’ensemble des colonnes calculées, la diversification par résolution calcule des solutions complémentaires en résolvant plusieurs fois le sous-problème, cela permet de calculer des colonnes de meilleurs coûts par rapport à la diversification par sélection. Les étapes principales de cette approche sont synthétisées dans l’algorithme 11.

Nous avons implémenté cette procédure dans la plateforme COGiTO, et nous l’avons testée sur les mêmes instances que précédemment.

Le tableau 3.7 montre une étude comparative sur les performances de la GCI, GCDS et la GCDR. Ces résultats montrent que l’intensification de solutions complémentaires avec de meilleurs coûts réduits permet de diminuer le nombre de colonnes générées. On génère avec la GCDR en moyenne 15% (resp. 91%) du nombre de colonnes générées avec la GCI (res. GCDS). Cela réduit aussi le nombre d’itérations par rapport à la GCDS et le temps de résolution des problèmes maîtres par rapport à la GCDS et la GCI. Le calcul de ces colonnes complémentaires consomme plus de temps, cela est dû aux résolutions multiples de sous-problèmes réduits. La procédure dominante entre la GCDR et GCDS est celle qui réalise le meilleur compromis entre le gain en temps de résolution obtenu sur la résolution

Algorithme 11: Algorithme générique d'une itération de la GC avec diversification par résolution

Entrées : $N^0 \leftarrow \mathcal{N}$, $A^0 \leftarrow A$, $G^0 \leftarrow (N^0, A^0)$ et $k \leftarrow 0$,
continuer $\leftarrow$ vrai
répéter

1. Résoudre le sous-problème SP^k sur le réseau G^k.
2. $x^k \leftarrow$ la solution optimale.
3. $S^k \leftarrow$ l'ensemble des sommets appartenant à x^k.
4. **si** *(x^k est de coût positif) ou ($S^k = \emptyset$)* **alors** continuer $\leftarrow$ faux.
5. **sinon**
   (a) $N^{k+1} \leftarrow N^k \setminus S^k$.
   (b) $G^{k+1} \leftarrow (N^{k+1}, A^{k+1})$ le sous-graphe partiel de G^k.
   (c) $k \leftarrow k+1$.

jusqu'à *(continuer = faux)*;

Instances de Solomon	C101_25				C101_50			
	nbI	nbC	tG	tM	nbI	nbC	tG	tM
GCDR	57	106	5,3"	0,01"	113	240	71,9"	0,3"
GCDS	69	107	5,7"	0,01"	135	315	72,9"	0,4"
GCI	24	246	2,1"	0,04"	55	1 003	31,7"	0,4"
Instances de Solomon	C101				C1_2_1			
	nbI	nbC	tG	tM	nbI	nbC	tG	tM
GCDR	173	806	18,1'	1,9"	251	1 730	308,4'	0,5'
GCDS	206	768	19,4'	1,7"	320	2 263	430,2'	1,4'
GCI	63	2 724	5,0'	1,4"	163	11 460	150,9'	4,5'
Instances générées	G_100				G_120			
	nbI	nbC	tG	tM	nbI	nbC	tG	tM
GCDR	60	404	6,5'	0,8"	78	537	27,4'	2,6"
GCDS	84	475	5,7'	0,9"	127	666	21,3'	3,3"
GCI	27	16 960	12,5'	84,0"	26	19 013	39,1'	42,0"
Instances générées	G_140				G_160			
	nbI	nbC	tG	tM	nbI	nbC	tG	tM
GCDR	107	731	26,3'	4,5"	84	632	12,1'	2,2"
GCDS	152	841	20,3'	5,0"	108	647	9,3'	2,3"
GCI	25	25 259	35,5'	126,0"'	27	23 864	20,4'	42,0"

nbI : nombre total d'itérations.
nbC : nombre total de colonnes générées.
tG : temps global de résolution.
tM : temps cumulé de résolution des problèmes maîtres.

TABLE 3.7 – *Génération de solutions complémentaires*

des problèmes maîtres et le surcoût de résolution des sous-problèmes. En moyenne sur nos instances test, la procédure dominante par rapport au temps de résolution global est la GCDS.

La diversification permet de diminuer le nombre total de colonnes générées sans accroître significativement le nombre d'itérations et permet de réduire le temps de résolution global pour les grandes instances pour lesquelles la taille des problèmes maîtres est grande, c'est notamment le cas dans nos expérimentations lorsque $n \geq 100$, où n est le nombre de sommets (Tableau 3.7).

Diversification lors des premières itérations

Lorsque le domaine réalisable du Sous-Problème (SP) est bien approché, c'est le cas lors des dernières itérations de la GC, une solution complémentaire du sous-problème courant peut être voisine par rapport à une solution calculée précédemment. Ainsi les procédures de diversification présentées dans la section 3.1.2 peuvent être plus efficaces lors des premières itérations afin de caractériser rapidement une bonne approximation du domaine réalisable du sous-problème.

Nous présentons dans le tableau 3.8 les résultats expérimentaux obtenus par les méthodes GCI, GCDS et GCDR d'un côté et deux nouvelles variantes de l'autre :

- la Génération de Colonnes avec Diversification par Résolution Partielle (GCDRP),
- la Génération de Colonnes avec Diversification par Sélection Partielle (GCDSP) où la diversification est appliquée lorsque l'écart relatif entre deux valeurs successives du PM est suffisamment grand, i.e., $\frac{v(PM^k)-v(PM^{k-1})}{v(PM^k)} \geq \varepsilon$, où $v(PM^k)$ est la valeur du PM à l'itération k de la GC.

Nous fixons dans nos expérimentations ε à 0.001.

Instances de Solomon	*C*101_25				*C*101_50			
	nbI	nbC	tG	nbId	nbI	nbC	tG	nbId
GCDR	57	106	5,3"	57	113	240	71,9"	113
GCDRP	35	251	3,0"	3	83	956	51,8"	4
GCDS	69	107	5,7"	69	135	315	72,9"	135
GCDSP	36	283	2,0"	6	83	830	26,6"	5
GCI	24	246	2,1"	0	55	1 003	31,7"	0
Instances de Solomon	*C*101				*C*1_2_1			
	nbI	nbC	tG	nbId	nbI	nbC	tG	nbId
GCDR	173	806	18,1'	173	251	1 730	308,4'	251
GCDRP	75	2 646	2,1'	2	114	10 141	102,2'	5
GCDS	206	768	19,4'	206	320	2 263	430,2'	320
GCDSP	128	3 159	3,4'	4	207	10 622	121,7'	12
GCI	63	2 724	5,0'	0	163	11 460	150,9'	0
Instances générées	*G*_100				*G*_120			
	nbI	nbC	tG	nbId	nbI	nbC	tG	nbId
GCDR	60	404	6,5'	60	78	537	27,4'	78
GCDRP	40	2 589	5,2'	22	50	2 579	17,5'	28
GCDS	84	475	5,7'	84	127	666	21,3'	127
GCDSP	63	1 537	5,6'	51	89	3 140	18,7'	73
GCI	27	16 960	12,5'	0	26	19 013	39,1'	0
Instances générées	*G*_140				*G*_160			
	nbI	nbC	tG	tM	nbI	nbC	tG	tM
GCDR	107	731	26,3'	107	84	632	12,1'	84
GCDRP	55	5 370	15,4'	37	48	1 298	6,8'	31
GCDS	152	841	20,3'	152	108	647	9,3'	108
GCDSP	92	4 009	17,6'	76	76	2 196	8,3'	53
GCI	25	25 259	35,5'	0	27	23 864	20,4'	0

nbI : nombre total d'itérations.
nbC : nombre total de colonnes générées.
tG : temps global de résolution.
nbId : nombre d'itérations avec diversification.

TABLE 3.8 – *Diversification lors des premières itérations*

L'application de la diversification par résolution lors des premières itérations (GCDRP) augmente le nombre total de colonnes générées, mais permet de diminuer fortement le nombre d'itérations. La GCDRP permet de diminuer significati-

vement le temps de résolution global. Les mêmes observations sont constatées pour la GCDS. L'application de la diversification lors des premières itérations permet de diminuer le nombre d'itérations ainsi que les temps de résolution. Remarquons que pour la plus grande instance de Solomon, avec uniquement 4 itérations avec diversification par résolution, nous avons diminué le temps de résolution de la GCDR de 66%, et avec 12 itérations avec diversification par sélection, nous avons réduit le temps de résolution de la GCDS de 71%.

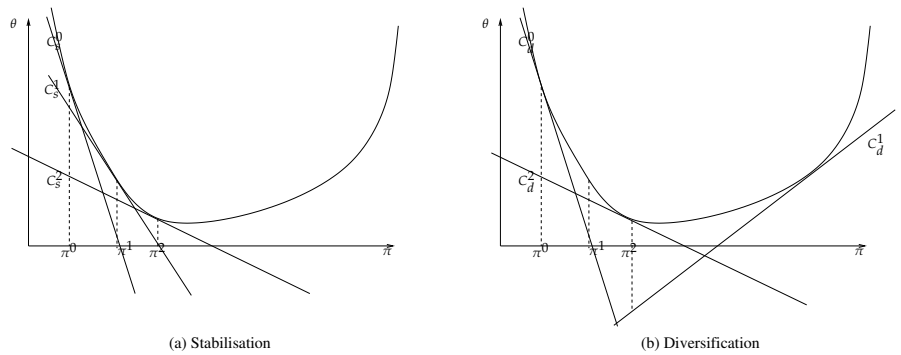
L'application de la diversification, lorsque l'écart relatif entre deux valeurs successives du PM est relativement grand, s'est avérée très efficace pour diminuer le temps de résolution de la GC. Pour nos expérimentations, le temps de résolution de la GCDSP (resp. GCDRP) est meilleur que celui de la GCDS (resp. GCDR) pour toutes les instances.

Les temps de résolution obtenus avec la GCDSP et la GCDRP sont inférieurs à ceux de la GCI pour toutes les instances test. Alors que la GCDS est en moyenne meilleure en temps de calcul que la GCDR, la GCDRP est meilleure en moyenne que la GCDSP. L'impact de la diversification lors des premières itérations est plus visible dans la GCDRP.

Relation entre la diversification et la stabilisation dans la génération de colonnes

Les méthodes de stabilisation de la GC visent à calculer de "bonnes" solutions duales lors de chaque itération, afin d'éviter des itérations inutiles, cela a donc pour effet de diminuer le nombre d'itérations. Le critère de choix de ces bonnes solutions duales se base sur le calcul d'un voisinage de la meilleure solution duale courante. Cela permet d'éviter des déplacements de grande amplitude des solutions duales. Cette méthode vise alors à générer des coupes relativement proches, suivant le voisinage fixé. La figure 3.4-(a) illustre l'approximation de la fonction duale à l'itération 3 de la génération de colonnes, en calculant lors de chaque itération $i \in \{0,1,2\}$, une coupe C_s^i.

La diversification quant à elle consiste à générer des coupes différentes associées à des colonnes complémentaires. Ces coupes ont la propriété d'être plus profondes et permettent de caractériser efficacement une bonne approximation de la fonction duale. De même que la figure 3.4-(a), nous présentons sur la figure 3.4-(b) l'approximation de la fonction duale obtenue avec la diversification, en calculant lors de chaque itération $i \in \{0,1,2\}$, une coupe complémentaire C_d^i.



(a) Stabilisation (b) Diversification

FIGURE 3.4 – *Diversification et stabilisation dans la génération de colonnes*

Alors que la stabilisation vise à construire une bonne approximation locale de

la fonction duale autour de la meilleure solution duale trouvée, la diversification vise à construire une bonne approximation globale de la fonction duale.

Conclusion

Nous avons étudié dans cette section les caractéristiques des colonnes générées lors de chaque itération de la GC. Nous avons montré que la recherche de colonnes de meilleurs coûts réduits n'est pas un critère améliorant pour la GC car ces colonnes n'apportent pas d'informations pertinentes au modèle. Nous avons ensuite établi un critère de choix de colonnes permettant d'améliorer l'approximation du domaine réalisable du sous-problème. Nous avons proposé d'insérer des colonnes complémentaires lors de chaque itération. En effet, il a été prouvé, sur le plan théorique et expérimental sur le problème de tournées de véhicules avec fenêtres de temps, que la génération de colonnes complémentaires améliore significativement l'approximation du domaine réalisable du sous-problème.

Nous avons proposé dans ce chapitre deux variantes de diversification : par résolution et par sélection. Ces techniques réduisent significativement le nombre total de colonnes générées, la taille des problèmes maîtres et le temps de résolution global. La GCDR a l'avantage de générer plus de colonnes complémentaires de bons coûts réduits par rapport à la GCDS. Enfin, nous avons montré que la diversification est plus efficace lors des premières itérations pour construire rapidement une bonne approximation du domaine réalisable du sous-problème.

Nous étudierons dans la section suivante, un autre aspect de l'amélioration de la génération de colonnes, qui consiste à étudier les possibilités de résolution efficace des sous-problèmes. En effet, les sous-problèmes traités, de type plus court chemin avec fenêtres de temps, sont NP-difficiles et leurs résolutions consomment une grande partie du temps d'exécution global. Nous proposerons à cet effet des méthodes efficaces de résolution basées sur le principe de réoptimisation.

3.1.3 Réoptimisation dans la génération de colonnes

Nous nous intéressons dans cette section à l'accélération de la Génération de Colonnes (GC) en améliorant la résolution des sous-problèmes, qui consomment la plus grande partie du temps d'exécution dans plusieurs applications, notamment pour la résolution du Problème de Tournées de Véhicules avec Fenêtres de Temps (PTVFT).

Nous nous focalisons pour cela sur l'apport de techniques de réoptimisation. Ces techniques ont été utilisées efficacement dans plusieurs méthodes itératives pour la résolution d'une séquence d'instances d'un même problème (Desrochers et Soumis (1988), Dionne (1978), Murchland (1967), Pape (1974), Radionov (1968), Thiongane et al. (2005; 2006)). Cette approche consiste à utiliser certaines informations sur la résolution d'une instance à une itération k, afin de résoudre efficacement l'instance de l'itération $k+1$. Nous étudions dans cette section l'apport de ces méthodes dans un schéma de GC. Nous distinguons deux cas de figure où la réoptimisation peut être utilisée. Nous présentons d'abord une nouvelle technique de réoptimisation dans la GC qui vise à diminuer le temps de résolution des sous-problèmes, ensuite en s'inspirant du travail présenté dans Desrochers et Soumis (1988), nous étudierons l'impact de la réoptimisation dans une méthode de diversification, enfin, nous exposons une étude sur l'hybridation de techniques de diversification et de réoptimisation. Ces études sont validées expérimentalement sur le PTVFT.

Résolution des sous-problèmes

Le sous-problème associé au PTVFT résolu avec la GC est un Problème de Plus Court Chemin avec Fenêtres de Temps et Contraintes de Capacité (PPCCFTCC). Ce problème est NP-difficile et sa résolution consomme une grande part du temps de résolution global de la GC. Dans cette section, les sous-problèmes sont résolus avec la programmation dynamique avec correction d'étiquettes, la dominance est appliquée sur les critères coût et temps. Le tableau 3.9 présente une étude expérimentale sur la proportion du temps cumulé de résolution des sous-problèmes par rapport au temps cumulé de résolution des sous-problèmes et des problèmes maîtres pour les instances test. La méthode de résolution considérée est la Génération de Colonnes Intensifiée (GCI).

Instances de Solomon	*C*101_25		*C*101_50		*C*101		*C*1_2_1	
	tCPMSP	tCSP	tCPMSP	tCSP	tCPMSP	tCSP	tCPMSP	tCSP
GCI	1,0"	96%	16,9"	97%	5,1'	99%	125,4'	96%
Instances générées	*G*_100		*G*_120		*G*_140		*G*_160	
	tCPMSP	PtCSP	tCPMSP	PtCSP	tCPMSP	PtCSP	tCPMSP	PtCSP
GCI	7,7'	81%	16,9'	95%	15,8'	86%	7,4'	90%

tCPMSP : temps cumulé de résolution des problèmes maîtres et des sous-problèmes.
PCSP : pourcentage du temps cumulé de résolution des sous-problèmes par rapport au tCPMSP.

TABLE 3.9 – *Résolution des sous-problèmes*

Le temps cumulé de résolution des sous-problèmes représente en moyenne 97% du temps de résolution cumulé des sous-problèmes et des problèmes maîtres pour les instances de Solomon et 88% pour les instances générées aléatoirement. Il serait donc intéressant d'étudier les possibilités de résolutions plus efficaces des sous-problèmes, afin de réduire le temps d'exécution global. Nous étudions dans ce qui suit une possibilité de résolution efficace des sous-problèmes, exploitant le principe de réoptimisation.

Principe de réoptimisation dans les méthodes itératives

Nous présentons dans cette section le principe de réoptimisation pour la résolution d'une séquence finie d'instances non définies, ayant la même structure, mais des données différentes. Nous considérons le problème suivant :

$$(P)\ \max cx, x \in X \subset \mathbb{N}^n.$$

Soient $\sigma(P)$ l'ensemble des solutions optimales de (P) et $x^* \in \sigma(P)$. L'ensemble de toutes les variations des données $\mathcal{R}$ qui laissent la solution x^* optimale est appelé région de stabilité et est défini comme suit :

$$\mathcal{R} = \{(c', X') | c' \in \mathbb{R}^n, X' \subset \mathbb{N}^n, x^* \in \sigma(P')\},$$

avec

$$(P')\ \max c'x, x \in X' \subset \mathbb{N}^n.$$

Lorsque $(c', X') \notin \mathcal{R}$, il est plus intéressant de résoudre (P') en exploitant la résolution de (P) ; nous utilisons dans ce cas la réoptimisation. Cette approche consiste à étudier les possibilités de résolution efficace de (P') en utilisant certaines informations sur la résolution de (P). Nous présentons dans l'algorithme 12 les étapes principales pour la résolution d'une séquence de K instances d'un problème (P) en utilisant la réoptimisation.

Nous exploitons dans ce qui suit ce principe dans deux alternatives : dans un schéma de GC et au sein d'une méthode de diversification.

Algorithme 12: Prototype de résolution avec réoptimisation

Entrées : $(P^1), ..., (P^K)$ différentes instances du problème (P)
$I^0 \leftarrow \emptyset$, informations à sauvegarder
pour $k = 1$ *à* K **faire**
1. Résoudre (P^k) en utilisant les informations dans I^{k-1}
2. $I^k \leftarrow$ des informations sur la résolution de (P^k)

Réoptimisation dans un schéma de génération de colonnes

Dans un schéma de résolution par la GC, nous résolvons une suite de sous-problèmes qui ont le même domaine réalisable et qui diffèrent uniquement par certains coefficients de la fonction objectif (évolution du coût réduit de chaque arc du graphe). La réoptimisation dans ce cas, peut donc être exploitée. Nous proposons dans cette section une nouvelle méthode de réoptimisation et étudions ses performances pour la résolution du PTVFT avec la GC, où les sous-problèmes sont résolus par la programmation dynamique à correction d'étiquettes.

La principale question à se poser lorsque l'on souhaite appliquer la réoptimisation dans un processus itératif est : quelles sont les informations pertinentes à retenir sur la résolution du sous-problème à une itération k, pour résoudre efficacement le sous-problème à l'itération $k+1$? Nous décrivons ci-dessous quelles sont les informations disponibles sur la résolution du sous-problème à une itération k de la GC, afin de sélectionner des informations pertinentes susceptibles de faciliter la résolution du sous-problème à l'itération $k+1$.

À l'itération k de la génération de colonnes

On note par $\mathcal{E}_j^k$ l'ensemble des étiquettes efficaces calculées au sommet $j \in N$ à l'itération k de la GC, où N est l'ensemble des clients (de manière équivalente, l'ensemble des sommets internes du graphe associé au PTVFT). Notons par $E_j^{l(k)}$, la l-ème étiquette du sommet j à l'itération k. Toutes les étiquettes appartenant à l'ensemble $\mathcal{E}_j^k$ (figure 3.5-(a)) construisent une frontière Pareto optimale $(\mathcal{P}^k)$, les étiquettes appartenant à l'espace hachuré sont donc dominées.

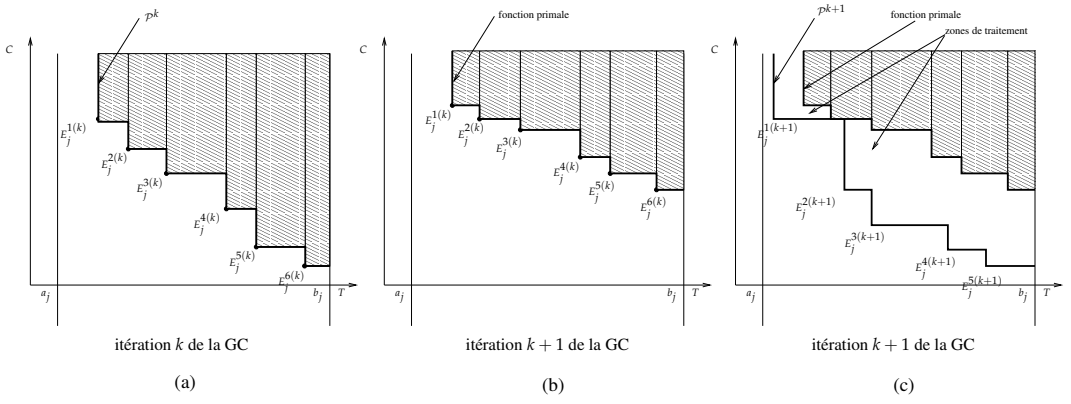


FIGURE 3.5 – *Principe de réoptimisation entre deux itérations successives de la génération de colonnes*

À l'itération k, on dispose donc d'un espace ne contenant que des étiquettes

dominées. Nous allons dans ce qui suit montrer que l'on peut translater cet espace d'étiquettes dominées à l'itération $k+1$.

À l'itération $k+1$ de la génération de colonnes

À l'itération $k+1$, un nouveau problème maître est résolu, on obtient alors de nouvelles valeurs des variables duales. Les coûts des étiquettes efficaces de l'itération k changent (figure 3.5-(b)) en fonction des nouvelles valeurs des variables duales, alors que leurs durées restent inchangées. Les étiquettes efficaces de l'itération k sont donc réalisables à l'itération $k+1$, et constituent une fonction primale représentant une borne supérieure sur le coût des nouvelles étiquettes efficaces en fonction du temps (figure 3.5-(b)).

Les étiquettes potentiellement efficaces à l'itération $k+1$ appartiennent à l'espace majoré par la fonction primale appelée zone de traitement (figure 3.5-(c)).

La procédure proposée consiste donc à sauvegarder les étiquettes efficaces de chaque sommet pour les réutiliser à l'itération suivante. L'algorithme 13 présente les principales étapes de cette procédure.

Algorithme 13: Algorithme de la réoptimisation dans la GC

Entrées : continuer $\leftarrow$ vrai

1. Résoudre le sous-problème avec la programmation dynamique.
2. Sauvegarder les étiquettes efficaces de chaque sommet.
3. **répéter**
   - (a) Pour chaque sommet dans l'ordre topologique :
     - i. Mettre à jour les coûts des étiquettes efficaces sauvegardées.
     - ii. Prolonger les étiquettes efficaces de tous les sommets prédécesseurs dans la zone de traitement.
     - iii. Appliquer la dominance.
     - iv. Sauvegarder les nouvelles étiquettes efficaces.
   - (b) $\mathcal{X} \leftarrow$ les solutions optimales de coût négatif.
   - (c) **si** *($\mathcal{X} = \emptyset$)* **alors** continuer $\leftarrow$ faux

jusqu'à *(continuer = faux)*;

Remarque 6. *Cette technique de réoptimisation peut être utilisée dans un algorithme de GC pour la résolution de problèmes dont le sous-problème peut être résolu avec la programmation dynamique.*

Expérimentations Nous avons implémenté cette technique dans la plateforme logicielle COGiTO. Le tableau 3.10 présente le temps global de résolution, le temps moyen de résolution des sous-problèmes et le nombre total d'étiquettes traitées, obtenus pour les instances test, résolues avec la GCI et la Génération de Colonnes avec Réoptimisation (GCReopt).

Ces résultats montrent que le nombre d'étiquettes traitées dans la GCReopt est largement inférieur à celui de la GCI. Cela est dû à la construction d'une fonction

Instances de Solomon	C101_25			C101_50		
	tG	tMoySP	nbLb	tG	tMoySP	nbLb
GCI	2,1”	0,04”	9 112	31,7”	0,3”	86 073
GCReopt	1,5”	0,02”	2 945	26,7”	0,2”	16 376
Instances de Solomon	C101			C1_2_1		
	tG	tMoySP	nbLb	tG	tMoySP	nbLb
GCI	5,0’	3,6”	384 810	150,9’	46,0”	4 262 215
GCReopt	3,2’	1,1”	81 141	98,2’	29,6”	410 794
Instances générées	G_100			G_120		
	tG	tMoySP	nbLb	tG	tMoySP	nbLb
GCI	12,5’	14,1”	446 080	39,1’	37,5”	5 841 413
GCReopt	9,4’	5,4”	246 578	22,1’	11,9”	545 822
Instances générées	G_140			G_160		
	tG	tMoySP	nbLb	tG	tMoySP	nbLb
GCI	35,5’	33,0”	454 729	20,4’	15,1”	475 701
GCReopt	23,7’	10,6”	352 893	19,2’	7,1”	271 053

tG : temps global de résolution.
tMoySP : temps moyen de résolution des sous-problèmes.
nbLb : nombre d’étiquettes traitées.

TABLE 3.10 – *Réoptimisation dans la génération de colonnes*

primale pour chaque sommet du graphe lors de chaque itération de la GCReopt, qui permet de restreindre significativement le nombre d’étiquettes traitées et donc de diminuer le coût de la procédure de dominance. Le nombre d’étiquettes traitées avec la GCI est diminué en moyenne de plus de 64% avec la GCReopt. Naturellement, cela décroît le temps moyen de résolution des sous-problèmes ainsi que le temps global de résolution. La GCReopt réduit le temps global de résolution de la GCI de plus de 43% en moyenne.

Remarquons que le gain en temps de résolution de la GCReopt par rapport à la GCI (43%) n’est pas aussi important que le gain sur le nombre d’étiquettes (64%), car la sauvegarde des étiquettes efficaces ainsi que la construction de la fonction primale pour chaque sommet du graphe, lors de chaque itération de la GCReopt, consomme un temps non négligeable d’exécution. Cependant, le compromis entre le temps de traitement associé à la réoptimisation et le gain obtenu sur le temps de résolution des sous-problèmes est très satisfaisant.

Réoptimisation lors des dernières itérations et combinaison avec la diversification

Plus les fonctions primale et Pareto optimale sont proches, plus l’espace de recherche de nouvelles étiquettes efficaces est petit et plus la réoptimisation est efficace. Les travaux de Thiongane et al. (2005) et Thiongane et al. (2006) ont montré que la réoptimisation est plus efficace lorsque les instances du problème résolu sont voisines. On observe souvent dans un schéma de résolution avec la GC, que la valeur du problème maître évolue faiblement lors des dernières itérations, ce phénomène est appelé l’amortissement de la génération de colonnes.

Nous présentons dans cette section une étude sur l’application de la réoptimisation lors des dernières itérations de la GC et proposons une combinaison de la réoptimisation et de la diversification dans un schéma de GC.

Réoptimisation lors des dernières itérations Le tableau 3.11 présente une étude comparative entre les performances de la GCI, la GCReopt et de la GCReoptP où la résolution est entamée avec la GCI tant que l’écart relatif entre

deux valeurs successives du problème maître est faible, i.e. $\frac{v(PM^k)-v(PM^{k-1})}{v(PM^k)} \leq \varepsilon$, où $v(PM^k)$ est la valeur du problème maître à l'itération k de la GC, la résolution est ensuite poursuivie avec la GCReopt. Nous fixons dans ces expérimentations $\varepsilon = 0.001$.

Instances de Solomon	C101_25				C101_50			
	tG	tMoySP	nbLb	nbItR	tG	tMoySP	nbLb	nbItR
GCI	2,1"	0,04"	9 112	0/24	31,7"	0,3"	86 073	0/55
GCReopt	1,5"	0,02"	2 945	24/24	26,7"	0,2"	16 376	55/55
GCReoptP	1,2"	0,03"	3 968	18/24	29,0"	0,2"	18 178	48/55
Instances de Solomon	C101				C1_2_1			
	tG	tMoySP	nbLb	nbItR	tG	tMoySP	nbLb	nbItR
GCI	5,0'	3,6"	384 810	0/63	150,9'	46,0"	4 262 215	0/163
GCReopt	3,2'	1,1"	81 141	63/63	98,2'	29,6"	410 794	63/163
GCReoptP	2,1'	0,7"	118 292	52/63	76,8'	31,5"	1 148 708	114/163
Instances générées	G_100				G_120			
	tG	tMoySP	nbLb	nbItR	tG	tMoySP	nbLb	nbItR
GCI	12,5'	14,1"	446 080	0/27	39,1'	37,5"	5 841 413	0/26
GCReopt	9,4'	5,4"	246 578	27/27	22,1'	11,9"	545 822	26/26
GCReoptP	8,9'	4,9"	315 241	8/27	21,2'	10,2"	611 876	9/26
Instances générées	G_140				G_160			
	tG	tMoySP	nbLb	nbItR	tG	tMoySP	nbLb	nbItR
GCI	35,5'	33,0"	454 729	0/25	20,4'	15,1"	475 701	0/27
GCReopt	23,7'	10,6"	352 893	25/25	19,2'	7,1"	271 053	27/27
GCReoptP	21,1'	7,7"	405 266	7/25	17,3'	10,9"	423 800	4/27

tG : temps de résolution global.
tMoySP : temps de résolution moyen d'un sous-problème.
nbLb : nombre total d'étiquettes traitées
nbItR : Nombre d'itérations avec réoptimisation/nombre d'itérations global.

TABLE 3.11 – *Réoptimisation dans un schéma de génération de colonnes*

Nous observons à travers ces résultats une conséquence évidente de la réoptimisation dans la GC sur le nombre d'étiquettes traitées. La GCReoptP réduit le nombre d'étiquettes traitées, qui représente en moyenne 47% de celui de la GCI. Cela entraîne la diminution du temps de résolution des sous-problèmes de l'ordre de 51% par rapport à la GCI. Cependant, on génère plus d'étiquettes avec la GCReoptP qu'avec la GCReopt car la réoptimisation est appliquée uniquement lors des dernières itérations, on perd ainsi en nombre d'étiquettes traitées (de 26% par rapport à la GCReopt), mais on gagne en moyenne sur le temps de calcul (de 13% par rapport à la GCReopt).

On déduit à travers cette étude que, dans notre cas, la réoptimisation est plus productive lors des dernières itérations que lors des premières itérations. Cela se justifie pour deux raisons :

- les sous-problèmes résolus lors des premières itérations engendrent plus d'étiquettes que ceux résolus lors des dernières itérations car les coûts réduits des arcs diminuent en moyenne au cours de la résolution. La construction des fonctions primales est donc plus coûteuse lors des premières itérations.

- Les zones de traitement calculées lors des premières itérations sont plus grandes que celles des dernières itérations, car les solutions duales lors des premières itérations sont plus dispersées alors qu'elles sont voisines lors des dernières itérations.

Nous remarquons que pour les deux plus petites instances de Solomon, le temps d'exécution de la GCReopt est meilleur que celui de la GCReoptP, dans ce cas, le coût de construction des fonctions primales ne compense pas le gain obtenu sur le nombre d'étiquettes traitées lors des premières itérations.

Combinaison de la diversification avec la réoptimisation Nous avons vu précédemment que la diversification des solutions dans un schéma de GC permet d'améliorer les temps de résolution et nous avons montré que la diversification est plus efficace lors des premières itérations pour obtenir assez rapidement une bonne approximation du domaine réalisable du sous-problème. Nous avons étudié dans ce chapitre une autre approche pour l'amélioration des performances de la GC en utilisant le principe de la réoptimisation, cette méthode est plus efficace lors des dernières itérations où les variables duales sont voisines.

Nous étudions dans cette section l'apport de la diversification et de la réoptimisation à la GC. Nous appliquons la diversification par résolution lors des premières itérations et la réoptimisation lors des dernières itérations. Nous validerons cette étude sur les instances test. Nous appliquons la diversification par résolution lorsque l'écart relatif entre deux valeurs successives du PM ($\frac{v(PM^k)-v(PM^{k-1})}{v(PM^k)}$) est supérieur à $\varepsilon = 0.001$, où $v(PM)^k$ est la valeur du PM à l'itération k de la GC. La réoptimisation est appliquée lors des itérations restantes i.e, lorsque $\frac{v(PM^k)-v(PM^{k-1})}{v(PM^k)} \leq \varepsilon$. Cette procédure est appelée Génération de Colonnes avec Combinaison de la Diversification par Résolution et de la Réoptimisation (GCCDRReopt). Le tableau 3.12 présente une étude comparative entre les temps globaux de résolution de la GCI, GCDRP (section 3.1.2), GCReoptP (section 3.1.3) et la GCCDRReopt.

instance	C101_25	C101_50	C101	C1_2_1	G_100	G_120	G_140	G_160
GCI	2,1"	31,7"	5,0'	150,9'	12,5'	39,1'	35,5'	20,4'
GCDRP	1,7"	20,8'	2,1'	102,2'	5,2'	17,5'	15,4'	6,8'
GCReoptP	1,2"	29,0"	2,1'	76,8'	8,9'	21,2'	21,1'	17,3'
GCCDRReopt	1,3"	23,5"	2,0'	71,6'	4,3'	10,6'	15,6'	10,4'

TABLE 3.12 – *Diversification et réoptimisation dans la génération de colonnes*

Le schéma de la GCI est amélioré en utilisant d'une part la diversification lors des premières itérations où le gain sur les temps d'exécution obtenus dépasse 50%, et d'autre part la réoptimisation qui est appliquée lors des dernières itérations, ce qui assure un gain de l'ordre de 36% sur les temps de résolution. Lorsque ces deux techniques sont combinées, le temps de résolution diminue considérablement, le gain moyen en temps obtenu avec la GCCDRReopt par rapport à la GCI est de l'ordre de 51%, voire 60% pour les instances générées aléatoirement. Le gain en temps de résolution de la GCCDRReopt par rapport à la GCReoptP (21%) est supérieur à celui de la GCDRP (4%) car la diversification est moins coûteuse (en temps de traitement) à mettre en œuvre que la réoptimisation.

Hybridation de la réoptimisation avec la diversification et combinaison avec la réoptimisation lors des dernières itérations

Nous nous intéressons dans cette section à l'amélioration des performances de la Génération de Colonnes avec Diversification par Résolution (GCDR) présentée dans la section précédente, en utilisant le principe de réoptimisation dans la phase de diversification. Nous présenterons ensuite une étude sur la combinaison de l'approche proposée avec la réoptimisation lors des dernières itérations d'un schéma de GC.

Intégration de la réoptimisation au sein de la diversification La GCDR consiste à résoudre plusieurs fois le sous-problème lors de chaque itération de la GC. Même si cette méthode a donné des résultats très satisfaisants sur les temps de résolution de la GC, elle subit un surcoût sur la résolution des sous-problèmes. Nous appliquons ici une technique de réoptimisation proposée dans Desrochers et Soumis (1988) pour diminuer le surcoût de résolution des sous-problèmes.

Notons Z^0 le sous-problème de départ et S^0 l'ensemble des sommets du chemin optimal calculé avec l'algorithme de la programmation dynamique. Étant donné que l'objectif est de générer des chemins complémentaires, tous les sommets appartenant à S^0 sont supprimés du graphe, on cherche donc à calculer un chemin optimal dans le sous-graphe partiel, on appellera le nouveau sous-problème Z^1. Pour chaque sommet j, l'ensemble Q_j des étiquettes associées aux chemins efficaces de Z^0 peut être partitionné en deux sous-ensembles :

1. $P_j = \{(C_j^k, T_j^k) |$ le k-ème chemin de s à j X_{sj}^k est réalisable pour $Z^1\}$.
2. $D_j = \{(C_j^k, T_j^k) |$ le k-ème chemin de s à j X_{sj}^k n'est pas réalisable pour $Z^1\}$.

Sachant que les éléments de l'ensemble P_j sont des étiquettes efficaces pour le problème Z^0, elles sont aussi efficaces pour Z^1, on dispose donc d'une partie de la solution du problème Z^1. Cette solution doit être complétée en remplaçant les éléments de D_j.

Remarque 7.

(a) Les étiquettes de l'ensemble $Q_j = P_j \cup D_j$ définissent une fonction duale représentant un minorant des coûts des étiquettes efficaces au sommet j du problème Z^1 en fonction de la date d'arrivée, car les étiquettes de Q_j dominent leurs éventuels remplaçants.

(b) D'un autre côté, les étiquettes de l'ensemble P_j définissent une fonction primale représentant un majorant des coûts des étiquettes efficaces au sommet j du problème Z^1 en fonction de la date d'arrivée, car les étiquettes de P_j sont efficaces pour les deux problèmes Z^0 et Z^1. La figure 3.6 montre ces deux fonctions pour un sommet j donné.

Les nouvelles solutions efficaces du problème Z^1 appartiennent aux zones de traitement. L'objectif est donc de réduire ces zones jusqu'à les éliminer. Pour ce faire, on peut exhiber deux possibilités :

• Augmenter la fonction duale en remplaçant le coût d'une étiquette de D_j par un nouveau minorant. Ce nouveau minorant est calculé en cherchant tous les chemins X_{si} pour lesquels un prolongement par un arc $(i, j) \in A$ produit un chemin X_{sj} dont l'étiquette appartient à la zone de traitement considérée. L'Ensemble

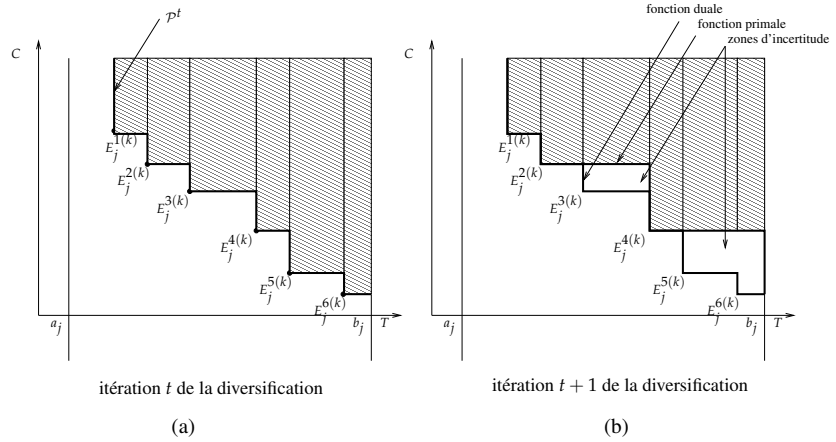


FIGURE 3.6 – *Zones de traitement*

des Étiquettes Efficaces (EEE) est calculé pour obtenir de nouvelles étiquettes qui définissent le nouveau minorant comme suit :

$R_j = EEE\{(C_j, T_j) = (C_i + c_{ij}, max\{a_j, T_i + t_{ij}\}) | \forall (i,j) \in A, (C_i, T_i) \in P_i \cup D_i$ et (C_j, T_j) appartient à la zone de traitement considérée $\}$

• Améliorer la fonction primale en calculant tous les chemins réalisables X_{si} dont le prolongement par un arc $(i,j) \in A$ produit un chemin X_{sj} dont l'étiquette appartient à la zone de traitement considérée. L'ensemble des solutions efficaces qui définit la nouvelle borne primale est calculé comme suit :

$RP_j = EEE\{(C_j, T_j) = (C_i + c_{ij}, max\{a_j, T_i + t_{ij}\}) | \forall (i,j) \in A, (C_i, T_i) \in P_i$ et (C_j, T_j) appartient à la zone de traitement considérée $\}$

Les ensembles P_j et D_j sont alors mis à jour comme suit :

$P_j = P_j \cup (R_j \cap RP_j)$, car les étiquettes appartenant à la fois à R_j et RP_j sont efficaces, et $D_j = (D_j - \{(C_j, T_j)\}) \cap (R_j - (R_j \cap RP_j))$.

Cette procédure s'arrête lorsqu'il ne reste plus de zones d'incertitude, elle présente l'avantage de générer lors de chaque itération des solutions complémentaires (couvrant le maximum de tâches possible) en un temps réduit grâce à la réoptimisation.

Expérimentations

Nous avons implémenté cette approche dans la plateforme logicielle COGiTO et nous l'avons testée sur les instances test. Les résultats obtenus sont exposés dans le tableau 3.13. Nous comparons les temps de résolution globaux des méthodes GCI et GCDRP, avec celui de la Génération de Colonnes avec Diversification par Résolution avec Réoptimisation (GCDRReoptP), où la diversification avec réoptimisation est appliquée lorsque l'écart relatif entre deux valeurs successives du PM est supérieur à $\varepsilon = 0.001$ et la génération de colonnes intensifiée lors des itérations restantes.

Nous remarquons à travers ces résultats que la GCDRReoptP fournit des temps de résolution meilleurs que ceux de la GCDRP pour toutes les instances test générées aléatoirement et les deux plus petites instances de Solomon. Pour ces instances, le compromis entre le surcoût de calcul des fonctions primales et le gain sur le nombre total d'étiquettes traitées est satisfaisant. Pour les deux instances de

instance	C101_25	C101_50	C101	C1_2_1	G_100	G_120	G_140	G_160
GCI	2,1”	31,7”	5,0’	150,9’	12,5’	39,1’	35,5’	20,4’
GCDRP	1,7”	20,8’	2,1’	102,2’	5,2’	17,5’	15,4’	6,8’
GCDRReoptP	0,6”	7,9”	3,9’	212,7’	2,9’	9,0’	8,9’	4,4’

TABLE 3.13 – *Réoptimisation dans la phase de diversification dans la génération de colonnes*

Solomon de plus grande taille, la réoptimisation engendre des surcoûts qui augmentent le temps de résolution global par rapport à la GCDRP. Ces résultats ne dépendent donc pas de l'algorithme de réoptimisation utilisé, mais de l'instance traitée. En moyenne sur toutes les instances test, la GCDRReoptP réduit le temps de résolution de la GCDRP de 12% et celui de la GCI de 54%.

Combinaison entre la diversification par résolution avec réoptimisation et la réoptimisation dans la génération de colonnes

Nous aboutissons ainsi à un schéma intégrant la diversification et la réoptimisation à deux niveaux, au sein de la diversification et entre deux itérations de la GC. Nous présentons une combinaison de la GCDRReoptP et de la GCReoptP, qui est appelée GCDRReopt_R. Cette approche consiste à appliquer la GCDRReoptP lorsque l'écart relatif entre deux valeurs successives du PM est supérieur à un paramètre ε et la GCReoptP lors des itérations restantes. On fixe ici ε à 0.001. Le tableau 3.14 présente les temps globaux de résolution obtenus avec chaque procédure pour les instances tests.

instance	C101_25	C101_50	C101	C1_2_1	G_100	G_120	G_140	G_160
GCI	2,1”	31,7”	5,0’	150,9’	12,5’	39,1’	35,5’	20,4’
GCDRP	1,7”	20,8’	2,1’	102,2’	5,2’	17,5’	15,4’	6,8’
GCReoptP	1,2”	29,0”	2,1’	76,8’	8,9’	21,2’	21,1’	17,3’
GCCDRReopt	1,3”	23,5”	2,3’	71,6’	4,3’	10,6’	15,6’	10,4’
GCDRReoptP	0,6”	7,9”	3,9’	212,7’	2,9’	9,0’	8,9’	4,4’
GCDRReopt_R	0,7”	7,1”	3,6’	202,4’	2,6’	8,4’	8,0’	3,9’

TABLE 3.14 – *Hybridation et combinaison de la diversification et de la réoptimisation*

Dans ce schéma, la diversification est appliquée lors des premières itérations, qui est renforcée en utilisant une technique de réoptimisation. La réoptimisation entre deux itérations est appliquée à la fin du processus de résolution. Ce schéma hybride diminue le temps de résolution de toutes les procédures présentées dans ce chapitre, ainsi que la GCDRP et la GCI. Le gain moyen obtenu sur le temps de résolution de la GCI est de l'ordre de 56%, ce gain est de l'ordre de 18% par rapport à la GCDRP, 21% par rapport à la GCReoptP, 6% par rapport à la GCCDRReopt et 5% par rapport à la GCDRReoptP.

Conclusion

Nous avons présenté dans cette section des techniques de réoptimisation dans un schéma de génération de colonnes. Notre objectif était de diminuer le temps de résolution des sous-problèmes (de type plus court chemin avec fenêtres de temps et contraintes de capacité), qui consomment la plus grande partie du temps d'exécution dans la génération de colonnes. La première méthode de réoptimisation présentée dans ce chapitre consiste à sauvegarder les étiquettes efficaces d'une itéra-

tion k de la génération de colonnes pour les réutiliser à l'itération $k+1$ afin de calculer une borne supérieure sur le coût des nouvelles étiquettes efficaces.

Cette technique est plus efficace lorsque les instances des sous-problèmes sont voisines. Les informations ainsi sauvegardées deviennent alors pertinentes. Nous avons, à cet effet, appliqué cette technique lors des dernières itérations de la génération de colonnes où les variables duales sont proches et donc les sous-problèmes voisins (ce qui est le cas notamment dans les phases d'amortissement).

Nous avons proposé dans la section précédente une autre technique d'amélioration de la génération de colonnes basée sur la diversification, et nous avons montré son interêt lors des premières itérations de la génération de colonnes. Naturellement, nous avons combiné les deux pour aboutir à un schéma plus efficace. Les résultats obtenus à travers cette combinaison sont très satisfaisants, le temps de résolution de la génération de colonnes avec intensification des solutions a été réduit de 51%.

L'intérêt de la diversification et de la réoptimisation ayant été établi, nous avons cherché à intégrer la réoptimisation dans le schéma de diversification. La réoptimisation a permis de diminuer significativement le nombre total d'étiquettes générées. Cette hybridation permet de diminuer le temps de résolution global de 54% en moyenne par rapport à la génération de colonnes avec intensification des solutions.

Nous avons abouti à la fin de cette section à un schéma intégrant la réoptimisation dans la phase de diversification et la réoptimisation entre deux itérations de la GC. Les résultats expérimentaux ont montré que ce schéma réduit le temps de résolution de 56% en moyenne par rapport à la génération de colonnes avec intensification des solutions.

3.1.4 Reformulation et décomposition pour un problème de tournées de véhicules avec dépôts de réapprovisionnement intermédiaires

Le problème de tournées de véhicules avec dépôts de réapprovisionnement intermédiaires (Vehicle Routing Problem with Intermediate Replenishment Facilities (VRPIRF)) est défini sur un graphe où les nœuds représentent un dépôt central, n clients et f dépôts intermédiaires. L'objectif est de déterminer l'ensemble de routes de coût minimal tel que chaque client soit visité une fois pour satisfaire sa demande par l'un des véhicules de la flotte homogène, de capacité fixée et basée au dépôt central. De plus, les véhicules peuvent se réapprovisionner à des dépôts intermédiaires, effectuant ainsi une séquence de routes, appelée rotation, qui part et se termine au dépôt central et telle que sa durée n'excède pas une certaine valeur fixée (voir les figures 3.7 et 3.8 pour un exemple d'instance et de solution).

Ce problème est un cas particulier avec un seul dépôt central du "Multiple Depot VRP with Inter-depot routes (MDVRPI).

Formulations pour le VRPIRF

Soit un graphe orienté $G = (V, A)$ avec V l'ensemble des sommets contenant l'ensemble des n clients V_c numérotés de 0 à $n-1$, le dépôt central Δ numéroté n et les f dépôts intermédiaires V_p numérotés de $n+1$ à $n+f$, et l'ensemble d'arcs $A = V \times V_c \cup V_c \times V$. Chaque arc $ij \in A$ a un coût d_{ij} et un temps de parcours τ_{ij}. Chaque client $i \in V_c$ a une demande q_i et un temps de service τ_i. K est l'ensemble des n_K véhicules qui ont une capacité Q et une durée maximale de parcours T. La recharge en $p \in V_p$ prend un temps τ_p. Nous pouvons alors définir

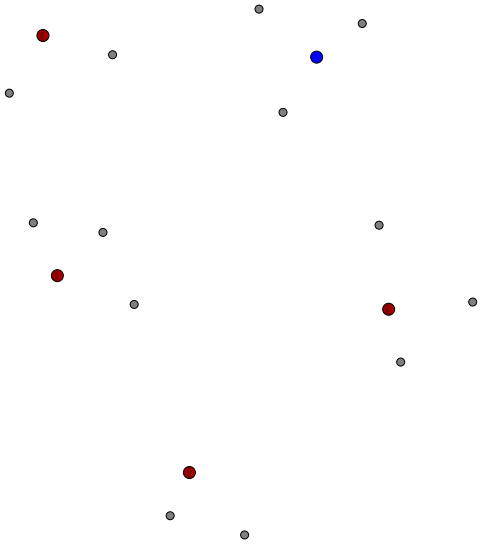


FIGURE 3.7 – *Une instance du VRPIRF : le dépôt central (bleu), les dépôts intermédiaires (rouges) et les clients.*

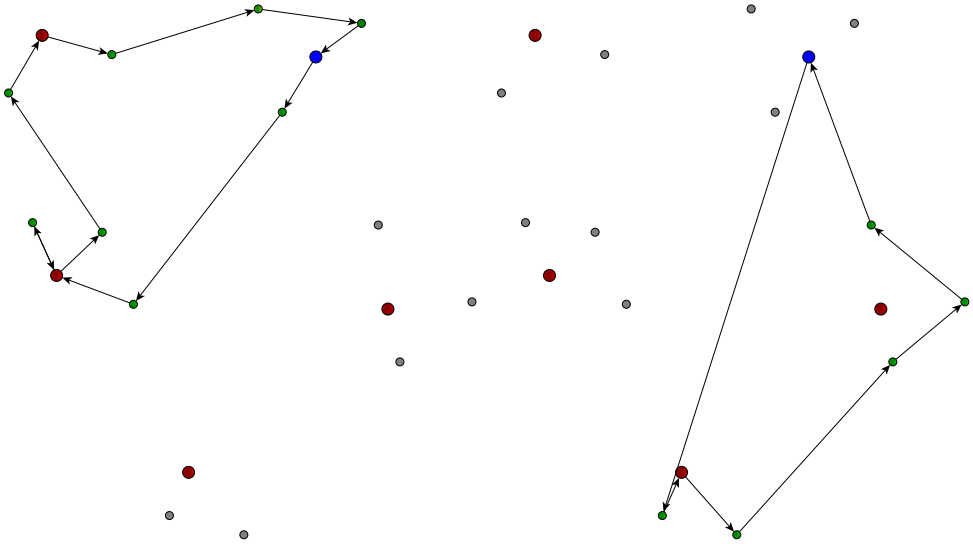


FIGURE 3.8 – *Une solution de l'instance précédente, avec les rotations des deux véhicules.*

les temps étendus de parcours des arcs comme :

$$t_{ij} = \tau_{ij} + \begin{cases} \tau_i & , i \in V_c \\ \tau_p & , i \in V_p \\ 0 & , i = \Delta \end{cases}$$

Nous introduisons également les notations suivantes :
Soient deux ensembles de clients $S_1, S_2 \subseteq V_c$:

- $\overline{S_1}$ est l'ensemble de sommets $V_c \setminus S_1$,
- $S_1 : S_2$ est la coupe $\{ij \in A : i \in S_1, j \in S_2\}$,
- $\delta^+(S_1)$ et $\delta^-(S_1)$ représentent les coupes $(V \setminus S_1) : S_1$ et $S_1 : (V \setminus S_1)$,
- $A(S_1)$ représente $S_1 : S_1$, i.e. les arcs qui ont leurs deux extrémités en S_1.

Nous introduisons enfin les notations suivantes :

- $i : S := \{ij \in A : j \in S\}$, $S : i := \{ji \in A : j \in S\}$, $\delta^-(i) := \delta^-(\{i\})$ et $\delta^+(i) := \delta^+(\{i\})$,

- $\mathcal{S}(V_c)$ est la collection $\{S \subset V_c : 2 \leq |S| \leq |V_c| - 2\}$ de sous-ensembles de clients,
- $\kappa(S)$ est le nombre minimal de routes nécessaire pour servir les clients de $S \subseteq V_c$ et représente la valeur de la solution d'une instance de Bin Packing Problem (BPP) sur S, et $r(S) = \lceil \frac{\sum_{i \in S} q_i}{Q} \rceil$ est un minorant trivial de $\kappa(S)$.

Une formulation à 3 indices Nous utilisons ici des variables binaires de type arc-flot x_{ij}^k telles que $x_{ij}^k = 1$ si l'arc $ij \in A$ est visité par le véhicule $k \in K$. $x^k(A')$ est la somme $\sum_{ij \in A'} x_{ij}^k$ pour tous les $A' \subseteq A$.

$$
\begin{aligned}
(\mathcal{M}_{BC}^1) \quad \min \quad & \sum_{k \in K} \sum_{ij \in A} d_{ij} x_{ij}^k && && (3.2)\\
\text{s.c.} \quad & \sum_{k \in K} (x^k(\delta^-(\Delta)) + \sum_{p \in V_p} x^k(\delta^-(p))) \geq \kappa(V_c) && && (3.3)\\
& \sum_{ij \in A} t_{ij} x_{ij}^k \leq T && \forall k \in K && (3.4)\\
& \sum_{k \in K} x^k(\delta^+(i)) = 1 && \forall i \in V_C && (3.5)\\
& x^k(\delta^+(i)) = x^k(\delta^-(i)) && \forall i \in V_C,\ k \in K && (3.6)\\
& x^k(\Delta : V_c) \leq 1 && \forall k \in K && (3.7)\\
& x^k(\Delta : V_c) = x^k(V_c : \Delta) && \forall k \in K && (3.8)\\
& x^k(p : V_c) = x^k(V_c : p) && \forall k \in K && (3.9)\\
& \sum_{k \in K} x^k(A(S)) \leq |S| - \kappa(S) && \forall S \in \mathcal{S}(V_c) && (3.10)\\
& x_{ij}^k \in \{0,1\} && \forall k \in K,\ ij \in A && (3.11)
\end{aligned}
$$

Les contraintes (3.3) assurent que le nombre d'arcs quittant le dépôt central et les dépôts intermédiaires sera plus grand que la solution du BPP sur l'ensemble des clients. Les contraintes (3.4) imposent un temps limite pour chaque véhicule. Les équations (3.5) et (3.6) imposent que chaque client soit visité une et une seule fois et par un seul véhicule. Les contraintes (3.7) et (3.8) traduisent le fait qu'un véhicule peut ne pas sortir du dépôt central mais s'il est utilisé, il doit alors y retourner. De même, les équations (3.9) assurent qu'à chaque fois qu'un véhicule k entre dans un dépôt intermédiaire p, il doit en sortir. Enfin les contraintes (3.10) sont les inégalités de capacité pour imposer qu'aucune route ne viole la capacité des véhicules.

Ces dernières inégalités sont en nombre exponentiel, nous avons donc développé un branch and cut pour le résoudre. Les instances utilisées sont celles de Tarantilis et al. (2008) et Crevier et al. (2007), de 50 à 175 clients, de 2 à 7 dépôts intermédiaires et de 2 à 8 véhicules. Nous avons obtenu des résultats prometteurs pour les petites instances (50 clients), où nous avons obtenu en temps raisonnable des solutions dont la valeur est très proche de la meilleure solution connue malgré de mauvais minorants. Malheureusement, les résultats se sont beaucoup détériorés pour des tailles d'instances plus importantes, rendant donc cette formulation inutilisable.

Reformulation basée sur des arcs de réapprovisionnement et des temps d'arrivées Nous introduisons dans cette section une nouvelle formulation pour le VRPIRF qui utilise deux séries de variables pour modéliser des arcs de réapprovisionnement et des temps d'arrivées.

Arcs de réapprovisionnement Ce concept a été introduit dans Smith et al. (2012) dans une généralisation du problème de plus court chemin avec contraintes de ressources (Shortest Path Problem with Resource Constraint (SPPRC)), appelée Weight Constrained Shortest Path Problem with Replenishment (WCSPP-R). Pour le VRPIRF, un arc de réapprovisonnement est un arc spécial, noté $i^p j$ qui connecte deux sommets i et j et qui représente un arrêt au dépôt intermédiaire p afin d'effectuer une recharge entre ces deux clients. L'information sur l'arrêt en p est ainsi contenue dans l'arc, ce qui implique que nous n'avons désormais plus besoin des sommets de V_p ; et les rotations peuvent être représentées comme une séquence d'arcs pour laquelle l'on ne rentre et l'on ne sort qu'une seule fois de chaque nœud, sauf lorsqu'il s'agit du dépôt central.

Un arc de réapprovisionnement $i^p j$ a un coût $d_{i^p j} = d_{ip} + d_{pj}$ et un temps de parcours étendu $t_{i^p j} = t_{ip} + t_{pj}$. Étant donnés deux clients $i, j \in V_c$ et un dépôt intermédiaire p, $i^p j$ est dit dominé s'il existe un dépôt intermédiaire q tel que $d_{i^q j} \leq d_{i^p j}$ et $t_{i^q j} \leq t_{i^p j}$, sinon $i^p j$ est non dominé.
L'ensemble $V = \{0, \ldots, n\} = V_c \cup \{\Delta\}$ comprend désormais le dépôt central et les clients. Nous disposons dorénavant d'un ensemble de dépôts intermédiaires $P = \{1, \ldots, f\}$ et d'un ensemble d'arcs $A = A_0 \cup A_P$ qui est composé de l'ensemble $A_0 = V \times V_c \cup V_c \times V$ des arcs de base et de l'ensemble A_P des arcs de réapprovisonnement non dominés :

$$A_P = \{i^p j : \ i, j \in V_c,\ i \neq j,\ p \in P,\ (\nexists q \in P \setminus p)\ d_{i^q j} \leq d_{i^p j} \wedge t_{i^q j} \leq t_{i^p j}\}$$

Nous étendons certaines notations introduites précédemment afin de distinguer les ensembles composés d'arcs de base (indicés par 0) et les ensembles d'arcs de réapprovisonnement (indicés par P). Étant donnés $S_1, S_2 \subseteq V_c$:

- $(S_1 : S_2)_0$ est la coupe d'arcs de base $\{ij \in A_0 : i \in S_1, j \in S_2\}$,
- $(S_1 : S_2)_P$ est la coupe d'arcs de réapprovisonnement $\{ij \in A_P : i \in S_1, j \in S_2\}$,
- $\delta_0^+(S_1) := (S_1 : (V \setminus S_1))_0$ et $\delta_0^-(S_1) := ((V \setminus S_1) : S_1)_0$,
- $\delta_P^+(S_1) := ((V \setminus S_1) : S_1)_P$ et $\delta_P^-(S_1) := (S_1 : (V \setminus S_1))_P$,
- $A_0(S_1) := (S_1 : S_1)_0$ et $A_P(S_1) := (S_1 : S_1)_P$

et pour un nœud générique $i \in V$:

- $(i : S)_0 := (\{i\} : S)_0$, $(S : i)_0 := (S : \{i\})_0$, $\delta_0^+(i) := \delta_0^+(\{i\})$, $\delta_0^-(i) := \delta_0^-(\{i\})$
- $(i : S)_P := (\{i\} : S)_P$, $(S : i)_P := (S : \{i\})_P$, $\delta_P^+(i) := \delta_P^+(\{i\})$, $\delta_P^-(i) := \delta_P^-(\{i\})$

Temps d'arrivées Nous allons voir ici comment nous pouvons nous passer de l'indice sur les véhicules dans les variables. Dans la formulation précédente, il est utile pour calculer le temps total de trajet pour chaque $k \in K$ et ainsi imposer la contrainte de durée. Nous reprenons ici la reformulation introduite dans Kara (2011) et Almoustafa et al. (2013) pour l'Asymmetric Distance-Constrained VRP (ADVRP) et qui n'est pas sans rappeler la technique proposée dans Bard et al. (1998).

Dans un graphe orienté $G = (V, A)$, soient x_{ij}, $ij \in A$ les variables arc-flot binaires usuelles avec $x_{ij} = 1$ si l'arc ij est utilisé. Soient z_{ij}, $ij \in A$ des variables réelles positives où z_{ij} représente la distance parcourue du dépôt central jusqu'au nœud j si son prédécesseur est le nœud i. Sa valeur est obtenue de la façon suivante :

$$\sum_{j \in V \setminus i} z_{ij} = \sum_{h \in V \setminus i} (z_{hi} + t_{hi} x_{hi}) \tag{3.12}$$

Cette technique peut être facilement adaptée à notre problème en utilisant les arcs de repositionnement vus précédemment.

Formulation à deux et trois indices Nous considérons ici les variables binaires des arcs de base x_{ij}, $ij \in A_0$ où $x_{ij} = 1$ si et seulement si le nœud j suit le nœud i dans une route, les variables binaires des arcs de réapprovisionnement $x_{i^p j}$, $i^p j \in A_P$ avec $x_{i^p j} = 1$ si et soulement si le véhicule recharge en p entre les clients i et j et les variables réelles positives des temps d'arrivées z_{ij}, $i, j \in V$ qui représentent la distance parcourue du dépôt central jusqu'au nœud j si son prédécessur est le nœud i.
La notation $x()$ introduite précédemment est étendue de la manière suivante : $x(A')$ est la somme $\sum_{ij \in A'} x_{ij}$ quelque soit $A' \subseteq A_0$, ou $\sum_{i^p j \in A'} x_{i^p j}$, quelque soit $A' \subseteq A_P$.

$$(\mathcal{M}_{RA}^{BC}) \quad \min \quad \sum_{ij \in A} d_{ij} x_{ij} + \sum_{i^p j \in A_P} d_{i^p j} x_{i^p j} \tag{3.13}$$

$$\text{s.c.} \quad x(\delta_0^-(\Delta)) + x(A_P) \geq \kappa(V_c) \tag{3.14}$$

$$x(\delta_0^+(i)) + x(\delta_P^+(i)) = x(\delta_0^-(i)) + x(\delta_P^-(i)) \quad \forall i \in V_c \tag{3.15}$$

$$x(\delta_0^+(i)) + x(\delta_P^+(i)) = 1 \quad \forall i \in V_c \tag{3.16}$$

$$x(\delta_0^-(\Delta)) \leq n_K \tag{3.17}$$

$$x(\delta_0^+(\Delta)) = x(\delta_0^-(\Delta)) \tag{3.18}$$

$$x(A_0(S)) \leq |S| - \kappa(S) \quad \forall S \in \mathcal{S}(V_c) \tag{3.19}$$

$$x(A_0(S)) + x(A_P(S)) \leq |S| - 1 \quad \forall S \in \mathcal{S}(V_c) \tag{3.20}$$

$$z_{0i} = t_{0i} x_{0i} \quad \forall i \in V_c \tag{3.21}$$

$$z_{iv} \geq (t_{0i} + t_{iv}) x_{iv} + \sum_{p \in P} (t_{0i} + t_{i^p v}) x_{i^p v} \quad \forall i \in V_c, v \in V \setminus i \tag{3.22}$$

$$z_{vi} \leq (T - t_{i0})(x_{vi} + \sum_{p \in P} x_{v^p i}) \quad \forall i \in V_c, v \in V \setminus i \tag{3.23}$$

$$z_{i0} \leq T x_{i0} \quad \forall i \in V_c \tag{3.24}$$

$$\sum_{v \in V \setminus i} (z_{iv} - z_{vi} - t_{vi} x_{vi} - \sum_{p \in P} t_{v^p i} x_{v^p i}) = 0 \quad \forall i \in V_c \tag{3.25}$$

$$x_{ij} \in \{0, 1\} \quad \forall ij \in A \tag{3.26}$$

$$x_{i^p j} \in \{0, 1\} \quad \forall ij \in A, p \in P \tag{3.27}$$

$$z_{ij} \in \mathbb{R} \quad \forall ij \in A \tag{3.28}$$

La fonction objectif (3.13) prend en compte les arcs de base et de réapprovisionnement. Les contraintes peuvent être divisées en trois grandes familles :

- contraintes de degré :
  - (3.14) assure un minorant sur le nombre de routes requises (cf (3.3)) ;
  - (3.15) et (3.16) imposent de visiter chaque client une et une seule fois ;
  - (3.17) et (3.18) imposent une borne égale à n_K pour le nombre de rotations et pour chacune d'elle d'entrer et de quitter le dépôt central une et une seule fois ;
- contraintes de capacité et de connectivité :
  - (3.19) forcent chaque route à respecter la capacité des véhicules ;
  - (3.20) assurent que chaque rotation est connectée à une séquence de routes ;
- contraintes des temps d'arrivées :
  - (3.21) déterminent les temps d'arrivées au premier client pour chaque rotation ;
  - (3.22) et (3.23) imposent respectivement des minorants et majorants sur les variables z_{ij} si l'un des arcs (de base ou de réapprovisionnement) de i à j est choisi, ou les mettent à 0 sinon ;
  - (3.24) imposent que chaque arc qui rentre dans le dépôt central, i.e. le dernier d'une rotation d'un véhicule, son temps d'arrivée doit être inférieur ou égal à T ;
  - (3.25) imposent récursivement les temps d'arrivées aux sommets clients d'une rotation.

Puisque les inégalités (3.19) et (3.20) sont en nombre exponentiel, nous avons donc développé un branch and cut pour résoudre cette formulation. Nous avons testé ce dernier sur les mêmes instances que dans la section précédente et nous obtenons de meilleurs résultats qu'avec le précédent branch and cut. En effet, nous arrivons à obtenir des gaps de quelques pourcents en limitant le temps de calcul à une heure et nous arrivons à améliorer certaines des meilleures solutions connues pour ces instances.

Décomposition et formulation étendue pour le VRPIRF

Nous proposons ici une formation avec un nombre exponentiel de variables basé sur une décomposition de Dantzig-Wolfe du modèle arc-flot proposé précédemment. Nous introduisons donc l'ensemble R de toutes les routes, avec $a_r^i = 1$ si la route $r \in R$ visite le sommet i, $e_r^{'p} = 1$ et $e_r^{''p} = 1$, avec $p \in V_p$ si respectivement la route r commence en p et se termine en p (de même pour $e_r^{'\Delta}$ et $e_r^{''\Delta}$), $b_r^{'s} = 1$ et $b_r^{''s} = 1$, avec $s \subseteq V \backslash \{\Delta\}$ si et seulement si la route r va de $V \backslash s$ vers s et va de s vers $V \backslash s$, $b_r^{ij} = 1$, avec $ij \in A$, si la route r visite i puis j en séquence, et $c_r = \sum_{ij \in A} b_r^{ij}\ d_{ij}$ et $t_r = \sum_{ij \in A} b_r^{ij}\ t_{ij} + \sum_{i \in V_C} a_r^i\ t_i + \sum_{p \in V_p} (e_r^{'p} + e_r^{''p})\ \frac{t_p}{2}$ le coût et la durée de la route r. Notons enfin $S_p = \{s \subset V : \Delta \notin s,\ p \in s\}$ la collection de sous-ensembles de nœuds pour les contraintes de connectivité en p.

Nous introduisons également des variables binaires x_r^k qui valent 1 si et seulement si la route $r \in R$ est dans la solution et est effectuée par le véhicule $k \in K$, des variables binaires $\tilde{x}^k$ si le véhicule $k \in K$ est utilisé et des variables binaires y_p^k qui valent 1 si au moins une route part de $p \in V_p$ avec le véhicule $k \in K$.

Nous obtenons la formulation suivante (avec les variables duales associées à chaque série de contraintes) :

$$(P) \quad \min \sum_{k\in K}\sum_{r\in R} c_r x_r^k \tag{3.29}$$

$$\text{s.t.} \sum_{r\in R}\sum_{k\in K} a_r^i\, x_r^k = 1 \qquad \forall i \in V_c \qquad \alpha_i \tag{3.30}$$

$$\sum_{r\in R} a_r^i\, e_r^{'p}\, x_r^k \le y_p^k \qquad \forall k \in K, p \in V_p, i \in V_c \qquad \varphi_{kpi} \tag{3.31}$$

$$\sum_{r\in R} (e_r^{'p} - e_r^{''p})\, x_r^k = 0 \qquad \forall k \in K, p \in V_p \qquad \theta_{kp} \tag{3.32}$$

$$\sum_{r\in R} t_r\, x_r^k \le T\, \tilde{x}^k \qquad \forall k \in K \qquad \beta_k \tag{3.33}$$

$$y_p^k \le \tilde{x}^k \qquad \forall k \in K, p \in V_p \qquad \sigma_{kp} \tag{3.34}$$

$$\sum_{r\in R} e_r^{'\Delta}\, x_r^k = \tilde{x}^k \qquad \forall k \in K \qquad \mu'_k \tag{3.35}$$

$$\sum_{r\in R} e_r^{''\Delta}\, x_r^k = \tilde{x}^k \qquad \forall k \in K \qquad \mu''_k \tag{3.36}$$

$$\sum_{r\in R} b_r^{'s}\, x_r^k \ge y_p^k \qquad \forall k \in K, p \in V_p, s \in S_p \qquad \delta'_{kps} \tag{3.37}$$

$$\sum_{r\in R} b_r^{''s}\, x_r^k \ge y_p^k \qquad \forall k \in K, p \in V_p, s \in S_p \qquad \delta''_{kps} \tag{3.38}$$

$$\tilde{x}^k, x_r^k, y_p^k \in \{0,1\} \qquad \forall k \in K, r \in R, p \in V_p \tag{3.39}$$

Les contraintes (3.30) assurent que chaque client sera visité par une seule route et par un seul véhicule. Les contraintes (3.31) permettent d'activer la variable y_p^k si le véhicule k effectue une route partant de p. Les équations (3.32) imposent que, pour chaque véhicule k, le nombre de routes partant de p soit égal au nombre de routes entrant en p. Les inégalités (3.33) imposent une durée maximale pour l'ensemble des routes effectuées par chaque véhicule k s'il est utilisé. Les contraintes (3.34) imposent que toute activité du véhicule k sur un dépôt intermédiaire p est interdite si le véhicule k n'est pas utilisé. Les contraintes (3.35) et (3.36) imposent que, soit k n'est pas utilisé, soit il doit quitter et revenir au dépôt central une et une seule fois. Enfin les contraintes (3.37) et (3.38) sont les contraintes de connectivité pour chaque véhicule k du dépôt central à chaque dépôt intermédiaire p et inversement.

Nous pouvons reformuler les contraintes (3.37) et (3.38) de la manière suivante :

$$\sum_{r\in R}\left(\sum_{\substack{i\in s\\ j\notin s}} (b_r^{ij} - b_r^{ji})\right) x_r^k \ge y_p^k \qquad \forall k \in K, p \in V_p, s \in S_p \qquad \delta'_{kps} \tag{3.40}$$

$$\sum_{r\in R}\left(\sum_{\substack{i\in s\\ j\notin s}} (b_r^{ji} - b_r^{ij})\right) x_r^k \ge y_p^k \qquad \forall k \in K, p \in V_p, s \in S_p \qquad \delta''_{kps} \tag{3.41}$$

Nous pouvons également redéfinir les collections de sous-ensembles S_p (et son complémentaire $\overline{S_p}$) comme :

$$S_p = \{s : s \subseteq V_p, p \in s\} \qquad ; \qquad \overline{S_p} = \{s : s \subseteq V_p, p \notin s\} = \mathcal{P}(V_p) \setminus S_p$$

et ainsi définir les contraintes (3.42) et (3.43) sur des sous-ensembles du graphe

des dépôts intermédiaires :

$$\sum_{r\in R} b_r^{'s}\, x_r^k \geq y_p^k \qquad \forall k \in K, s \subseteq V_p, p \in s \qquad \delta'_{kps} \qquad (3.42)$$

$$\sum_{r\in R} b_r^{''s}\, x_r^k \geq y_p^k \qquad \forall k \in K, s \subseteq V_p, p \in s \qquad \delta''_{kps} \qquad (3.43)$$

avec

$$b_r^{'s} = (e_r^{'\Delta} + \textstyle\sum_{p\notin s} e_r^{'p})(\sum_{p\in s} e_r^{''p}) \qquad (3.44)$$

$$b_r^{''s} = (\textstyle\sum_{p\in s} e_r^{'p})(e_r^{''\Delta} + \sum_{p\notin s} e_r^{''p}) \qquad (3.45)$$

Le coût réduit $\bar{c}_r^k$ des variables x_r^k est donné par :

$$\begin{aligned} & c_r - \sum_{i\in V_C} a_r^i.(\alpha_i^* + \sum_{p\in V_p} e_r^{'p}.\varphi_{kpi}^*) - \sum_{p\in V_p} (e_r^{'p} - e_r^{''p}).\theta_{kp}^* - t_r.\beta_k^* - e_r^{'\Delta}.\mu_k^{'*} \\ & \quad - e_r^{''\Delta}.\mu_k^{''*} - \sum_{p\in V_p}\sum_{s\in S_p} (b_r^{'s}.\delta_{kps}^{'*} + b_r^{''s}.\delta_{kps}^{''*}) \\ = & c_r - \sum_{i\in V_C} a_r^i.(\alpha_i^* + \sum_{p\in V_p} e_r^{'p}.\varphi_{kpi}^*) - \sum_{p\in V_p} (e_r^{'p} - e_r^{''p}).\theta_{kp}^* - t_r.\beta_k^* - e_r^{'\Delta}.\mu_k^{'*} \\ & \quad - e_r^{''\Delta}.\mu_k^{''*} - \sum_{s\subseteq V_p}\sum_{p\in s} (b_r^{'s}.\delta_{kps}^{'*} + b_r^{''s}.\delta_{kps}^{''*}) \end{aligned}$$

avec $(\alpha_i^*, \varphi_{kpi}^*, \theta_{kp}^*, \beta_k^*, \sigma_{kp}^*, \mu_k^{'*}, \mu_k^{''*}, \delta_{kps}^{'*}, \delta_{kps}^{''*})$ la solution duale du problème maître restreint courant. Nous pouvons aisément réécrire l'expression de $\bar{c}_r^k$ afin d'obtenir, comme sous problème, un problème de plus court chemin avec contraintes de ressources, que l'on peut résoudre par la programmation dynamique.
Nous avons développé un algorithme de branch and price pour résoudre cette reformulation et avons obtenu des premiers résultats encourageants.

Conclusion

Nous avons proposé plusieurs formulations pour le problème de tournées de véhicules avec dépôts de réapprovisionnement intermédiaires et nous avons développé différents algorithmes de branch and cut et branch and price afin de résoudre efficacement ces problèmes difficiles.

3.1.5 Résolution d'un problème de localisation et routage en logistique urbaine

Le problème du dernier kilomètre dans la distribution et le transport de marchandises est un problème complexe. Chaque jour, de très grandes quantités de marchandises sont transportées de l'extérieur vers l'intérieur et de l'intérieur vers l'extérieur des villes dans de gros camions, capables de transporter de grandes quantités mais qui sont inappropriés pour les livraisons et les ramassages en ville. Dans de tels cas, il est alors nécessaire d'effectuer des opérations de transbordement. Dans ce contexte, nous nous intéressons dans cette section à un problème de logistique urbaine que nous avons appelé "Multicommodity-Ring Location Routing Problem" (MRLRP), qui s'appuie sur l'utilisation de dépôts (potentiellement financés par des institutions publiques) appelés CDU (Centres de Distribution

Urbaine) qui collectent les marchandises entrantes et sortantes. Le système logistique combine alors des opérations de cross-docking sur le premier niveau où le transport est effectué par de gros véhicules, et des transports de détail sur le second niveau dans la ville. Chaque site potentiel d'installation d'un CDU possède un coût d'installation et une capacité, qui peut-être considérée comme une capacité de stockage à court-terme. Une fois les CDU installés, ils seront les points d'entrée et de sortie des chemins reliant les clients finaux. Les portes, situées dans les faubourgs de la ville, représentent les points d'accès à la ville via les routes principales. Les clients dans la ville peuvent avoir des demandes de livraison et de ramassage, caractérisées par la quantité à livrer ou à ramasser et par la porte concernée. Cependant les livraisons et ramassages doivent être acheminés séparément. De plus, un anneau doit être conçu afin de connecter les différents CDU entre eux. Ainsi les marchandises à livrer, après avoir été transportées de leur porte à un CDU, pourront soit quitter directement le CDU afin d'être acheminées au client, soit transiter via l'anneau vers un second CDU avant d'être acheminées.

Pour minimiser les transports à vide et pour des raisons environnementales, les transports de détails peuvent être ouverts, i.e. peuvent ne pas se terminer au CDU de départ. Pour les mêmes raisons, le système utilisera dans la ville des véhicules électriques à faible impact environnemental, qui auront donc une contrainte de longueur maximale de trajet et une contrainte de capacité. De plus, un système de véhicules de location en libre-service sera disponible et les stations PLS (Parkings en Libre Service) associées pourront servir comme points de départs ou d'arrivées additionnels pour les trajets ouverts. Les CDU et PLS partageront ainsi la même flotte de véhicules électriques. Enfin, pour simplifier le repositionnement de la flotte à la fin de la journée, des contraintes d'équilibrage de flotte seront imposées.

L'objectif sera alors de déterminer :

(i) un sous-ensemble de CDU à ouvrir et un anneau pour les connecter ;

(ii) les flots de marchandises entre les portes et les CDU et les flots sur l'anneau ;

(iii) l'affectation des demandes aux CDU via des trajets de livraison et de ramassage.

La fonction objectif consiste à minimiser la somme des coûts d'installation des CDU et des arcs de l'anneau, les coûts de transport des flots (sur le premier niveau) et les coûts de routage (sur le second niveau).
MRLRP est NP-difficile puisqu'il généralise le problème de tournées de véhicules avec capacité (CVRP : Capacitated Vehicle Routing Problem) dans le cas particulier d'un seul CDU de capacité infinie, pas de PLS, uniquement des livraisons et pas de contrainte de distance maximale sur les trajets de détail. Il appartient à la famille des problèmes de localisation-routage (LRP : Location-Routing Problems).

Le temps n'est ici pas pris en compte et nous considérons l'horizon comme étant une journée générique. De plus, les marchandises sont uniquement caractérisées par une quantité entière non divisible, l'unité de demande pourra donc être la tonne, des m^3, des containers ou encore des palettes.

La figure 3.9 présente les différents éléments d'une instance du problème ainsi qu'une solution.

Nous présentons une formulation pour ce problème. Une résolution exacte de ce modèle permet de résoudre des instances de petite taille. Afin de résoudre des instances de plus grande taille, nous proposons une matheuristique basée sur une décomposition du problème en 4 sous-problèmes, dont 3 d'entre eux sont résolus

optimalement. De plus, une approche hybride est proposée et testée pour résoudre des instances de taille moyenne et pour prouver l'efficacité de la matheuristique. Enfin, nous avons généré les instances de test pour ce problème à partir d'instances de la littérature pour le LRP.

La section suivante présente un état de l'art des problèmes proches du MRLRP ; puis la formulation est donnée avec l'introduction de différentes classes d'inégalités valides. La matheuritique et les résultats expérimentaux sont ensuite présentés, avant de conclure.

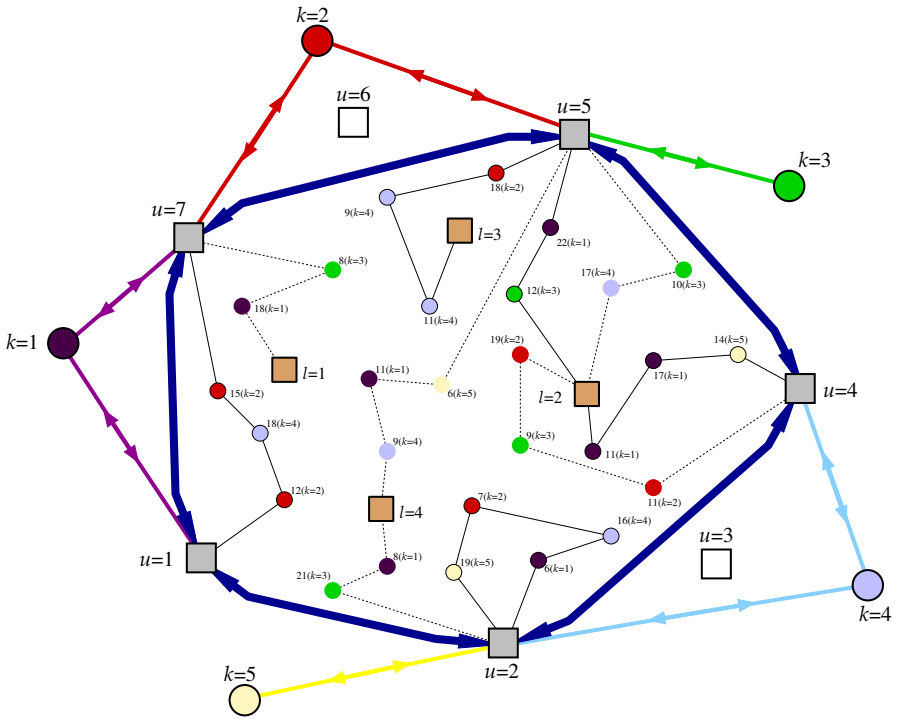


FIGURE 3.9 – *Un exemple d'instance de MRLRP et une solution réalisable. Elle dispose de 5 portes, associées à chaque commodité k (représentée par une couleur) ; 7 sites potentiels u pour installer les CDU ; 4 PLS l ; 15 demandes de livraison (ronds entourés) et 12 de ramassage (non entourés). Dans la solution, les chemins de livraison sont représentés par une ligne continue et ceux de ramassage par une ligne en pointillés. Tous les CDU et PLS ont des bornes d'équilibrage de flotte comprises entre -1 et +1, sauf le PLS l=4 qui a -2. Cinq CDU sont choisis et connectés par un anneau ; 3 portes utilisent le nombre maximal de CDU possible par porte, qui est 2 pour cette instance.*

État de l'art

Des états de l'art sur les problèmes de LRP peuvent être trouvés dans Laporte (1988), Laporte (1989), Min et al. (1998) et plus récemment dans Nagy et Salhi (2007).

Parmi cette classe de problème, le plus étudié est sans doute le "Capacitated Location-Routing Problem (CLRP)".

Il existe différentes variantes du LRP, avec par exemple des bornes sur le nombre de dépôts ouverts (Laporte et al. (1986), Laporte et Nobert (1981)), une flotte hétérogène (Ambrosino et Scutellà (2001)), des bornes sur la longueur des routes (Laporte et Nobert (1981), Berger et al. (2007)) ou la possibilité de faire plusieurs trajets pour chaque véhicule (Lin et al. (2002)), des routes ouvertes (Berger (1997)) ou encore des routes de livraison et de ramassage (Nagy et Salhi (1998)).

Il existe également des variantes du LRP qui ont des points communs avec le MRLRP. Dans Min (1996), l'auteur s'intéresse au problème d'installation de bornes de consolidation où à la fois les clients et les fournisseurs sont pris en compte. De même, dans Perl et Daskin (1985), les auteurs proposent d'introduire un niveau d'entrepôts pour définir le "Warehouse Location-Routing Problem (WLRP)". Dans Min (1996) et Perl et Daskin (1985), l'affectation des demandes des clients de la source d'origine à un dépôt intermédiaire crée une interaction importante entre les niveaux stratégiques et opérationnels.

Dans Singh (1998), le LRP-HTSP est introduit : seuls les dépôts et les clients sont considérés, comme dans le CLRP, mais les décisions de localisation sont aussi affectées par la contsruction d'un anneau qui connecte ces dépôts. Dans le MRLRP les aspects de niveau d'entrepôts et leur connection par un anneau sont pris en compte ; de plus, on considère les flots sur l'anneau pour représenter complètement le transport des marchandises des fournisseurs aux clients.

Le problème le plus proche du MRLRP est sans doute le "Many-to-Many LRP (MMLRP)", introduit dans Nagy et Salhi (1998). Résoudre le MMLRP consiste à choisir un ensemble de trajets de livraison/ramassage et de terminaux qui minimisent la somme des coûts fixes des terminaux et des coûts de transport. Les auteurs ont proposé une heuristique à deux niveaux dans laquelle la localisation des terminaux et le routage sont résolus heuristiquement, le premier problème jouant le rôle de maître et l'autre celui de sous problème, le tout dans un schéma hiérarchique. Le MMLRP généralise d'autres problèmes tels que le "VRP-PD (VRP with Pick-up and Delivery)" ou encore le "Hub Location Problem" et le "Many-to-Many Transportation Problem".

Une différence majeure entre le MRLRP et le MMLRP concerne l'aspect de conception de réseau puisque dans le MRLRP, nous devons construire un circuit Hamiltonien entre les dépôts sélectionnés alors qu'ils sont complètement connectés sans coût fixe dans le MMLRP. De plus, les structures des routes du second niveau sont très différentes.

Il existe également des liens avec le "Two-Echelon LRP (2E-LRP)" (Boccia et al. (2010; 2011), Nguyen et al. (2012b;a)) qui consiste à créer un système de distribution à 3 niveaux dans lequel le nombre et la localisation des dépôts de deux niveaux doivent être déterminés.

Dans Berger (1997), Berger propose un Branch-and-Price pour le "Distance-Constrained LRP", en utilisant une formulation du type partitionnement dans laquelle les variables binaires sont associées aux routes possibles. Berger et al. (2007) améliorent les travaux de Berger pour obtenir un meilleur branch and price (voir aussi Nemhauser et Savelsbergh (1996)) capable de résoudre optimalement des instances de 100 clients et 10 dépôts potentiels. Des inégalités valides sont présentées dans Berger (1997) et Berger et al. (2007) pour renforcer la relaxation continue. Une génération de colonnes est également proposée pour le LRP dans Akca et al. (2009) et un branch and cut dans Belenguer et al. (2011). Plus récemment, Baldacci et al. (2011) fournit une nouvelle approche exacte avec des bornes plus fines pour les versions avec et sans capacité du LRP en partant d'une formulation de type partitionnement. Des procédures de type "additive bounding", basée sur la programmation dynamique et des méthodes de type "dual ascent" sont introduites afin de décomposer le LRP en un ensemble de "Multi-Capacitated Depot VRPs (MCDVRPs)".

Graphe, données et notations

Le MRLRP est défini par un graphe mixte complet $G = (V, E \cup A)$, avec $V = K \cup U \cup L \cup P \cup D$. K est l'ensemble des portes, U l'ensemble des localisations possibles pour les CDU et L l'ensemble des PLS. P et D sont respectivement les ensembles de demandes de ramassage et de livraison. Chaque demande de ramassage $i \in P$ est définie par une quantité q_i à ramasser et une porte $k_i \in K$ où va la marchandise. Pour une demande de livraison $i \in D$, les mêmes notations sont utilisées pour la quantité q_i livrée depuis la porte k_i. Chaque CDU $u \in U$ a un coût d'installation fixe F_u et une capacité Q_u pour les marchandises concernées par les opérations de cross-docking dans u. Nous supposerons, sans perte de généralité, que $|U| > 3$ et que les capacités des CDU Q_u et les demandes totales $\sum_{i \in P \cup D} q_i$ garantissent qu'au moins trois CDU doivent être ouverts. D'un autre côté, le nombre de CDU à construire est majoré par $N \geq 3$ pour des raisons budgétaires. Dans la suite, nous identifions chaque porte $k \in K$ par une commodité. Notons $P_k = \{i \in P : k_i = k\}$ l'ensemble des demandes de ramassage qui iront à la porte $k \in K$, et $D_k = \{i \in D : k_i = k\}$ l'ensemble des demandes de livraison qui proviennent de la porte k. Les ensembles P_k forment ainsi une partition de P, et de même pour les ensembles D_k par rapport à D.
L'ensemble d'arcs A représente le premier niveau et est défini par $A = (K \times U) \cup (U \times K) \cup A_U$, avec $A_U = \{(u,v) : u,v \in U, u \neq v\}$ l'ensemble des arcs définis sur U. Chaque arc $(u,v) \in A_U$ a un coût de transport par unité de flot c_{uv} et une capacité q_{uv}. Des coûts de transport par unité de flot c_{ku} et c_{uk} sont aussi respectivement associés aux arcs $(k,u) \in K \times U$ et $(u,k) \in U \times K$. Une porte peut directement échanger des marchandises avec un nombre maximal B de CDU. L'ensemble d'arêtes E représente le second niveau ; il est défini par $E = E^p \cup E^d$, où $E^p = (U \cup L \cup P) \times P$ est l'ensemble des arêtes de ramassage et $E^d = (U \cup L \cup D) \times D$ est l'ensemble des arêtes de livraison. Les coûts associés aux arêtes $(i,j) \in E$ sont des coûts de routage c_{ij} proportionnels aux distances Euclidiennes. La partie de G qui représente le premier niveau est orienté puisque les coûts et les capacités des arcs sont potentiellement asymétriques et pour modéliser les flots dans les deux directions entre deux CDU connectés. Néanmoins, étant donnés $u, v \in U$, ils sont supposés être connectés dans chaque direction ou pas du tout ; c'est pourquoi nous définissons le sous ensemble d'arcs $A'_U = \{(u,v) : u,v \in U, u < v\}$ et un coût d'installation g_{uv} associé à chaque arc $(u,v) \in A'_U$, i.e. le coût de connection de u et v dans les deux sens. De même, les CDU avec lesquels une porte peut être reliée doivent être les mêmes pour la livraison et le ramassage.
Chaque CDU ou PLS $h \in U \cup L$ possède des bornes $-\delta_h^-$ et δ_h^+ pour imposer les contraintes d'équilibrage de flotte : le nombre de véhicules du second niveau en chaque CDU $u \in U$ et en chaque PLS $l \in L$ à la fin de la journée doit être le même qu'au début, modulo une déviation comprise dans $[-\delta_h^-, \delta_h^+]$. Une flotte illimitée et homogène de véhicules électriques est disponible dans chaque $u \in U$ et chaque véhicule a une capacité limitée q et une longueur maximale de trajet M. Dans la suite, le terme *route*, avec l'indice r, représentera une route du second niveau qui part d'un CDU (ou éventuellement d'un PLS pour un ramassage), visite séquentiellement un ensemble de clients et se termine en un CDU (ou éventuellement un PLS pour une livraison). Quand les points de départ et d'arrivée sont les mêmes, on parle d'itinéraire classique avec retour au dépôt, sinon on parle de route inter-dépôts. On note par $R_r = (i_{r_1}, \ldots, i_{r_{|R_r|}})$ la séquence de demandes ser-

vies par la route r, et par E_r le sous ensemble d'arêtes de E^p ou E^d (en fonction de $R_r \subset P$ ou $R_r \subset D$) qui composent la route r. Pour chaque route r, nous pouvons calculer sa charge $q(r) = \sum_{i \in R_r} q_i$ et son coût de routage $c(r) = \sum_{(i,j) \in E_r} c_{ij}$; le premier pouvant être classé en fonction de la porte/commodité en définissant $q_k(r) = \sum_{\substack{i \in R_r : \\ k_i = k}} q_i$. Une *route réalisable* est une route r telle que $q(r) \leq q$ et $c(r) \leq M$. L'ensemble de toutes les routes réalisables est noté par $\mathscr{R}$, partitionné en $\mathscr{R}^p$ et $\mathscr{R}^d$ en fonction du ramassage ou de la livraison. $\mathscr{R}_i$ est l'ensemble de toutes les routes qui desservent la demande $i \in P \cup D$. On définit $\mathscr{R}_h^+$ et $\mathscr{R}_h^-$, $h \in U \cup L$, comme les ensembles de routes qui respectivement partent et finissent en h ; et finalement $\mathscr{R}_u^{+d} = \mathscr{R}_u^+ \cap \mathscr{R}^d$ et $\mathscr{R}_u^{-p} = \mathscr{R}_u^- \cap \mathscr{R}^p$, les ensembles de routes de livraison et de ramassage qui respectivement partent et finissent en $u \in U$.

Formulation

Nous modélisons le MRLRP avec une formulation de type partitionnement qui utilise des variables associées aux chemins de routage plutôt qu'une formulation utilisant des variables à 2 ou 3 indices qui seraient associées aux arcs et aux véhicules. L'équivalence entre ces deux familles de modèles a déjà été prouvée pour divers problèmes de type VRP et notamment le CLRP Akca et al. (2009) ; de plus les formulations de type partitionnement fournissent une meilleure relaxation continue. Pour le MRLRP, l'utilisation de variables de type chemin sur le second niveau permet de mieux exprimer la relation entre les réseaux des deux niveaux. Les travaux de Berger (1997) et Berger et al. (2007) utilisent de telles formulations. Ce choix de modélisation ne sera par ailleurs pas affecté si l'on supprime l'hypothèse que les coûts du second niveau sont proportionnels aux distances Euclidiennes (i.e. symétriques et qui respectent les inégalités triangulaires).

Variables Nous définissons quatre familles de variables :

(i) les *variables binaires d'anneau* :
- $y_u = 1$ si un CDU est implanté sur le site $u \in U$, 0 sinon (variables de localisation)
- $z_{uv} = 1$ si les sites u et v sont directement connectés (dans les deux directions) dans l'anneau, 0 sinon, pour $(u,v) \in A'_U$ (variables de connectivité)

(ii) les *variables binaires de service* $\chi_{ku} = 1$ si la porte k échange directement des marchandises avec le CDU u, 0 sinon

(iii) les *variables binaires de routage* du second niveau $x_r = 1$ si la route $r \in \mathscr{R}$ est sélectionnée, 0 sinon

(iv) les *variables de flot* du premier niveau, qui sont postives et continues :
- φ_{ku} = flot de marchandises transportées de la porte k au site u
- φ_{uk} = flot de marchandises collectées du site u à la porte k
- φ_{uv}^{dk} = flot de marchandises de livraison de la commodité k qui passent par l'arc $(u,v) \in A_U$ (k-flots sortants)
- φ_{uv}^{pk} = flot de marchandises de ramassage de la commodité k qui passent par l'arc $(u,v) \in A_U$ (k-flots entrants)
- ϕ_{ku} = valeur maximale entre la quantité qui va directement de la porte k au site u et la quantité en provenance de la porte k et qui quitte u pour une route de livraison

– ϕ_{uk} = valeur maximale entre la quantité qui va directement du site u à la porte k et la quantité en provenance de la porte k et qui arrive en u avec une route de ramassage

Formulation de MRLRP Le programme linéaire en variables mixtes associé au MRLRP est le suivant :

$$\min \underbrace{\sum_{u\in U} F_u y_u}_{(A)} + \underbrace{\sum_{(u,v)\in A'_U} g_{uv} z_{uv}}_{(B)} + \underbrace{\sum_{(k,u)\in K\times U} c_{ku}\varphi_{ku} + \sum_{(u,k)\in U\times K} c_{uk}\varphi_{uk}}_{(C)} +$$

$$\underbrace{\sum_{(u,v)\in A_U} c_{uv} \sum_{k\in K} (\varphi^{pk}_{uv} + \varphi^{dk}_{uv})}_{D} + \underbrace{\sum_{r\in\mathscr{R}} c(r) x_r}_{(E)}$$

s.c.

$$\sum_{u\in U} \varphi_{ku} = \sum_{i\in D_k} q_i \qquad \forall k \in K \qquad (3.46)$$

$$\sum_{u\in U} \varphi_{uk} = \sum_{i\in P_k} q_i \qquad \forall k \in K \qquad (3.47)$$

$$\varphi_{uk} + \varphi_{ku} \le \chi_{ku} \sum_{i\in P_k\cup D_k} q_i \qquad \forall k \in K, u \in U \qquad (3.48)$$

$$\chi_{ku} \le y_u \qquad \forall k \in K, u \in U \qquad (3.49)$$

$$\sum_{u\in U} \chi_{ku} \le B \qquad \forall k \in K \qquad (3.50)$$

$$\varphi_{ku} + \sum_{\substack{v\in U\\ v\neq u}} \varphi^{dk}_{vu} = \sum_{\substack{v\in U\\ v\neq u}} \varphi^{dk}_{uv} + \sum_{r\in\mathscr{R}^{+d}_u} q_k(r) x_r \qquad \forall k \in K, u \in U \qquad (3.51)$$

$$\sum_{r\in\mathscr{R}^{-p}_u} q_k(r) x_r + \sum_{\substack{v\in U\\ v\neq u}} \varphi^{pk}_{vu} = \sum_{\substack{v\in U\\ v\neq u}} \varphi^{pk}_{uv} + \varphi_{uk} \qquad \forall k \in K, u \in U \qquad (3.52)$$

$$\sum_{r\in\mathscr{R}_i} x_r = 1 \qquad \forall i \in P \cup D \qquad (3.53)$$

$$x_r \le y_u \qquad \forall r \in \mathscr{R}^+_u \cup \mathscr{R}^-_u, u \in U \qquad (3.54)$$

$$-\delta^-_h \le \sum_{r\in\mathscr{R}^-_h} x_r - \sum_{r\in\mathscr{R}^+_h} x_r \le \delta^+_h \qquad \forall h \in U \cup L \qquad (3.55)$$

$$\sum_{v\in U: u<v} z_{uv} + \sum_{v\in U: v<u} z_{vu} = 2y_u \qquad \forall u \in U \qquad (3.56)$$

$$\sum_{\substack{u\in S\\ v\notin S\\ u<v}} z_{uv} + \sum_{\substack{u\notin S\\ v\in S\\ u<v}} z_{uv} \ge 2(y_w + y_{w'} - 1) \qquad \forall S \subset U : 3 \le |S| \le |U| - 3,$$

$$\forall w \in S, \forall w' \in U \setminus S \qquad (3.57)$$

$$\sum_{k\in K} (\varphi^{dk}_{uv} + \varphi^{pk}_{uv}) \le q_{uv}\, z_{uv} \qquad \forall (u,v) \in A'_U \qquad (3.58)$$

$$\sum_{k\in K} (\varphi^{dk}_{vu} + \varphi^{pk}_{vu}) \le q_{vu}\, z_{uv} \qquad \forall (u,v) \in A'_U \qquad (3.59)$$

$$\sum_{k\in K} (\phi_{ku} + \phi_{uk}) \le Q_u y_u \qquad \forall u \in U \qquad (3.60)$$

$$\phi_{ku} \ge \varphi_{ku} \qquad \forall k \in K, u \in U \qquad (3.61)$$

$$\phi_{ku} \ge \sum_{r\in\mathscr{R}^{+d}_u} q_k(r) x_r \qquad \forall k \in K, u \in U \qquad (3.62)$$

$$\phi_{uk} \geq \varphi_{uk} \qquad \forall k \in K, u \in U \qquad (3.63)$$

$$\phi_{uk} \geq \sum_{r \in \mathscr{R}_u^{-p}} q_k(r) x_r \qquad \forall k \in K, u \in U \qquad (3.64)$$

$$\sum_{u \in U} y_u \leq N \qquad (3.65)$$

$$y_u \in \{0,1\} \qquad \forall u \in U \qquad (3.66)$$

$$z_{uv} \in \{0,1\} \qquad \forall (u,v) \in A'_U \qquad (3.67)$$

$$\chi_{ku} \in \{0,1\} \qquad \forall k \in K, u \in U \qquad (3.68)$$

$$x_r \in \{0,1\} \qquad \forall r \in \mathscr{R} \qquad (3.69)$$

$$\varphi_{uv}^{pk}, \varphi_{uv}^{dk} \geq 0 \qquad \forall k \in K, (u,v) \in A_U \qquad (3.70)$$

$$\varphi_{ku}, \varphi_{uk}, \phi_{uk}, \phi_{ku} \geq 0 \qquad \forall k \in K, u \in U \qquad (3.71)$$

La fonction objectif minimise la somme des coûts d'installation des CDU sélectionnés (A), des coûts de connection sur l'anneau (B), des coûts de transport entre les portes et les CDU (C), des coûts de transport entre CDU sur l'anneau (D) et enfin des coûts de routage sur les chemins de livraison et de ramassage du second niveau (E).

Les contraintes (3.46) et (3.47) expriment que le flot qui quitte ou arrive à la porte k est égal à la quantité totale de demandes qui proviennentt et qui vont en k, alors que les contraintes (3.48) forcent le flot de la commodité k qui entre ou sort de u à être égal à 0 si la commodité k n'échange pas directement de marchandises avec le site u ; les contraintes (3.49) expriment la relation directe entre la localisation et les variables de service, alors que les contraintes (3.50) majore à B le nombre de CDU auxquels une porte peut envoyer et recevoir des demandes. Les contraintes de conservation de flot (3.51) assurent que pour chaque $u \in U$ et $k \in K$, la quantité totale de la commodité k qui arrive en u, soit directement depuis k, soit depuis l'anneau (k-flots sortants), soit égale à la quantité totale qui quitte u pour aller sur l'anneau (k-flots sortants) ou qui quitte u pour des chemins de livraison. L'inverse est aussi valable pour les flots de ramassage grâce aux contraintes (3.52). Les contraintes (3.53) affectent chaque livraison ou ramassage à exactement une route. Les contraintes (3.54) assurent qu'aucune route ne peut démarrer d'un CDU s'il n'est pas sélectionné. (3.55) sont les contraintes d'équilibrage de flotte en chaque CDU et PLS. Les relations (3.56) et (3.57) concernent la construction de l'anneau avec les CDU sélectionnés. (3.57) sont les contraintes d'élimination de sous-tours adaptées au fait que les sommets qui doivent être connectés sur l'anneau ne sont pas connus a priori. (3.58) et (3.59) sont les contraintes de capacité sur les arcs de l'anneau. Notons que, étant donnés deux CDU $u, v \in U$ tels que $u < v$, tous les flots sur (u,v) et (v,u) dépendent de z_{uv}.
(3.60)–(3.64) sont les contraintes de stockage dans les CDU. En effet, supposons qu'une demande de livraison $i \in D$ soit transportée de k_i au CDU u puis expédiée à partir du CDU v. Pour modéliser les opérations de cross-docking, on considère que i occupe de la capacité à la fois sur u et sur v (alors que les éventuels CDU intermédiaires ne sont pas concernés), à moins qu'une expédition directe soit réalisée (i.e. $v \equiv u$), puisque dans ce cas uniquement q_i est occupée, et non $2 \cdot q_i$. Ainsi, étant donné $k \in K$, l'occupation de u par les demandes D_k n'est pas la somme des φ_{ku} et $\sum_{r \in \mathscr{R}_u^{+d}} q_k(r) x_r$ (i.e. la somme des demandes D_k expédiées depuis u), mais le maximum des deux. Les termes ϕ_{ku} et ϕ_{uk} représentent respectivement ces quantités pour les livraisons ((3.61), (3.62)) et les ramassages ((3.63), (3.64)) ; et donc la somme des termes de gauche de (3.60) est un majorant de l'oc-

cupation de u. Une conséquence de ce choix de modélisation est que la condition $\sum_{i \in D \cup P} q_i > \max_{u,v \in U, u \neq v} (Q_u + Q_v)$ permet de garantir qu'au moins trois CDU seront ouverts, puisque les termes de gauche représentent la valeur minimale de l'occupation complète des CDU, ce qui arrive dans le cas particulier où il n'y a que des expéditions directes.
Enfin, (3.65) est la contrainte de budget sur le nombre maximal de CDU qui peuvent être construits.

Renforcement du modèle

Nous ajoutons ici des contraintes qui sont redondantes pour la formulation initiale, mais utile pour accélérer la résolution. Nous montrons de plus comment écrire les contraintes d'élimination de sous-tours de manière plus efficace et nous ajoutons quelques inégalités valides connues dans la littérature.

Inégalités logiques Les contraintes (3.54) et la fonction objectif sont suffisantes pour relier les variables y et x_r. Néanmoins, nous ajoutons les contraintes complémentaires (3.72) :

$$y_u \leq \sum_{r \in \mathscr{R}_u^{+d}} x_r + \sum_{r \in \mathscr{R}_u^{-p}} x_r \quad \forall u \in U \tag{3.72}$$

Ces contraintes affirment principalement qu'il doit y avoir des routes de livraison partant de u, ou de ramassage finissant en u, pour justifier l'ouverture de u.

Inégalités d'élimination de sous-tours Les contraintes (3.56) et (3.57) sont des variantes des contraintes classiques d'élimination de sous-tours dans la cas où les sommets à connecter dans le tour ne sont pas connus à l'avance. Les contraintes d'élimination de sous-tours (3.57) proviennent des travaux sur les contraintes de coupe généralisée et sur le TSP généralisé Fischetti et al. (1997). Cependant, les contraintes (3.57) peuvent être remplacées de la manière suivante. Nous ajoutons de nouvelles variables $\zeta_u \in \{0,1\}$ pour chaque $u \in U$ et remplaçons (3.57) par les contraintes suivantes :

$$\zeta_u \leq y_u \quad \forall u \in U \tag{3.73}$$

$$\sum_{u \in U} \zeta_u = 1 \tag{3.74}$$

$$\sum_{\substack{u,v \in S, \\ u<v}} z_{uv} \leq \sum_{u \in S} y_u + 1 - y_w - \sum_{u \in S} \zeta_u \quad \forall S \subset U : 3 \leq |S| \leq |U| - 3, \forall w \in U \setminus S \tag{3.75}$$

Nous réduisons ainsi de manière drastique le nombre de contraintes puisque si l'on compare (3.75) et (3.57), pour chaque sous-ensemble S nous avons désormais $\mathcal{O}(|U|)$ contraintes au lieu de $\mathcal{O}(|U|^2)$. L'équivalence entre les contraintes (3.73)–(3.75) et (3.57) et la preuve de cette équivalence ont été inspirées par Fischetti et al. (1997).

Lemme 2. *L'ensemble de contraintes* (3.56) *et* (3.73)–(3.75) *permettent d'interdire les sous-tours pour le MRLRP, i.e. quand les sommets à connecter ne sont pas connus a priori.*

Preuve 14. *Soit $\mathcal{S}_U = \{S \subset U : 3 \leq |S| \leq |U| - 3\}$ les sous-ensembles de U pour les contraintes* (3.57) *ou* (3.75)*, et soit S un élément de $\mathcal{S}_U$. Supposons, sans perte de généralité, que $\sum_{u \in U} y_u \geq 3$. Nous pouvons distinguer quatre cas :*

Cas 1 *Si $\sum_{u \in S} y_u = 0$, alors à partir de* (3.73)-(3.75) *on obtient $\sum_{u,v \in S : u<v} z_{uv} \leq 0$, i.e. qu'aucune arête ne peut avoir ses deux extrémités dans S.*

Cas 2 *Si $0 < \sum_{u \in S} y_u < \sum_{u \in U} y_u$ et $\sum_{u \in S} \zeta_u = 1$, alors il existe $w \in U \setminus S$: $y_w = 1$, telle que la contrainte* (3.75) *devient :*

$$\sum_{\substack{u,v \in S \\ u<v}} z_{uv} \leq \sum_{u \in S} y_u - 1$$

et donc les sous-tours dans S sont interdits.

Cas 3 *Si $0 < \sum_{u \in S} y_u < \sum_{u \in U} y_u$ et $\sum_{u \in S} \zeta_u = 0$, alors il existe également $w \in U \setminus S : y_w = 1$, telle que la contrainte* (3.75) *devient :*

$$\sum_{\substack{u,v \in S \\ u<v}} z_{uv} \leq \sum_{u \in S} y_u$$

ce qui n'empêche pas d'avoir un sous-tour dans S, mais la contrainte (3.75) *sur $U \setminus S$ (qui appartient aussi à $\mathcal{S}_U$) avec $w \in S : y_w = 1$ devient :*

$$\sum_{u,v \in U \setminus S} z_{uv} \leq \sum_{u \in U \setminus S} y_u + 1 - y_w - \sum_{u \in U \setminus S} \zeta_u = \sum_{u \in U \setminus S} y_u - 1$$

ce qui empêche les sous-tours dans $U \setminus S$, et par conséquence dans S.

Cas 4 *Si $\sum_{u \in S} y_u = \sum_{u \in U} y_u$ (i.e. $\sum_{u \in U \setminus S} y_u = 0$), alors $y_w = 0$ pour tout $w \in U \setminus S$ ce qui implique $\sum_{u \in U} \zeta_u = 1$. Pour tout $w \in U \setminus S$, la contrainte devient*

$$\sum_{u,v \in S : u<v} z_{uv} \leq \sum_{u \in S} y_u$$

Un seul sous-tour est alors autorisé dans S. En effet, s'il existe un sous-tour dans $S' \subset S$, ce sous-tour sera éliminé par la contrainte (3.75) *sur S' au lieu de S. Ce qui termine la preuve.*□

Inégalités chemins En reprenant une idée figurant dans Berger (1997), qui a été ensuite réutilisée dans Berger et al. (2007), nous remplaçons les contraintes (3.54) par :

$$\sum_{r \in \mathscr{R}_i \cap (\mathscr{R}_u^+ \cup \mathscr{R}_u^-)} x_r \leq y_u \qquad \forall u \in U, \forall i \in P \cup D \tag{3.76}$$

Ces contraintes sont valides puisqu'au plus une route parmi celles partant ou finissant en u peut être sélectionnée pour servir la demande i. De plus, les contraintes (3.54) sont en nombre exponentiel, alors qu'il y a un nombre polynomial de contraintes (3.76). Enfin, le remplacement de ces contraintes nous permet d'avoir une relaxation continue de meilleure qualité. Dans Berger (1997), l'auteur applique cette substitution à une formulation de type partitionnement pour le LRP, prouvant que ces nouvelles contraintes sont des coupes de Chvátal de rang 1.

Matheuristique

Cette section présente GALW, une matheuristique qui permet de résoudre des instances de grande taille de MRLRP.
L'approche exacte échoue généralement à résoudre les instances de taille moyenne à cause du trop grand nombre de routes possibles ; de plus le temps de résolution croît très vite en raison du très grand nombre de contraintes (notamment d'affectation, de STSP (Symmetric Traveling Salesman Problem) et de flots), qui sont fortement interconnectées. Le choix d'une heuristique pour le MRLRP n'est pas trivial. Sa complexité, son hétérogénéité et l'interdépendance des différentes décisions (localisation, conception du réseau, routage, flot) rendent l'élaboration de recherches locales, fondées sur des voisinages, difficile.
Nous avons par conséquent privilégié une approche de type matheuritique fondée sur une décomposition du problème. GALW utilise deux principales idées : ne générer qu'un sous-ensemble de routes et décomposer le modèle en étapes successives. Chacune de ces étapes est dévolue à un aspect du problème et est résolue à l'optimalité. Pour générer les routes, de façon heuristique, nous utilisons un algorithme du type plus proche voisin modifié pour prendre en compte le chargement et la longueur du trajet pour les véhicules du second niveau. Les trois autres étapes consistent en la résolution d'un programme linéaire en nombres entiers associé à un problème d'affectation généralisée, d'un STSP et d'un problème de multiflot sur un anneau.

Générateur de routes Soient $R = (i_1, \ldots, i_{|R|})$ une séquence de demandes, $r(R) \in \mathscr{R}$ la route (entre dépôts ou avec retour au dépôt) obtenue en ajoutant un point de départ qui est le plus proche de i_1 et un point d'arrivée qui est le plus proche de $i_{|R|}$. R est une séquence valide de demandes si $c(r(R)) \leq M$ et $q(r(R)) \leq q$, i.e. si r est une route réalisable. Il existe de nombreuses autres routes que $r(R)$ que l'on peut obtenir à partir de R. Nous construisons l'ensemble $\overline{\mathscr{R}} \subset \mathscr{R}$ de routes en utilisant une procédure de plus proche voisin. Notons $\overline{\mathscr{R}}_i \subset \mathscr{R}_i$ l'ensemble des routes desservant la demande i. L'algorithme assure que $\overline{\mathscr{R}}$ contient des routes de différentes longueurs maximales $M^t = \beta^t M \leq M$, avec $\beta < 1$, $t = 0 \ldots t^*$, puisqu'il serait difficile de trouver des solutions si toutes les routes avaient toutes la même longueur maximale M. La génération utilise les paramètres β et τ, $0 < \tau < \beta < 1$; elle commence avec $t = 0$ et augmente progressivement t jusqu'à ce que $\beta^t < \tau$; et donc, $t^* = \lfloor \log_\beta \tau \rfloor$. Chaque nouvelle route est initialisée avec une demande i, qui est choisie aléatoirement parmi celles dont le nombre de routes associées est inférieur à la moyenne sur toutes les demandes. L'algorithme effectue donc itérativement les deux étapes suivantes jusqu'à ce qu'aucune demande ne puisse être insérée sans dépasser M^t ou q :

1. Choix d'une demande i à ajouter à la séquence courante R selon la règle du plus proche voisin minimisant la fonction $d(R,i)$ ci-dessous ;
2. Amélioration de R via une recherche locale qui consiste en une séquence d'échanges 2-opt sur $r(R)$ tant qu'une amélioration est possible.

La fonction à minimiser est :

$$d(R,i) = p_q \frac{q(r)+q_i}{q} + p_M \frac{\min\{c(r(i,i_1..i_{|R|})), c(r(i_1..i_{|R|},i))\}}{M} + p_\omega \frac{|\overline{\mathscr{R}}_i|}{\omega}$$

où le second terme est le minimum entre le coût d'insertion d'une nouvelle demande i au début ou à la fin de R, et $p = (p_q, p_M, p_\omega)$ est un vecteur de poids. La seule différence avec le 2-opt classique (voir e.g. Lin et Kernighan (1973)) est qu'un échange qui modifie la première (resp. la dernière) demande visitée peut aussi modifier le plus proche point de départ (resp. d'arrivée) de la route. La figure 3.10 explique la différence entre un échange qui ne modifie pas les points de départ et d'arrivée d'une route (b) et ce qui arrive quand un tel échange est effectué (c). La t-ème itération se termine quand chaque demande i est couverte par au moins ω routes afin de garantir que $\overline{\mathscr{R}}_i$ contient un nombre minimal de routes. Comme β, ω et τ sont constants, la complexité de la phase de génération de route est en $\mathcal{O}(n^4)$. On utilise $\overline{\mathscr{R}}$ comme ensemble de routes réalisables par la suite.

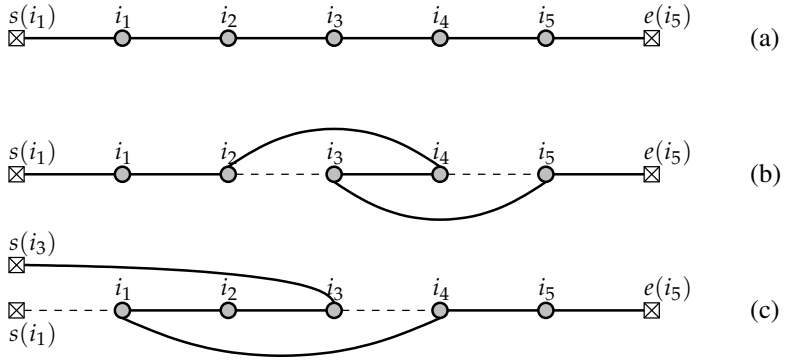


FIGURE 3.10 – *Illustration du principe du 2-opt lors de la génération des routes. On appelle respectivement $s(i)$ et $e(i)$ les points de départ et d'arrivée les plus proches pour la demande i. Quand l'échange ne modifie pas la première ni la dernière demande, le calcul n'est pas modifié (b), et vaut $c_{i_2,i_3} + c_{i_4,i_5} - (c_{i_2,i_4} + c_{i_3,i_5})$. Mais si la première demande change (c), le point de départ peut alors être modifié (dans l'exemple $s(i_3) \neq s(i_1)$) : le calcul sera alors $c_{s(i_1),i_1} + c_{i_3,i_4} - (c_{s(i_3),i_3} + c_{i_1,i_4})$.*

Sous-problème d'affectation La deuxième étape de la matheuristique GALW concerne le routage, l'activation des CDU et l'affectation des demandes. Les deux niveaux de distribution sont disjoints dans le graphe : le problème est résolu en affectant les demandes du premier niveau (des portes aux CDU) et du deuxième niveau et en prenant en compte les capacités des CDU. Nous n'avons pas besoin ici de construire l'anneau ni de router les flots sur celui-ci ; si une demande de livraison $i \in D_k$ est transportée de la porte k au CDU u et plus tard livrée à partir d'un autre CDU v, alors nous avons soit une expédition directe si $u = v$, soit une utilisation implicite du réseau externe dans le cas contraire ; de même pour une demande de ramassage. Nous utilisons un sous-ensemble des variables du modèle général du MRLRP : les variables de localisation y_u, $u \in U$, les variables de routage x_r, avec $r \in \overline{\mathscr{R}}$, les variables de service χ_{ku}, $(k,u) \in K \times U$ et un sous-ensemble des variables de flot : φ_{ku}, φ_{uk}, ϕ_{ku} et ϕ_{uk}, $k \in K$, $u \in U$. Toutes ces variables ont la même signification et les mêmes domaines de définition que dans le modèle général, sauf évidemment pour les variables de routage x_r. La fonction objectif est composée des termes (A), (C) et (E) du modèle général. Les contraintes sont également un sous-ensemble de celles du modèle général : (3.46)–(3.50), (3.53), (3.55), (3.60)–(3.65), (3.72) and (3.76), ainsi que les contraintes de bornes sur les variables. Les variables et contraintes supprimées concernent l'anneau et le flot associé. Comme ce modèle n'utilise pas les termes (B) et (D) dans sa fonction

objectif, on peut décider d'ouvrir un sous-ensemble de CDU qui seront par la suite reliés par un anneau dont le coût fixe et le coût de transport seront très importants. Nous devons donc estimer ces coûts a priori.

Pour cela, nous cherchons une solution du STSP sur tous les sous-ensembles $O \subset U$ de CDU tels que $3 \leq |O| \leq N$; ensuite, pour tout $u \in U$ et tout $o = 3 \dots N$, nous déterminons le coût $F_{u,o} = \frac{1}{2} g_{uv_o} + \frac{1}{2} g_{uw_o}$, où v_o et w_o sont les sommets connectés à u dans l'anneau de o sommets, le moins coûteux parmi ceux contenant u. On peut noter que la fonction $F_{u,o}$ peut être pré-calculée a priori lors d'une phase de pré-traitement. Nous ajoutons ensuite au modèle un nouvel ensemble ξ_o^u de variables binaires, $u \in U, o = 3 \dots N : \xi_o^u = 1$ signifie que $u \in O$ et $|O| = o$. Enfin, nous introduisons les contraintes suivantes :

$$\sum_{v \in U \setminus u} y_v \geq \sum_{o=3}^{N} (o-1) \xi_o^u \qquad \forall u \in U \qquad (3.77)$$

$$\sum_{v \in U \setminus u} y_v \leq \sum_{o=3}^{N} (o-1) \xi_o^u + N(1 - y_u) \qquad \forall u \in U \qquad (3.78)$$

$$\xi_o^u \in \{0,1\} \qquad \forall u \in U, o \in \{3..N\} \qquad (3.79)$$

et nous ajoutons le terme suivant à la fonction objectif $\sum_{u \in U} \sum_{o=3}^{N} F_{u,o} \xi_o^u$, terme qui repésente un minorant du coût fixe d'installation de l'anneau. L'impact des coûts de transport du flot et des capacités des arcs de l'anneau est le suivant : pour chaque commodité k et chaque CDU u, les termes $(\varphi_{ku} - \sum_{r \in \overline{\mathscr{R}}_u^{+d}} q_k(r) x_r)$ et $(\sum_{r \in \overline{\mathscr{R}}_u^{-p}} q_k(r) x_r - \varphi_{uk})$, quand ils sont positifs, représentent les quantités de marchandises qui doivent quitter u ; cela nous permet de calculer la quantité totale de marchandises que u doit envoyer sur l'anneau, pour laquelle nous pouvons imposer un majorant et calculer un minorant du coût de transport à ajouter à l'objectif. Pour cela, nous considérons respectivement les deux arcs $(u,v), (u,w) \in A_U$ tels que la capacité totale $q_{uv} + q_{uw}$ soit maximale et les deux arcs $(u,v'), (u,w') \in A_U$ qui minimisent le coût par unité de flot $\frac{1}{2}(c_{uv'} + c_{uw'})$.

Construction de l'anneau Étant donné $O \subset U$ le sous-ensemble de CDU sélectionnés, la troisième étape de GALW consiste à trouver une façon de les connecter par un anneau à coût minimum. Les coûts ici sont $g_{uv}, (u,v) \in A'_U, u, v \in O$, i.e. les coûts pour connecter deux CDU u et v dans les deux sens. Ainsi, même si les sous-graphes d'origine sont orientés, le problème ici est un TSP symétrique (STSP). Cette étape est donc également NP-difficile et est résolue en utilisant CONCORDE (Applegate et al. (2006)). Les flots sur l'anneau peuvent circuler dans les deux directions : *dans le sens des aiguilles d'une montre* et *dans le sens inverse des aiguilles d'une montre*.

Flots sur l'anneau Une fois résolu le sous-problème d'affectation et l'anneau construit, nous devons router les flots sur celui-ci pour les expéditions indirectes. Nous avons deux fois le nombre de commodités d'origine (i.e. $2|K|$) puisque l'on distingue entre le ramassage et la livraison. Pour chaque commodité k, et pour le ramassage comme pour la livraison, nous avons des CDU avec des marchandises disponibles, i.e. :

- ceux qui ont reçu des marchandises d'une porte (cas $\varphi_{ku} > \sum_{r \in \overline{\mathscr{R}}_u^{+d}} q_k(r) x_r$) ou d'une route de ramassage (cas $\sum_{r \in \overline{\mathscr{R}}_u^{-p}} q_k(r) x_r > \varphi_{uk}$) ;

– ceux avec une demande de marchandises, car ils doivent l'expédier vers une porte (cas $\sum_{r\in\overline{\mathscr{R}}_u^{-p}} q_k(r)x_r < \varphi_{uk}$), ou vers un client en livraison (cas $\varphi_{ku} < \sum_{r\in\overline{\mathscr{R}}_u^{+d}} q_k(r)x_r$) ;
– et enfin ceux non impliqués par ces cas.

Dans la suite, nous noterons, pour les livraisons de la commodité k, $O_{dk}^{+} \subset O$ le sous-ensemble de CDU sélectionnés qui sont disponibles et $O_{dk}^{-} \subseteq O \setminus O_{dk}^{+}$ le sous-ensemble des CDU sélectionnés avec une demande. De plus, soit $O_{dk}^{2} = \{\{u,v\} : u \in O_{dk}^{+}, v \in O_{dk}^{-}\}$ l'ensemble des paires source–puits $\{u,v\}$, et Φ_{uv}^{dk} le flot à envoyer de la source u au puits v, toujours pour la livraison de la commodité k. La disponibilité totale des sites $u \in O_{dk}^{+}$ est égale à la demande totale des sites $v \in O_{dk}^{-}$; les marchandises devant être envoyées des premiers aux derniers tel que chaque $u \in O_{dk}^{+}$ consomme complètement sa disponibilité et que les besoins de chaque $v \in O_{dk}^{-}$ soient satisfaits. Nous obtenons ainsi un ensemble de contraintes de conservation de flots sur les variables Φ_{uv}^{dk} (et Φ_{uv}^{pk} pour chaque k). Soient respectivement $\theta'_{\{uv\}}$ et $\theta''_{\{uv\}}$ les chemins dans le sens des aiguilles d'une montre et dans le sens inverse des aiguilles d'une montre qui vont de u à v, $u,v \in O$, sur l'anneau. Chaque flot que nous devons router de u à v peut être séparé et peut donc être partiellement envoyé dans le sens des aiguilles d'une montre et dans le sens inverse des aiguilles d'une montre, i.e. sur $\theta'_{\{uv\}}$ ou $\theta''_{\{uv\}}$. Cela signifie qu'un arc a de l'anneau orienté dans le sens des aiguilles d'une montre doit transporter tous les flots (indépendamment de la commodité) de u à v pour chaque paire $\{u,v\}$ telle que $a \in \theta'_{\{uv\}}$; la même chose est valable pour les arcs orientés dans le sens inverse des aiguilles d'une montre. La dernière étape de GALW consiste donc en un problème de multiflot sur un anneau, avec plusieurs sources et plusieurs puits pour chaque commodité, problème qui sera résolu via un programme linéaire, qui prendra en compte l'équilibre entre les disponibilités et les demandes associées aux ramassages et aux livraisons de chaque commodité et qui décidera comment séparer les flots afin de respecter les capacités des arcs et minimiser les coûts des flots. Le flot sur chaque arc a de l'anneau est :

$$
\begin{aligned}
f_a &= \textstyle\sum_k (\varphi_a^{dk} + \varphi_a^{pk}) \\
&= \begin{cases} \textstyle\sum_k \left(\sum_{\{u,v\}\in O_{dk}^2 : a\in\theta'_{\{uv\}}} \Phi_{uv}^{\prime dk} + \sum_{\{u,v\}\in O_{pk}^2 : a\in\theta'_{\{uv\}}} \Phi_{uv}^{\prime pk}\right) & \text{, } a \text{ orienté dans le sens} \\ & \text{des aiguilles d'une montre} \\ \textstyle\sum_k \left(\sum_{\{u,v\}\in O_{dk}^2 : a\in\theta''_{\{uv\}}} \Phi_{uv}^{\prime\prime dk} + \sum_{\{u,v\}\in O_{pk}^2 : a\in\theta''_{\{uv\}}} \Phi_{uv}^{\prime\prime pk}\right) & \text{, sinon} \end{cases}
\end{aligned}
\tag{3.80}
$$

où $\Phi_{uv}^{\prime dm}$ et $\Phi_{uv}^{\prime\prime dm}$ sont les parties de Φ_{uv}^{dm} que nous envoyons dans le sens des aiguilles d'une montre et dans le sens inverse des aiguilles d'une montre. Les flots φ_a^{dm} et φ_a^{pm} correspondent aux flots sortants et entrants du modèle général du MRLRP : les contraintes du problème de multiflot sur l'anneau sont donc simplement les contraintes de capacité sur les arcs a de l'anneau (voir (3.58)–(3.59)) et les contraintes de conservation de flots sur les variables Φ_{uv}^{dk} and Φ_{uv}^{pk}.
L'objectif se réduit au terme (D) du modèle général. Puisque le programme linéaire considéré est continu, cette quatrième étape de GALW est donc polynomiale. La figure 3.11 montre un exemple.

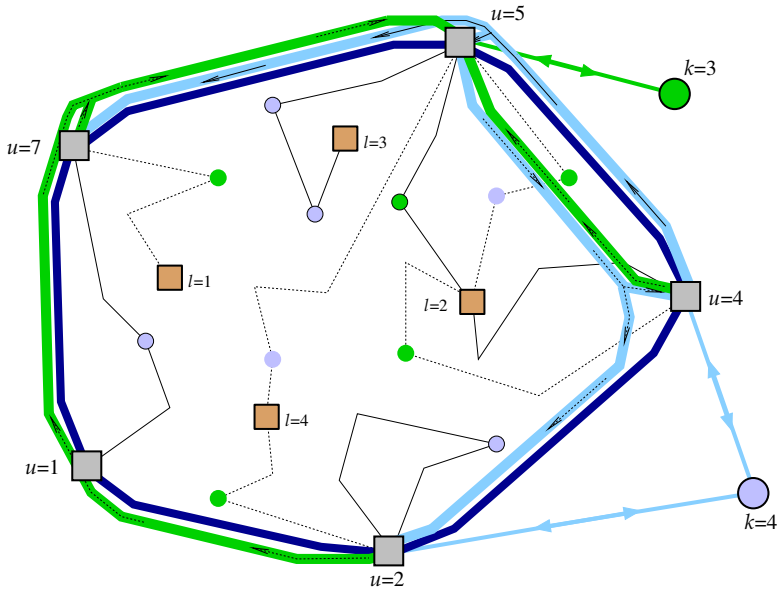


FIGURE 3.11 – *Une solution de l'étape de multiflot sur l'anneau. L'étape d'affectation fournit les CDU sélectionnés et détermine les chemins du second niveau et les flots CDU-porte ; une fois l'anneau construit, le problème de multiflot sur l'anneau cherche à satisfaire les expéditions indirectes à coût minimum. Dans cet exemple, les chemins du second niveau et les flots CDU-porte sont les mêmes que ceux de la figure 3.9. Comme pour les chemins du second niveau, les flots sur l'anneau correspondant à des livraisons sont avec des flèches continues, alors que les ramassages sont en pointillés. Nous ne considérons ici que les commodités k=3 et k=4.*

Résultats expérimentaux

- La *méthode exacte* correspond à la résolution, via un solveur, de la formulation renforcée par les inégalités valides, nous l'appellerons la *méthode X*.
- La *matheuristique* est l'algorithme GALW. En l'implémentant, nous avons intégré deux astuces lors de la résolution du sous-problème d'affectation : la première consiste, quand l'instance contient b arcs $(u,v) \in A'_U$ avec des coûts d'installation g_{uv} beaucoup plus grands que la moyenne, à modifier légèrement l'objectif en imposant de payer g_{uv} si u et v sont sélectionnés ; la deuxième consiste à arrêter le solveur quand le saut d'intégrité atteint un seuil α, on sauvegarde alors les S premières solutions et les deux dernières étapes de GALW (peu coûteuses en temps) sont effectuées sur l'ensemble de ces solutions. Seule la meilleure solution globale est retenue à la fin.
- La méthode *hybride*, que nous appellerons *méthode Y*, consiste à résoudre la formulation en utilisant $\overline{\mathscr{R}}$ au lieu de $\mathscr{R}$. Cette méthode est utile pour vérifier l'efficacité du générateur de routes de GALW.

La génération des instances est brièvement décrite ci-dessous. Le solveur utilisé est CPLEX Callable Library d'IBM ILOG CPLEX suite, version 12.5.

Instances Il n'existait pas de générateur d'instances puisqu'à notre connaissance, le MRLRP est un nouveau problème. Nous n'avons pas utilisé des instances du WLRP (Perl et Daskin (1985)) car il n'y avait qu'un petit nombre d'instances de petite taille. Nous avons donc construit nos instances du MRLRP à partir de celles du CLRP (Prins et al. (2006)), que nous avons complétées afin d'y inclure les spé-

cificités du MRLRP. Une brève description de la génération est donnée ci-dessous (pour une description complète, voir Gianessi et al. (en révision)).

Génération des instances Les tests ont été menés dans le but :
a) d'établir une relation entre la difficulté d'une instance et sa taille, notamment la taille de U, K, L, D et P ;
b) d'évaluer les performances de GALW et de ses différents sous-problèmes.
Pour cette raison, nous avons créé une collection d'instances à partir d'une instance du CLRP. Les instances d'une collection (cf a)) sont divisées en cinq *scénarios* : le *scénario de base* ou *scénario 0*, comporte une taille initiale pour U, K, L, D et P, et quatre scénarios dans lesquels chacun des ensembles est modifié : le *scénario 1* contient des demandes additionnelles dans D et P, le *scénario 2* contient des CDU supplémentaires dans U, le *scénario 3* a des portes en plus dans K, et enfin le *scénario 4* contient des PLS en plus dans L. Notons que chaque instance vérifie $|D| = |P|$. Chaque scénario (cf b)) contient un ensemble de quatre *instances* qui diffèrent en fonction des coûts de construction de l'anneau (faibles/élevés) et des coûts de transport (faibles/élevés) : le type L|L correspond à des coûts de construction faibles et des coûts de transport faibles, le type L|H à des coûts faibles/élevés, le type H|L à des coûts élevés/faibles et le type H|H à des coûts élevés/élevés. Le tableau 3.15 montre la correspondance entre les instances du CLRP et les collections du MRLRP, en montrant pour chaque collection les principales caractéristiques de ces scénarios telles que les valeurs minimales et maximales de $|K|$, $|U|$, $|L|$ et $|P| = |D|$, ainsi que les caractéristiques communes de chaque scénario : les valeurs de q, M et le nombre de clusters dans lesquels les clients sont regroupés. Dans chaque scénario, $N = |U|$. Nous rappelons que dans Prins et al. (2006), les instances utilisent les noms suivants *n-m-c*[b], où n est le nombre de clients, m le nombre de dépôts et c le nombre de clusters, alors que le 'b' final indique si la capacité du véhicule est 150 ou 70. Le nombre de clusters et la valeur de q sont ainsi les mêmes que dans les instances initiales.

benchmark	collection	#instances	$\|K\|$	$\|U\|$	$\|L\|$	$\|P\|, \|D\|$	#clusters	q	M
20-5-1	galwc01	5×4	5,10	5,10	5,10	15,20	1	70	50
20-5-2	galwc02	5×4	5,10	5,10	5,10	15,20	2	70	50
20-5-1b	galwc03	5×4	5,10	5,10	5,10	15,20	1	150	65
50-5-1	galwc06	5×4	5,10	5,10	5,10	25,40	1	70	60
50-5-3b	galwc07	5×4	5,10	5,10	5,10	25,40	3	150	65
100-5-1	galwc08	5×4	5,10	5,10	10,15	50,80	1	70	50
100-5-3	galwc10	5×4	5,10	5,10	10,15	50,80	3	70	50
100-10-1	galwc11	5×4	5,10	10,15	10,15	50,80	1	70	50

TABLE 3.15 – *Correspondance entre les instances originales du CLRP et les collections d'instances du MRLRP.*

Comme chacune des 8 collections comprend 5 scénarios avec 4 instances chacune, nous obtenons donc 160 instances du MRLRP.

Tests Les tests ont été effectués sur un Pentium dual core 2.6 Ghz avec 3.76Go de RAM. Nous avons imposé un temps limite de 3600 secondes pour le sous-problème d'affectation de GALW et pour la méthode hybride, alors que pour la méthode exacte, le temps limite était de 3600s pour les petites instances et de 10800s pour les instances de taille moyenne et de grande taille. Nous avons de plus utilisé les valeurs suivantes pour les différents paramètres de GALW :

$\beta = 0.9$, $\tau = 0.4$, $p_q = p_M = p_\omega = 1$, alors que ω varie en fonction de la valeur de n dans l'instance originale du CLRP ($\omega = 10 \cdot \lceil 2 \cdot \log_5 n \rceil - 20$). Les paramètres du sous-problème d'affectation dans GALW sont : $S = 3$, $b = 3$, $\alpha = 2\%$.
Dans les tableaux suivants, nous étudions les résultats en fonction des caractéristiques des instances. On s'intéresse d'abord à une structure de coût et l'on prend toutes les instances avec cette structure dans tous les scénarios de toutes les collections. Ainsi, nous analysons comment la taille d'une instance affecte les performances des trois méthodes, tous les autres paramètres étant égaux. Nous effectuons cette analyse pour le type L|L. Nous prenons ensuite le scénario de base de chaque collection et donnons les résultats pour toutes les instances afin de vérifier le point b). Les tableaux 3.17 et 3.18 donnent ces résultats pour les petites instances, les tableaux 3.20 et 3.21 concernent les collections de taille moyenne (galwc06, galwc07) et les tableaux 3.22 et 3.23 sont pour les instances de grande taille (galwc08, galwc10, galwc11). Le nom de chaque colonne est expliqué dans le tableau 3.16. Les résultats complets peuvent être trouvés dans Gianessi et al. (en révision).

notation	description
Instance	identifiant de l'instance dans le format *collection-scénario-type*
r	temps CPU de génération des routes (s)
t	temps CPU de prétraitement pour calculer les STSP (s ; GALW)
a	temps CPU pour résoudre les sous-problèmes d'affectation (s ; GALW)
T	temps CPU total, sans les entrées/sorties (s)
$\%^r$	gap à la racine (s ; méthode X)
$\%^z$	gap encore à fermer si le temps limite a été atteint
$\%^X$	gap de GALW ou de la méthode Y par rapport à la solution optimale de la méthode X
$\%^Y$	gap de GALW par rapport à la méthode Y
#ps	nombre de variables de chemins dans la formulation (exacte, hybride, sous-problème d'affectation de GALW)

TABLE 3.16 – *Explications sur les noms des colonnes des tableaux de résultats.*

	Caractéristiques							Exacte(X)				GALW							Hybride(Y)	
Instance	$\|K\|$	$\|U\|$	$\|L\|$	$\|D\|$	N	q	M	$\%^r$	r	$T/\%^z$	#ps	$\%^X$	t	r	a	T	#ps	$\%^Y$	$\%^X$	T
galwc01-0-L|L	5	5	5	15	5	70	50	3.5	9.6	24.8	21503	2.1	0.0	0.4	2.9	3.6	3122	0.0	2.1	3.6
galwc01-1-L|L	5	5	5	20	5	70	50	3.3	14.0	48.7	37839	3.3	0.0	0.7	4.2	5.2	4033	0.1	3.2	4.8
galwc01-2-L|L	5	10	5	15	10	70	50	4.5	24.9	642.5	64737	6.2	1.1	0.4	28.5	29.2	8388	2.4	3.9	37.3
galwc01-3-L|L	10	5	5	15	5	70	50	4.7	9.6	16.8	21503	3.8	0.0	0.4	4.0	4.7	3068	2.5	1.4	2.3
galwc01-4-L|L	5	5	10	15	5	70	50	4.4	13.6	31.0	31503	2.5	0.0	0.4	4.2	5.0	4348	0.0	2.5	2.7
average								4.1	14.3	152.8	35417	3.6	0.2	0.5	8.8	9.5	4592	1.0	2.6	10.1
galwc02-0-L|L	5	5	5	15	5	70	50	5.7	3.0	8.0	14066	1.4	0.0	0.3	1.6	2.2	2319	0.0	1.4	1.6
galwc02-1-L|L	5	5	5	20	5	70	50	6.0	9.0	13.3	33616	0.9	0.0	0.6	2.4	3.4	3253	0.1	0.8	1.9
galwc02-2-L|L	5	10	5	15	10	70	50	5.0	7.6	59.7	39438	3.6	1.1	0.3	4.7	5.3	4707	0.1	3.5	7.5
galwc02-3-L|L	10	5	5	15	5	70	50	3.2	3.1	2.8	14066	0.3	0.0	0.3	2.1	2.7	2324	0.1	0.2	1.4
galwc02-4-L|L	5	5	10	15	5	70	50	6.3	5.2	8.0	23190	0.2	0.0	0.3	2.4	3.1	3478	0.0	0.2	2.0
average								5.2	5.6	18.4	24875	1.3	0.2	0.4	2.6	3.3	3216	0.1	1.2	2.9
galwc03-0-L|L	5	5	5	15	5	150	65	18.6	6.6	10.8	18761	2.1	0.0	0.5	1.9	2.7	2082	0.0	2.1	2.2
galwc03-1-L|L	5	5	5	20	5	150	65	16.0	51.3	113.8	68987	3.7	0.0	1.2	2.5	4.0	3513	1.5	2.2	3.8
galwc03-2-L|L	5	10	5	15	10	150	65	19.2	24.1	181.9	79140	0.6	2.0	0.6	26.2	27.1	8449	0.0	0.6	13.7
galwc03-3-L|L	10	5	5	15	5	150	65	18.6	6.7	11.3	18761	2.7	0.0	0.5	1.7	2.5	2085	0.0	2.7	1.8
galwc03-4-L|L	5	5	10	15	5	150	65	18.1	11.0	22.7	31590	1.9	0.0	0.5	2.4	3.2	3319	0.0	1.9	2.7
average								18.1	19.9	68.1	43448	2.2	0.4	0.7	6.9	7.9	3890	0.3	1.9	4.8

TABLE 3.17 – *Résultats pour les petites instances de type L|L.*

Analyse des résultats On peut observer que :

– Sur les instances de *petite taille* du tableau 3.18, où toutes les dimensions sont égales (scénario 0) et où seule la structure des coûts varie, l'heuristique

Instance	Caractéristiques							Exacte(X)				GALW							Hybride(Y)	
	$\|K\|$	$\|U\|$	$\|L\|$	$\|D\|$	N	q	M	$\%^r$	r	$T/\%^z$	#ps	$\%^X$	t	r	a	T	#ps	$\%^Y$	$\%^X$	T
galwc01-0-L|L	5	5	5	15	5	70	50	3.5	9.6	24.8	21503	2.1	0.0	0.4	2.9	3.6	3122	0.0	2.1	3.6
galwc01-0-L|H	5	5	5	15	5	70	50	5.0	9.5	30.8	21503	4.9	0.0	0.4	4.2	4.9	3121	0.6	4.3	4.9
galwc01-0-H|L	5	5	5	15	5	70	50	6.5	9.6	16.3	21503	6.2	0.0	0.4	3.0	3.7	2926	4.6	1.7	2.4
galwc01-0-H|H	5	5	5	15	5	70	50	9.6	9.6	15.4	21503	9.3	0.0	0.4	7.4	8.1	3021	5.8	3.7	2.9
average								6.2	9.6	21.8		5.6	0.0	0.4	4.4	5.1	3048	2.8	3.0	3.5
galwc02-0-L|L	5	5	5	15	5	70	50	5.7	3.0	8.0	14066	1.4	0.0	0.3	1.6	2.2	2319	0.0	1.4	1.6
galwc02-0-L|H	5	5	5	15	5	70	50	10.0	3.0	10.3	14066	5.9	0.0	0.3	2.3	3.0	2418	1.3	4.6	2.0
galwc02-0-H|L	5	5	5	15	5	70	50	5.8	3.0	9.5	14066	1.7	0.0	0.3	2.3	2.9	2352	0.0	1.7	1.9
galwc02-0-H|H	5	5	5	15	5	70	50	7.4	3.0	4.0	14066	4.7	0.0	0.3	2.4	3.1	2421	0.2	4.5	1.7
average								7.2	3.0	8.0		3.4	0.0	0.3	2.1	2.8	2378	0.4	3.0	1.8
galwc03-0-L|L	5	5	5	15	5	150	65	18.6	6.6	10.8	18761	2.1	0.0	0.5	1.9	2.7	2082	0.0	2.1	2.2
galwc03-0-L|H	5	5	5	15	5	150	65	16.6	6.6	23.6	18761	6.3	0.0	0.5	4.9	5.7	2099	1.5	4.8	2.5
galwc03-0-H|L	5	5	5	15	5	150	65	26.5	6.6	20.2	18761	6.6	0.0	0.5	1.9	2.7	2012	1.2	5.4	1.8
galwc03-0-H|H	5	5	5	15	5	150	65	25.3	6.6	23.5	18761	6.9	0.0	0.5	5.0	5.8	2129	2.4	4.6	2.2
average								21.8	6.6	19.5		5.5	0.0	0.5	3.4	4.2	2080	1.3	4.2	2.2

TABLE 3.18 – *Résultats sur les instances de petites tailles de tous types sur le scénario 0.*

GALW a un gap ($\%^X$), par rapport à la solution optimale trouvée par la méthode exacte, compris entre 2% et 7%. Ces gaps sont obtenus dans des temps de calcul inférieurs (4 fois plus rapide en moyenne que la méthode exacte). Cela est dû principalement au nombre de routes générées par GALW qui ne représente que 11% à 17% du nombre de routes générées par X. De plus, ces gaps peuvent s'expliquer par la phase de génération de routes puisque le gap de GALW par rapport à $Y(\%^X)$ est inférieur à 1% dans la plupart des cas. On peut aussi constater que H|H est la structure de coût la plus difficile à résoudre avec un gap par rapport à l'optimum qui peut être multiplié par un facteur 4 par rapport aux autres coûts. Si l'on s'intéresse désormais à la taille des instances (voir le tableau 3.17), on remarque que le nombre $|U|$ de sites potentiels de CDU semble avoir un impact significatif sur la difficulté de résolution du problème : dans les instances du scénario 2 (i.e. celles dans lesquelles seul $|U|$ a été augmenté), le gap de GALW par rapport à l'optimum ou le temps de résolution de GALW est plus important comparé aux autres scénarios. Les instances du scénario 2 ont également le plus important temps de résolution pour la méthode exacte X (de 5 à 20 fois plus long). Cela peut notamment s'expliquer par le fait que pour X et GALW, le nombre de routes générées dépend directement du nombre de CDU. C'est aussi le cas pour le nombre de demandes $|D| = |P|$, qui augmente le nombre de routes générées. Il est important de remarquer que le nombre de portes a un faible impact puisqu'il ne concerne que le premier niveau. On peut aussi remarquer que le nombre de PLS a aussi un impact sur le nombre de chemins mais pas sur les performances des méthodes. En effet, les PLS ne concernent que le routage et les contraintes d'équilibrage de flotte mais ni les capacités des différentes ressources ni l'occupation en marchandises des CDU par exemple. Enfin, on peut constater que pour ces instances de petite taille, la méthode hybride est plus performante en moyenne que GALW puisqu'elle obtient de plus faibles gaps en moins de temps.

– Sur les instances de *taille moyenne* des tableaux 3.20 et 3.21, nous n'avons pas reporté les résultats de la méthode exacte X. En effet, une analyse de résultats préliminaires illustrés dans le tableau 3.19 suggère que CPLEX peut de temps en temps terminer sans trouver de solution réalisable pour X. Cette analyse a été effectuée sur un petit échantillon d'instances de taille moyenne et de grande taille avec les scénarios 2 et 3 (sauf pour galwc06-1-L|L). Seules les instances de type L|L ont été considérées pour simplifier les

calculs et nous souhaitions tester la méthode X quand $|U|$ augmente. Nous ne pouvons calculer la valeur de X que pour une poignée d'instances. Cela est dû au très grand nombre de routes générées par la méthode exacte (centaines de milliers de routes pour les instances de taille moyenne et quelques millions pour les instances de grande taille). Dans certains cas (i.e. ceux avec *NS* dans le tableau), CPLEX n'arrive même pas à trouver de solution à la relaxation continue dans le temps imparti. On peut aussi observer que le gap de GALW par rapport à X reste acceptable pour ces grandes instances quand il a pu être calculé, puisque compris entre 1% et 5%, sauf pour une instance. On peut donc espérer un gap du même ordre pour les instances de grande taille quand X ne fournit pas de solution. Pour les instances de taille moyenne (tableaux 3.20 et 3.21), nous ne comparons donc que GALW et la méthode hybride. Nous observons que la méthode hybride Y demeure compétitive et le gap entre GALW et Y est encore inférieur à 3% pour la plupart des instances en des temps de calcul comparables. On peut encore observer dans le tableau 3.20 que les dimensions critiques sont le nombre de CDU et le nombre de demandes. Le nombre de routes générées par GALW est deux à 3 fois supérieur à celui des petites instances mais les temps de calcul restent faibles.

– Sur les instances de *grande taille* des tableaux 3.22 et 3.23, nous atteignons les limites de la méthode hybride Y. Dans la plupart des cas GALW trouve une solution avec un gap de 1% à 3% par rapport à la méthode hybride en un temps comparable voire bien inférieur. Pour les plus grandes instances, GALW trouve des solutions meilleures que Y qui atteint son temps limite, voire qui n'arrive pas à trouver de solution réalisable. Les temps de calcul de GALW sont également plus importants pour les grandes instances puisque le nombre de routes générées peut dépasser 30000 dans certains cas.

Cette analyse permet de conclure que GALW est une méthode efficace pour trouver des solutions de bonne qualité en un temps raisonnable. Le nombre de routes générées pourrait être majoré pour contrôler le temps de calcul pour les plus grandes instances. La figure 3.12 montre l'évolution du nombre de routes en fonction de M dans la formulation générale, pour deux instances de taille moyenne. Plus la limite sur la longueur des routes est élevée, plus le nombre de façons de combiner les clients dans une route s'élève également, tout comme le nombre de chemins.

	Caractéristiques							Exacte(X)				GALW						Hybride(Y)	
Instance	$\|K\|$	$\|U\|$	$\|L\|$	$\|D\|$	N	q	M	$\%^r$	r	T	#ps	$\%X$	t	r	a	T	#ps	$\%Y$	$\%X$	T
galwc06-1-LlL	5	5	5	40	5	70	60	0,1	75,8	252,5	219879	3,8	0,0	4,0	5,7	10,0	7376	2,0	1,8	8,3
galwc06-2-LlL	5	10	5	25	10	70	60	6,7	21,4	216,4	100049	3,8	1,1	1,4	11,0	12,7	9428	1,4	2,4	33,2
galwc07-0-LlL	5	5	5	25	5	150	65	*NS*	434,6	*NS*	457812	$-\infty$	0,0	2,6	6,0	8,9	4791	0,7	$-\infty$	7,4
galwc07-2-LlL	5	10	5	25	10	150	65	*NS*	1833,4	*NS*	1566535	$-\infty$	1,2	2,6	22,7	25,7	13286	3,0	$-\infty$	31,1
galwc08-0-LlL	5	5	10	50	5	70	50	2,2	30,3	430,1	111289	4,9	0,0	9,2	11,2	20,7	6350	0,5	4,4	20,1
galwc08-2-LlL	5	10	10	50	10	70	50	2,7	102,3	2864,2	387820	3,4	1,6	7,3	56,5	64,2	15019	0,0	3,4	69,8
galwc10-0-LlL	5	5	10	50	5	70	50	2,1	163,5	4979,7	352647	1,7	0,0	7,7	17,1	25,2	10814	0,0	1,7	18,0
galwc10-2-LlL	5	10	10	50	10	70	50	*NS*	8537,6	*NS*	885801	$-\infty$	1,4	6,8	242,9	250,1	26212	0,2	$-\infty$	281,1
galwc11-0-LlL	5	10	10	50	10	70	50	2,9	83,5	6837,2	295662	10,4	1,4	7,9	466,4	474,8	14835	2,0	8,5	2681,4
galwc11-2-LlL	5	15	10	50	15	70	50	*NS*	208,6	*NS*	535059	$-\infty$	69,7	8,1	612,8	621,3	27451	-18,6	$-\infty$	(28,9%)

TABLE 3.19 – *Résultats sur un petit échantillon d'instances de taille moyenne et de grande taille.*

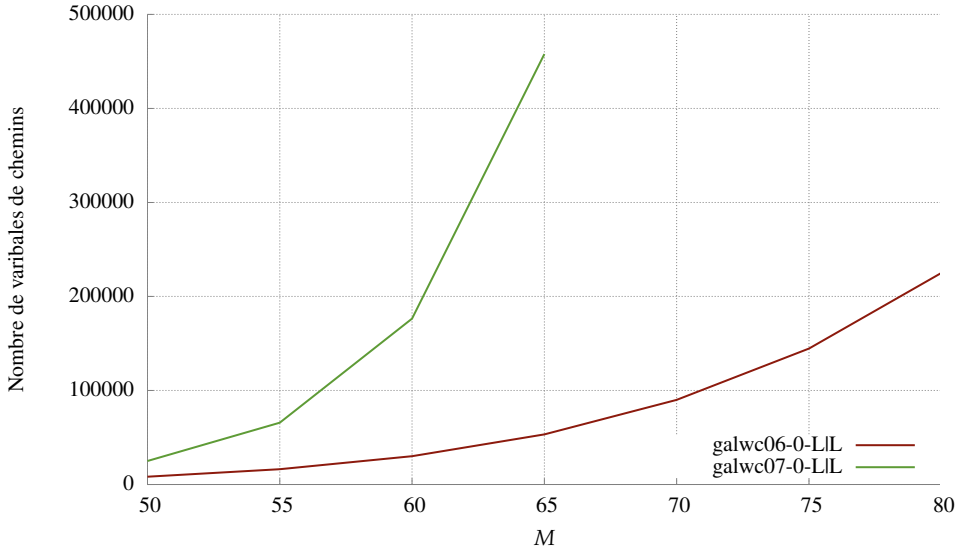


FIGURE 3.12 – *Relation entre le nombre de routes et la longueur maximale m d'un trajet.*

Instance	Caractéristiques $\|K\|$	$\|U\|$	$\|L\|$	$\|D\|$	N	q	M	GALW t	r	a	T	#ps	$\%Y$	Hybride(Y) T
galwc06-0-L|L	5	5	5	25	5	70	60	0.0	1.7	2.0	4.0	3180	3.0	3.9
galwc06-1-L|L	5	5	5	40	5	70	60	0.0	4.0	5.7	10.0	7376	2.0	8.3
galwc06-2-L|L	5	10	5	25	10	70	60	1.1	1.4	11.0	12.7	9428	1.4	33.2
galwc06-3-L|L	10	5	5	25	5	70	60	0.0	1.7	2.4	4.4	3155	1.1	4.1
galwc06-4-L|L	5	5	10	25	5	70	60	0.0	1.5	2.8	4.7	4448	0.6	4.2
average								0.2	2.1	4.8	7.2	5517	1.6	10.7
galwc07-0-L|L	5	5	5	25	5	150	65	0.0	2.6	6.0	8.9	4791	0.7	7.4
galwc07-1-L|L	5	5	5	40	5	150	65	0.0	7.7	18.2	26.2	10902	0.3	23.1
galwc07-2-L|L	5	10	5	25	10	150	65	1.2	2.6	22.7	25.7	13286	3.0	31.1
galwc07-3-L|L	10	5	5	25	5	150	65	0.0	2.6	5.8	8.7	4752	0.0	8.9
galwc07-4-L|L	5	5	10	25	5	150	65	0.0	2.6	7.7	10.6	6936	0.0	9.0
average								0.2	3.6	12.1	16.0	8133	0.8	15.9

TABLE 3.20 – *Résultats de GALW et de la méthode hybride sur les instances de taille moyenne de type L|L.*

Instance	Caractéristiques $\|K\|$	$\|U\|$	$\|L\|$	$\|D\|$	N	q	M	GALW t	r	a	T	#ps	$\%Y$	Hybride(Y) T
galwc06-0-L|L	5	5	5	25	5	70	60	0.0	1.7	2.0	4.0	3180	3.0	3.9
galwc06-0-L|H	5	5	5	25	5	70	60	0.0	1.6	2.4	4.3	3107	8.7	5.5
galwc06-0-H|L	5	5	5	25	5	70	60	0.0	1.7	3.7	5.7	3135	0.1	4.0
galwc06-0-H|H	5	5	5	25	5	70	60	0.0	1.6	3.5	5.5	3092	2.5	5.5
average								0.0	1.6	2.9	4.9	3128	3.6	4.7
galwc07-0-L|L	5	5	5	25	5	150	65	0.0	2.6	6.0	8.9	4791	0.7	7.4
galwc07-0-L|H	5	5	5	25	5	150	65	0.0	2.7	7.3	10.3	4880	0.1	7.1
galwc07-0-H|L	5	5	5	25	5	150	65	0.0	2.6	8.1	11.0	4813	0.2	7.8
galwc07-0-H|H	5	5	5	25	5	150	65	0.0	2.6	8.0	10.9	4839	0.1	9.6
average								0.0	2.6	7.3	10.3	4831	0.3	8.0

TABLE 3.21 – *Résultats de GALW et de la méthode hybride sur toutes les instances de taille moyenne du scénario 0.*

Conclusion

Nous avons considéré ici un problème stratégique de localisation, le "Multicommodity-Ring Location Routing Problem (MRLRP)". L'objectif de ce problème est de considérer le problème du dernier kilomètre et de l'étudier d'un point de vue stratégique avec un système de distribution fondé sur des Centres de Distribution Urbaine pour les opérations de cross-docking. Le MRLRP est un problème riche et complexe puisqu'il comprend un grand nombre de décisions différentes et interconnectées (localisation, conception de réseau, routage, trans-

	Caractéristiques							GALW						Hybride(Y)
Instance	$\|K\|$	$\|U\|$	$\|L\|$	$\|D\|$	N	q	M	t	r	a	T	#ps	$\%^Y$	$T/\%^Z$
galwc08-0-L|L	5	5	10	50	5	70	50	0.0	9.2	11.2	20.7	6350	0.5	20.1
galwc08-1-L|L	5	5	10	80	5	70	50	0.0	28.1	31.1	59.7	16593	0.7	271.0
galwc08-2-L|L	5	10	10	50	10	70	50	1.6	7.3	56.5	64.2	15019	0.0	69.8
galwc08-3-L|L	10	5	10	50	5	70	50	0.0	9.6	11.6	21.6	6412	0.1	16.9
galwc08-4-L|L	5	5	15	50	5	70	50	0.0	9.0	13.2	22.7	9334	0.5	21.0
average								0.3	12.6	24.7	37.8	10742	0.4	79.8
galwc10-0-L|L	5	5	10	50	5	70	50	0.0	7.7	17.1	25.2	10814	0.0	18.0
galwc10-1-L|L	5	5	10	80	5	70	50	0.0	24.0	106.7	131.2	28020	0.6	768.3
galwc10-2-L|L	5	10	10	50	10	70	50	1.4	6.8	242.9	250.1	26212	0.2	281.1
galwc10-3-L|L	10	5	10	50	5	70	50	0.0	8.2	31.0	39.7	10912	0.2	55.7
galwc10-4-L|L	5	5	15	50	5	70	50	0.0	7.8	19.5	27.7	13963	0.1	23.8
average								0.3	10.9	83.4	94.8	17984	0.2	229.4
galwc11-0-L|L	5	10	10	50	10	70	50	1.4	7.9	466.4	474.8	14835	2.0	2681.4
galwc11-1-L|L	5	10	10	80	10	70	50	1.1	21.6	476.7	499.0	36996	$-\infty$	$(+\infty)$
galwc11-2-L|L	5	15	10	50	15	70	50	69.7	8.1	612.8	621.3	27451	-18.6	(28.9%)
galwc11-3-L|L	10	10	10	50	10	70	50	1.1	8.3	834.0	842.8	14691	-0.1	(2.0%)
galwc11-4-L|L	5	10	15	50	10	70	50	1.2	7.6	391.6	399.7	19571	5.8	879.2
average								14.9	10.7	556.3	567.5	22709	-2.7	2690.2

TABLE 3.22 – *Résultats de GALW et de la méthode hybride sur les instances de taille moyenne et de grande taille de type L\|L.*

	Features							GALW						Hybrid(Y)
Instance	$\|K\|$	$\|U\|$	$\|L\|$	$\|D\|$	N	q	M	t	r	a	T	#ps	$\%^Y$	$T/\%^Z$
galwc08-0-L|L	5	5	10	50	5	70	50	0.0	9.2	11.2	20.7	6350	0.5	20.1
galwc08-0-L|H	5	5	10	50	5	70	50	0.0	9.4	6.3	16.1	6424	1.3	19.6
galwc08-0-H|L	5	5	10	50	5	70	50	0.0	9.7	8.9	19.0	6386	6.1	15.2
galwc08-0-H|H	5	5	10	50	5	70	50	0.0	9.5	8.1	18.0	6373	1.8	20.8
average								0.0	9.4	8.6	18.4	6383	2.4	18.9
galwc10-0-L|L	5	5	10	50	5	70	50	0.0	7.7	17.1	25.2	10814	0.0	18.0
galwc10-0-L|H	5	5	10	50	5	70	50	0.0	7.8	16.0	24.1	10647	3.2	23.3
galwc10-0-H|L	5	5	10	50	5	70	50	0.0	7.7	26.5	34.6	10820	0.5	42.2
galwc10-0-H|H	5	5	10	50	5	70	50	0.0	7.9	15.2	23.5	11015	0.9	39.0
average								0.0	7.8	18.7	26.9	10824	1.2	30.6
galwc11-0-L|L	5	10	10	50	10	70	50	1.4	7.9	466.4	474.8	14835	2.0	2681.4
galwc11-0-L|H	5	10	10	50	10	70	50	1.8	8.0	514.2	522.7	14663	3.1	(0.6%)
galwc11-0-H|L	5	10	10	50	10	70	50	1.1	8.1	414.7	423.1	14671	2.6	3211.2
galwc11-0-H|H	5	10	10	50	10	70	50	1.1	8.0	548.9	557.3	14532	2.6	1628.3
average								1.4	8.0	486.1	494.5	14675	2.6	2780.2

TABLE 3.23 – *Résultats de GALW et de la méthode hybride sur toutes les instances de taille moyenne et de grande taille du scénario 0.*

port de flots). Nous avons proposé une formulation mixte de type partitionnement qui ne peut être résolue à l'optimum que pour des instances de petite taille. Au lieu d'utiliser une énumération implicite basée sur la génération de colonnes, nous avons ici développé une matheuritique GALW fondée sur une décomposition du problème en 4 étapes. GALW prend en compte les différentes décisions du MRLRP séquentiellement et résout 3 d'entre elles de façon optimale. Nous avons aussi proposé une approche hybride qui consiste à appliquer la formulation exacte sur un sous-ensemble de routes générées par GALW. 160 instances du MRLRP, avec différentes caractéristiques, ont été conçues afin de mener des expérimentations. Sur les instances de petite taille, GALW et la méthode hybride ont fourni des solutions avec de faibles gaps en des temps de calcul faibles. L'approche hybride est légèrement meilleure que GALW sur les instances de taille moyenne pour lesquelles la méthode exacte commence à échouer à trouver des solutions réalisables. Sur les grandes instances, GALW offre clairement le meilleur compromis entre la qualité de la solution et le temps de calcul.

Nous avons également appliqué ces approches sur des instances "écologiques" pour montrer l'intérêt de ce type de système de transport en milieu urbain et nous développons actuellement un algorithme de branch and price dans le cas où l'anneau est fixé.

3.1.6 Réoptimisation locale

Cette section introduit le concept de Réoptimisation Locale ((LR) : Local Reoptimization) afin de gérer l'incertitude sur les données. Nous souhaitons obtenir une solution qui soit la plus proche possible d'une solution de référence qui n'est plus réalisable suite à une modification partielle des données. LR est conçue pour des formulations génériques du type partitionnement d'ensembles souvent rencontrées dans des schémas de génération de colonnes. Nous appliquerons LR sur les problèmes classiques de Bin Packing et P-Médian afin de démontrer l'efficacité de cette approche.

Introduction

En pratique, il est courant que les données changent dynamiquement. Deux types d'approches existent afin de prendre en compte cette dynamicité et l'incertitude sur les données qui en résulte : les approches *proactives* et *réactives*. Les premières intègrent le caractère stochastique des évènements afin de proposer des solutions robustes (Kouvelis et Yu (1997), Ben-Tal et al. (2009), Birge et Louveaux (2011)). L'approche *réactive* cherche à obtenir de nouvelles solutions prenant en compte ce changement, soit en repartant de zéro, soit en exploitant une connaissance a priori (sur la solution précédente par exemple). Un objectif naturel dans ce dernier cas, consiste à trouver une solution, pour la nouvelle instance, qui soit la plus proche possible (selon une certaine mesure de distance) de l'originale.

Il existe de nombreux problèmes d'optimisation avec des données incertaines, on peut citer par exemple la planification des trains et le redéploiement des véhicules d'urgence (Bélanger et al. (2010c),Bélanger et al. (2010b), Bélanger et al. (2010a), Bélanger et al. (2011)). Bien d'autres problèmes réels intègrent cet aspect de robustesse et/ou de réoptimisation, comme par exemple les challenges EURO/ROADEF proposés par Amadeus, EDF et Google.

La réoptimisation intègre deux principaux challenges : (i) calculer une solution optimale (ou proche de l'optimal) pour la nouvelle instance et (ii) convertir efficacement la solution courante en une nouvelle.

Notre approche diffère des précédents travaux sur deux aspects. Le premier est que notre mesure de performance combine deux fonctions objectifs, contrairement à la majorité des autres travaux qui se concentrent sur la complexité de résolution induite par les modifications. L'autre aspect est la généricité de notre approche puisque tout problème pouvant être résolu par une décomposition de Danzig-Wolfe impliquant un problème maître de type partitionnement ou couverture peut être considéré.

Contribution Notre première contribution consiste en une description mathématique du concept de *Réoptimisation Locale* LR. Partant d'une solution de référence, nous souhaitons obtenir une fonction composée de deux objectifs, la fonction de coût originale plus un second terme prenant en compte la proximité entre la nouvelle solution et la solution de référence.

La caractéristique commune consiste à rendre l'objectif quadratique en ajoutant un nouveau terme quadratique qui prend en compte la similarité entre les solutions.

La seconde contribution est, à notre connaissance, le fait qu'il s'agit de la première génération de colonnes pour un problème quadratique où le terme qua-

dratique de l'objectif est mis dans le sous-problème.

Les principales questions sont donc :
- étant donné un problème de partitionnement d'ensembles résolu par génération de colonnes, comment déterminer les coûts de transition afin de minimiser une distance locale ?
- étant donnée une solution de référence, comment ajouter un problème de partition pondéré au problème original afin de construire une solution proche de la solution de référence ?
- étant donné un schéma algorithmique, comment ajouter un terme quadratique sans changer la structure algorithmique ?

État de l'art

Le concept de réoptimisation a souvent été mentionné dans le contexte de la post-optimalité (Schaffter (1997), Thiongane et al. (2005)) qui consiste à déterminer à quel point une instance peut être modifiée sans changer ses solutions optimales (van Hoesel et Wagelmans (1999)).

Le problème le plus étudié dans le cadre de la réoptimisation est le plus court chemin (Radionov (1968), Murchland (1967), Dionne (1978), Pape (1974)). Puisque ce problème est polynomial, l'objectif consiste ici à trouver une solution optimale très efficacement en se basant sur la nature locale des modifications et sur les propriétés des solutions optimales. Une autre direction consiste en la recherche d'une bonne solution, pour un problème NP-difficile, étant donné une solution optimale pour une instance proche. On peut citer les travaux sur le TSP (Traveling Salesman Problem) (Archetti et al. (2003)), sur le problème de l'arbre de Steiner (Bockenhauer et al. (2009)) et sur le sac à dos linéaire et quadratique en 0-1 (Archetti et al. (2010), Létocart et al. (2012), Thiongane et al. (2005; 2006)).

Réoptimisation Locale et Distance Locale

Nous définissons ici le concept de Réoptimisation Locale (LR) appliqué aux problèmes de type partitionnement d'ensembles. Nous commençons par définir le concept de Proximité Locale (Local Adjacency (LA)) qui représente la mesure de distance entre la solution de référence et la solution réoptimisée. La proximité correspond à la somme des prix obtenus en réunissant des objets dans la solution réoptimisée, les prix étant définis en fonction de la solution de référence.

Soit I l'ensemble des m objets à partitionner avec leurs caractéristiques σ_i ($i = \{1,2,\ldots,m\}$), qui peuvent être des poids par exemple. Soit S l'ensemble des n sous-ensembles réalisables de I définis par $S = \{S_1, S_2, \ldots, S_n\}$; un sous-ensemble est réalisable s'il respecte l'ensemble de contraintes C ($Ay \leq b$ avec A la matrice de coefficients, b le vecteur des membres droits et y le vecteur de variables), qui peuvent par exemple être des contraintes de capacité. La partition P est une partition des objets de I telle que $P \subseteq S$, $\cup_{j \in P} S_j = I$ et $S_j \cap S_k = \emptyset \quad \forall j,k \in P, j \neq k$. De plus, la fonction de coût $f(S_j)$ est associée à chaque sous-ensemble réalisable S_j ($j = \{1,2,\ldots,n\}$). Le problème de partitionnement consiste alors à trouver la partition optimale $P^* \subseteq S$ de coût minimum :

$$z = \sum_{S_j \in P^*} f(S_j).$$

Soit Ψ une configuration initiale de I, S et C, et la solution de référence P^*. Si l'on considère une nouvelle configuration $\tilde{\Psi}$, i.e., de nouveaux ensembles $\tilde{I}$, $\tilde{S}$ et $\tilde{C}$ ainsi qu'une matrice Q de prix $q_{ii'}$ avec $i = \{1,2,\ldots,m\}$ et $i' = \{1,2,\ldots,m\}$, nous pouvons définir la proximité locale (LA) comme la fonction suivante d'une solution $\tilde{P}$ d'une nouvelle configuration $\tilde{\Psi}$:

$$LA(\tilde{P}) = \sum_{\tilde{S}_j \in \tilde{P}} \sum_{i \in \tilde{S}_j} \sum_{i' \in \tilde{S}_j} q_{ii'}.$$

Les prix $q_{ii'}$ peuvent prendre des valeurs génériques, modélisant un prix (ou une pénalité) associé au fait de mettre deux objets ensemble dans le même sous-ensemble de la nouvelle solution $\tilde{P}$. On se restreint ici au cas où des prix ne sont donnés qu'à des paires d'objets qui sont dans le même sous-ensemble dans la solution de référence P^*. Nous définissons $q_{ii'}$ en fonction de la structure de la solution de référence P^*, en prenant en compte les sous-ensembles $S_j \in P^*$. Si les objets i et i' appartiennent au même sous-ensemble S_j de la solution de référence P^*, nous définissons $q_{ii'}$ de la manière suivante :

$$q_{ii'} = \frac{2V}{(|S_j|(|S_j|-1))} \qquad i,i' = 1,\ldots m,\ i \in S_j,\ i' \in S_j,$$

avec $|S_j|$ la cardinalité du sous-ensemble S_j et V est une valeur positive entière. Ainsi si la solution réoptimisée est identique à la solution de référence, LA est maximale.

Le Problème de Réoptimisation Locale (LRP) consiste alors à trouver une nouvelle partition $\tilde{P}^*$ de coût optimal qui comprend les coûts initiaux et ceux de la proximité locale :

$$\tilde{z} = \sum_{\tilde{S}_j \in \tilde{P}^*} f(\tilde{S}_j) - LA(\tilde{P}^*).$$

Formulation mathématique

Le sous-ensemble $\tilde{S}_j$ peut être vu comme une colonne à n entrées, chaque entrée i ($i = 1,\ldots,n$) étant égale à 1 si l'élément est présent et 0 sinon. Chaque sous-ensemble réalisable $\tilde{S}_j$, qui respecte les contraintes $\tilde{C}$, est défini en utilisant les variables y :

$$\tilde{S} = \left\{\tilde{S}_j = (y_1, y_2, \ldots, y_n) : Ay \leq b,\ y_i \in \{0,1\}\ (i = 1,2,\ldots,n)\right\}$$

avec A la matrice des coefficients et b le vecteur des membres droits.

Nous introduisons les variables binaires $x_{\tilde{S}_j}$ ($\tilde{S}_j \in \tilde{S}$) qui sont égales à 1 si le sous-ensemble $\tilde{S}_j$ est dans la solution et 0 sinon. Enfin, chaque objet doit être couvert par un et un seul sous-ensemble. La formulation de LRP est ainsi :

$$(LRP) \quad \min \sum_{\tilde{S}_j \in \tilde{S}} C_{\tilde{S}_j} x_{\tilde{S}_j} \tag{3.81}$$

$$\sum_{\tilde{S}_j \in \tilde{S}(i)} x_{\tilde{S}_j} = 1 \qquad i = 1,\ldots,n \tag{3.82}$$

$$x_{\tilde{S}_j} \in \{0,1\} \qquad \tilde{S}_j \in \tilde{S} \tag{3.83}$$

où les coûts associés à un sous-ensemble réalisable, $\tilde{S}_j \in \tilde{S}$, peuvent être obtenus de la manière suivante :

$$C_{\tilde{S}_j} = f(\tilde{S}_j) - \sum_{i \in \tilde{S}_j} \sum_{i' \in \tilde{S}_j} q_{ii'}.$$

La fonction de coût $f(\tilde{S}_j)$ dépend normalement de la quantité d'objets couverts. Soit $y_i^{\tilde{S}_j}$ la i^{eme} composante de $\tilde{S}_j$, le coût $f(\tilde{S}_j)$ peut être réécrit en utilisant l'objectif linéaire en $(v_i, i = 1 \dots m)$ comme :

$$f(\tilde{S}_j) = \sum_{i=1}^{m} v_i y_i^{\tilde{S}_j}$$

Cette formulation peut avoir un nombre polynomial ou exponentiel de variables $x_{\tilde{S}_j}$ en fonction de l'ensemble de contraintes $\tilde{C}$. Nous nous concentrons sur le cas exponentiel lorsque le LRP est résolu par génération de colonnes.

En relâchant les contraintes d'intégrité (3.83), nous obtenons la relaxation continue de (3.81)-(3.83), définie par (3.81), (3.82) et :

$$x_{\tilde{S}_j} \geq 0, \quad (\tilde{S}_j \in \tilde{S}) \tag{3.84}$$

La formulation ((3.81),(3.82) et (3.84)), appelée Problème Maître (*Master Problem* (MP)), est résolue par génération de colonnes. Nous considérons le problème maître restreint (*Restricted Master Problem* (RMP)) qui contient un sous-ensemble des variables $x_{\tilde{S}_j}$, $\tilde{S}_j \in \tilde{S}$. Étant donnée une solution de RMP, de nouvelles variables, nécessaires pour résoudre optimalement MP, sont obtenues en séparant les contraintes duales suivantes :

$$\sum_{i=1}^{n} \pi_i^* \leq \tilde{C}_{\tilde{S}_j} \quad \tilde{S}_j \in \tilde{S} \tag{3.85}$$

où π_i^* $(i = 1, ..., n)$ représente le vecteur des variables duales optimales associées aux contraintes du type (3.82). Le sous-problème consiste alors à déterminer un sous-ensemble réalisable $\tilde{S}_j^* \in \tilde{S}$ tel que :

$$\sum_{i \in \tilde{S}_j^*} \pi_i^* - \tilde{C}_{\tilde{S}_j^*} > 0 \tag{3.86}$$

Le sous-problème consiste à générer des sous-ensembles réalisables satisfaisant les contraintes $\tilde{C}$. Il s'agit donc d'un problème quadratique en 0-1 qui dépend des variables duales π_i^* $(i = 1, \dots, n)$ et des prix (i.e., Q). Nous utilisons les variables binaires de type y_i qui sont égales à 1 si l'objet appartient au sous-ensemble et 0 sinon :

$$(SousProbleme) \quad \max \sum_{i=1}^{m} \pi_i^* y_i + \sum_{i=1}^{m} v_i y_i - \sum_{i=1}^{m} \sum_{i'=1}^{m} q_{ii'} y_i y_{i'} \tag{3.87}$$

$$Ay \leq b \tag{3.88}$$

$$y_i \in \{0, 1\} \qquad i = 1, \dots, n. \tag{3.89}$$

Si un sous-ensemble réalisable $\tilde{S}_j^*$ avec un profit maximal positif est trouvé, la colonne correspondant à $\tilde{S}_j^*$ est ajoutée au RMP courant ; sinon la valeur de la solution optimale du RMP est une borne inférieure pour le LRP.

Propriétés de la proximité locale

Le problème de réoptimisation peut être formulé par un problème de partitionnement pondéré maximal d'un graphe :

$$(WGPP) \quad \max \sum_{k=1}^{K} \sum_{j=1}^{m} \sum_{i=1}^{m} q_{ij} y_i^k y_j^k \tag{3.90}$$

$$\text{s.t.} \sum_{k=1}^{K} y_i^k = 1 \tag{3.91}$$

$$y_i^k \in \{0,1\} \qquad i = 1,\ldots,m;\ k = 1,\ldots,K \tag{3.92}$$

qui est ajouté à la formulation originale du problème à résoudre.

Si la solution optimale de la formulation ((3.90)-(3.92)) coïncide avec la solution de référence P^*, la partie associée au coût de transition dans l'objectif est nulle.

Proposition 1. *Quand la solution de référence est optimale et que rien ne change dans les données, la solution optimale du problème réoptimisé est encore la solution de référence.*

Preuve 15. *Soient deux ensembles S_j et S_k avec $c = |S_j|$ et $c' = |S_k|$ les cardinalités de ces ensembles d'objets. Soient $a = |S_j'|$, $b = |S_j''|$, $a' = |S_k'|$ et $b' = |S_k''|$ quatre ensembles tels que $c = a + b$ et $c' = a' + b'$. Les valeurs des fonctions objectif quand tous les objets des ensembles d'origine sont regroupés dans le même nouvel ensemble sont $\gamma = q_{ii'}\frac{c(c-1)}{2} = V$ et $\gamma' = q_{ii'}\frac{c'(c'-1)}{2} = V$. S'ils sont divisés en deux sous-ensembles, ils deviennent $\alpha = q_{ii'}\frac{a(a-1)}{2}$, $\beta = q_{ii'}\frac{b(b-1)}{2}$, $\alpha' = q_{ii'}\frac{a'(a'-1)}{2}$ et $\beta' = q_{ii'}\frac{b'(b'-1)}{2}$. Il est évident que $\gamma > \alpha + \beta$ et $\gamma = \alpha + \beta + (ab)q_{ii'}$, puisque lorsque tous les objets qui étaient regroupés dans l'ensemble d'origine S_j sont divisés en deux sous-ensembles, alors les arêtes ab sont manquantes. Nous avons la même chose pour α' et β' lorsque les arêtes $a'b'$ sont manquantes. Il est alors facile de prouver que quelle que soit la répartition des objets des ensembles de référence S_j et S_k, la somme dans la fonction objectif est toujours : $\alpha + \alpha' + \beta + \beta' < 2\gamma = 2V$.*

Corollaire 2. *Supprimer un ensemble dans la nouvelle solution optimale n'est possible que si et seulement si deux ensembles de la solution de référence peuvent être fusionnés ou si les objets d'un ensemble peuvent être répartis dans deux ou plus nouveaux ensembles tels qu'il existe au moins une arête de valeur $q_{ii'} > 0$ qui est utilisée dans au moins un de ces nouveaux ensembles.*

Les données d'origine peuvent être modifiées de différentes manières. La plus classique consiste à insérer ou à supprimer de nouveaux objets dans l'ensemble I. Une autre consiste à modifier leurs caractéristiques comme, par exemple, la valeur des coefficients de la matrice A et/ou la valeur du membre droit b.

Applications

Afin de démontrer la validité de notre approche, nous l'avons appliquée à deux problèmes connus : le Bin packing Problem et le Capacitated p-Median Problem. Nous souhaitons montrer que la structure de l'approche algorithmique ne change pas quand le terme quadratique est ajouté. Dans les deux cas, le sous-problème est un problème de sac à dos, qui devient ainsi un problème de sac à dos quadratique alors que le reste de l'approche reste identique (branchement, problème maître, etc.).

Bin Packing Problem Le problème de Bin Packing (BPP) (Valério de Carvalho (1999; 2002)) peut se formuler de la manière suivante : étant donnés un nombre de boîtes m de capacité W et de coût K, et une liste de n objets de taille $w_1, \ldots, w_n$ $(0 \leq w_i \leq W)$, le problème consiste à affecter les objets aux boîtes en respectant la capacité des boîtes et en minimisant le nombre de boîtes utilisées. Dans le contexte de la réoptimisation, nous avons une matrice symétrique Q définissant un prix $q_{ii'} \geq 0$ pour affecter les objets i et i' à la même boîte que dans la solution de référence $\mathcal{S}$ (potentiellement partielle ou non réalisable). L'objectif devient désormais de minimiser le coût des boîtes utilisées moins le prix associé au fait de reproduire en partie la solution $\mathcal{S}$. Une application de ce problème est par exemple la réaffectation de processus à des processeurs en partant d'une solution donnée.

Il existe de nombreuses formulations pour le BPP dans la littérature. La plus classique est une formulation compacte avec un nombre polynomial de variables et de contraintes, mais elle comporte de nombreuses symétries et la relaxation continue est généralement de mauvaise qualité. Nous pouvons néanmoins facilement l'adapter au cas non linéaire induit par le contexte de la réoptimisation. Il existe également une formulation avec un nombre exponentiel de variables associées avec les sous-ensembles réalisables d'objets (i.e. respectant la capacité W). Elle fournit une meilleure relaxation continue et ne contient que peu de symétries mais nécessite l'élaboration d'un Branch and Price pour la résoudre à l'optimum.

Formulations mathématiques Soient α_{ji}, $\forall j \, \forall i$, les variables binaires telles que α_{ji} vaut 1 si l'objet i est affecté à la boîte j et 0 sinon et soient $\beta_j, \forall j$ les variables binaires telles que β_j vaut 1 si la boîte j est utilisée et 0 sinon. La formulation compacte peut alors s'écrire :

$$\min \sum_{j=1}^{m} \left(\tilde{C}\beta_j - \sum_{i=1}^{n} \sum_{i'=1}^{n} q_{ii'} \alpha_{ji} \alpha_{ji'} \right) \tag{3.93}$$

$$\sum_{i=1}^{n} w_i \alpha_{ji} \leq W \beta_j \qquad j = 1, \ldots, m \tag{3.94}$$

$$\sum_{j=1}^{m} \alpha_{ji} = 1 \qquad i = 1, \ldots, n \tag{3.95}$$

$$\alpha_{ji} \in \{0,1\} \qquad i = 1, \ldots, n \quad j = 1, \ldots, m \tag{3.96}$$

$$\beta_j \in \{0,1\} \qquad j = 1, \ldots, m \tag{3.97}$$

Concernant la formulation étendue, introduisons en premier lieu la collection de tous les sous-ensembles réalisables d'objets :

$$\tilde{S}^{BPP} = \left\{ \tilde{S}_j^{BPP} = (y_1, y_2, \ldots, y_n) : \sum_{i=1}^{n} w_i y_i \leq W,\ y_i \in \{0,1\} (i = 1, 2, \ldots, n) \right\}.$$

Le coût associé au sous-ensemble $\tilde{S}_j^{BPP}$ peut être calculé ainsi :

$$\tilde{C}_{\tilde{S}_j^{BPP}} = K - \sum_{i \in \tilde{S}_j^{BPP}} \sum_{i' \in \tilde{S}_j^{BPP}} q_{ii'}.$$

La formulation étendue (A) utilise alors des variables binaires γ_j qui valent 1 si le sous-ensemble j est dans la solution optimale et peut s'écrire ainsi :

$$(A) \quad \min \sum_{j \in \tilde{S}^{BPP}} \tilde{C}_{\tilde{S}_j^{BPP}} \gamma_j \tag{3.98}$$

$$\sum_{j \in \tilde{S}^{BPP} : i \in j} \gamma_j = 1 \qquad i = 1, \ldots, n \tag{3.99}$$

$$\gamma_j \in \{0,1\} \qquad j \in \tilde{S}^{BPP}. \tag{3.100}$$

Procédure de séparation Pour la formulation étendue (A), le *sous-problème* obtenu est un problème de sac à dos quadratique en 0-1 (Kellerer et al. (2004)). Il peut être modélisé avec une variable binaire y_i qui vaut 1 si l'objet i est sélectionné dans le sous-ensemble $\tilde{S}_j^{BPP^*}$:

$$\max \sum_{i=1}^{n} \pi_i^* y_i + \sum_{i=1}^{n} \sum_{i'=1}^{n} q_{ii'} y_i y_{i'} \tag{3.101}$$

$$\sum_{i=1}^{n} w_i y_i \leq W \tag{3.102}$$

$$y_i \in \{0,1\} \qquad i = 1, \ldots, n. \tag{3.103}$$

Si la solution optimale de ((3.101) - (3.103)) est inférieure ou égale à K, alors toutes les contraintes duales sont satisfaites par la solution π_i^*, sinon le sous-ensemble $\tilde{S}_j^{BPP^*} = \{i : y_i = 1\}$ définit la plus violée.

Résolution exacte Nous avons développé un algorithme de branch-and-price pour résoudre optimalement ce problème. La stratégie de branchement utilisée est celle de Ryan-Foster : en chaque nœud deux objets sont mis ensemble ou séparemment. Nous avons utilisé SCIP et le langage C++ pour l'implémentation avec une stratégie en profondeur d'abord pour l'exploration de l'arbre.

Instances Les tests ont été effectués sur les instances classiques de la littérature. Pour chacune des 5 tailles d'instances utilisées ($n = 10$, $n = 20$, $n = 30$, $n = 50$, $n = 100$), nous avons considéré différentes valeurs possibles pour la capacité et pour les poids afin d'obtenir 9 instances différentes à chaque fois, soit 45 instances au total. Nous avons ensuite modifié aléatoirement les poids et la capacité afin d'obtenir des instances différentes sur lesquelles appliquer la réoptimisation.

			Base Model				Local Reopt A				Local Reopt B			
		Weights	10%		20%		10%		20%		10%		20%	
		Capacity	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%
n=10	*C1*	*W1*	0.0	0.0	0.0	0.0	0.1	0.0	0.1	0.1	0.1	0.0	0.0	0.0
		W2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
		W4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	C2	*W1*	0.0	0.1	0.0	0.0	0.8	0.1	0.7	0.1	0.2	0.0	0.2	0.1
		W2	0.0	0.0	0.0	0.0	0.1	0.0	0.1	0.1	0.0	0.0	0.0	0.0
		W4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.0
	C3	*W1*	0.1	0.1	0.1	0.1	1.4	0.3	1.3	0.3	0.2	0.0	0.3	0.0
		W2	0.0	0.1	0.0	0.1	0.6	0.1	0.5	0.1	0.2	0.0	0.2	0.0
		W4	0.0	0.0	0.0	0.0	0.2	0.0	0.3	0.0	0.1	0.0	0.1	0.0
		avg	0.0	0.0	0.0	0.0	0.4	0.1	0.3	0.1	0.1	0.0	0.1	0.0
n=20	*C1*	*W1*	0.0	0.1	0.0	0.1	0.6	0.0	0.1	0.1	0.1	0.0	0.1	0.1
		W2	0.0	0.0	0.0	0.0	0.1	0.0	0.1	0.0	0.0	0.0	0.0	0.0
		W4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	C2	*W1*	0.1	0.2	0.1	0.2	4.8	0.2	4.8	0.7	0.9	0.1	0.5	0.2
		W2	0.0	0.1	0.0	0.1	0.1	0.0	0.3	0.1	0.1	0.0	0.1	0.1
		W4	0.0	0.0	0.0	0.1	0.1	0.0	0.1	0.1	0.0	0.0	0.0	0.0
	C3	*W1*	0.2	0.3	0.2	0.3	23.0	0.8	6.8	1.7	3.3	0.1	1.0	0.3
		W2	0.1	0.2	0.1	0.2	5.0	0.1	1.4	0.3	0.8	0.1	0.5	0.2
		W4	0.0	0.2	0.1	0.2	5.1	0.0	0.5	0.2	0.4	0.0	0.3	0.1
		avg	0.1	0.1	0.1	0.1	4.3	0.1	1.6	0.4	0.6	0.0	0.3	0.1
n=30	*C1*	*W1*	0.1	0.2	0.1	0.2	2.5	0.2	3.8	0.8	0.5	0.1	0.3	0.2
		W2	0.1	0.1	0.0	0.1	0.5	0.0	0.4	0.2	0.1	0.0	0.1	0.1
		W4	0.0	0.1	0.0	0.1	0.1	0.0	0.2	0.1	0.0	0.0	0.0	0.0
	C2	*W1*	0.2	0.5	0.2	0.5	13.4	0.8	17.5	4.0	8.1	0.2	2.9	0.6
		W2	0.1	0.2	0.1	0.2	1.5	0.0	1.1	0.4	0.3	0.1	0.2	0.2
		W4	0.0	0.1	0.1	0.1	0.1	0.0	0.2	0.1	0.0	0.0	0.1	0.1
	C3	*W1*	0.5	0.6	0.6	0.7	37.3	4.2	24.2	8.5	29.3	0.3	9.0	1.0
		W2	0.3	0.7	0.3	0.9	11.9	0.4	10.6	1.6	8.2	0.2	4.2	0.6
		W4	0.1	0.4	0.2	0.5	10.6	0.1	1.3	0.8	2.0	0.1	1.0	0.3
		avg	0.2	0.3	0.2	0.4	8.7	0.6	6.6	1.8	5.4	0.1	2.0	0.3

TABLE 3.24 – *Temps d'exécution* – `BPP petites instances`

Tests et résultats Tous les tests ont été effectués sur un Pentium 4 cadencé à 3.2 GHz avec 1 GB RAM, sous Linux Ubuntu 12. Nous avons de plus utilisé Cplex 12.4 avec un thread unique et ses paramètres par défaut comme solveur. Le temps limite a été fixé à 1800 secondes.

Nous ne présenterons pas ici les résultats de la formulation compacte qui prend en compte la réoptimisation puisqu'elle ne permet de trouver une solution optimale que dans peu de cas (petites instances), la formulation étendue fournissant de bien meilleurs résultats.

Le modèle de base correspond à la formulation étendue initiale que l'on résout sur la nouvelle instance sans prendre en compte la solution de référence, i.e. sans réoptimiser. Nous avons ensuite les résultats pour la formulation (A) qui prend en compte le terme linéaire de la formulation initiale et le terme quadratique de la réoptimisation dans la fonction objectif, et enfin la formulation (B) qui ne comprend que le terme quadratique de la réoptimisation dans la fonction objectif.

Concernant les temps d'éxécution sur les petites instances (tableau 3.24), ils restent faibles quelle que soit l'approche, la résolution du modèle de base prend le moins de temps puisque celui-ci ne prend pas en compte la solution de référence, la résolution de la formulation (B) est un peu plus rapide que celle avec le terme linéaire original.

Pour les grandes instances (tableau 3.25), on observe le même comportement et le temps limite est parfois obtenu en utilisant la réoptimisation, notamment lors de la diminution de la valeur de la capacité des boîtes.

Concernant la valeur de l'objectif (tableaux 3.26 et 3.27), en utilisant la réoptimisation, nous arrivons à diminuer de manière très significative cette valeur, voire

			Base Model				Local Reopt A				Local Reopt B			
		Weights	10%		20%		10%		20%		10%		20%	
		Capacity	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%
n=50	*C1*	*W1*	0.5	0.8	0.2	1.6	tl	0.2	58.9	3.9	189.0	0.6	14.0	2.6
		W2	0.1	0.5	0.1	0.5	1.2	0.3	20.7	1.1	0.4	0.2	0.2	0.3
		W4	0.1	0.1	0.1	0.1	0.2	0.0	0.5	0.1	0.1	0.1	0.1	0.1
	C2	*W1*	0.1	2.1	1.0	1.4	tl	30.9	1063.1	tl	tl	1.0	6.2	2.4
		W2	0.4	0.4	0.4	1.8	15.1	0.1	tl	1.4	55.4	0.3	2.6	0.6
		W4	0.1	0.7	0.2	1.3	843.9	0.1	0.9	0.8	2.9	0.2	2.3	0.4
	C3	*W1*	2.1	1.8	1.6	1.9	tl	3.9	8.8	555.5	1677.2	4.1	25.0	12.8
		W2	12.7	3.3	8.0	4.9	tl	0.1	tl	11.4	tl	1.2	85.2	2.9
		W4	0.5	25.1	0.4	30.5	551.2	0.1	1.5	8.6	235.9	0.3	246.8	1.4
		avg	1.8	3.9	1.3	4.9	282.3	4.0	164.9	72.8	308.7	0.9	42.5	2.6
n=100	*C1*	*W1*	1.8	5.2	3.2	6.4	tl	31.8	tl	48.2	tl	5.8	tl	34.2
		W2	0.4	0.3	0.3	1.1	tl	0.1	1.5	1.4	1.0	0.4	0.5	0.9
		W4	0.3	0.4	0.4	0.5	1.0	0.1	tl	1.1	0.3	0.3	0.7	0.5
	C2	*W1*	0.7	8.1	5.4	6.5	tl	35.2	350.1	tl	tl	11.5	96.4	935.8
		W2	0.8	9.8	1.0	2.7	2.4	0.2	5.5	2.7	0.9	0.7	3.0	1.9
		W4	0.5	0.9	0.6	1.5	22.0	0.3	922.8	3.6	2.1	1.2	1.1	1.6
	C3	*W1*	8.7	6.3	10.0	9.2	tl	1803.4	tl	1278.6	tl	314.4	tl	tl
		W2	5.0	27.6	12.5	287.8	tl	0.6	tl	tl	tl	8.5	tl	43.6
		W4	2.3	37.8	1.5	5.4	63.3	0.3	tl	29.9	tl	2.3	tl	5.1
		avg	2.3	10.7	3.9	35.7	22.2	208.0	320.0	195.1	1.1	38.3	20.3	127.9

TABLE 3.25 – *Temps d'exécution* – `BPP grandes instances`

			Base Model				Local Reopt A				Local Reopt B			
		Weights	10%		20%		10%		20%		10%		20%	
		Capacity	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%
n=10	*C1*	*W1*	3.5	0.8	2.9	2.5	2.6	-0.2	2.0	0.2	-2.4	-4.5	-2.8	-4.0
		W2	3.9	1.4	4.2	2.4	3.2	-0.2	3.1	1.0	-3.2	-5.7	-3.5	-4.9
		W4	3.4	0.5	3.8	1.0	3.2	-0.4	3.4	0.1	-4.6	-7.0	-4.5	-6.3
	C2	*W1*	3.2	0.7	3.3	2.0	2.2	-0.2	2.0	0.6	-2.0	-3.7	-2.3	-3.1
		W2	4.5	1.3	4.7	2.7	3.6	-0.2	3.2	1.3	-2.1	-4.8	-2.4	-3.8
		W4	5.2	0.8	4.8	3.6	4.3	-0.2	3.8	1.6	-2.3	-5.4	-2.6	-4.2
	C3	*W1*	2.1	0.3	2.3	0.8	1.4	-0.3	1.5	-0.2	-2.0	-3.2	-2.0	-3.1
		W2	3.2	0.8	3.4	1.7	2.3	-0.1	2.3	0.5	-2.1	-3.9	-2.1	-3.6
		W4	4.1	1.5	4.4	1.5	3.2	-0.2	3.2	0.5	-1.8	-4.3	-2.1	-3.9
		avg	3.7	0.9	3.8	2.0	2.9	-0.2	2.7	0.6	-2.5	-4.7	-2.7	-4.1
n=20	*C1*	*W1*	7.9	4.5	7.8	5.8	6.8	-0.4	5.3	0.7	-5.4	-10.9	-6.7	-9.9
		W2	7.6	2.4	6.8	3.6	7.1	-0.5	5.8	1.3	-7.9	-13.4	-9.0	-11.7
		W4	5.6	0.6	5.5	1.6	5.3	-0.6	4.9	0.6	-11.7	-15.6	-12.0	-14.2
	C2	*W1*	8.4	5.0	8.4	6.3	6.5	-0.3	5.3	1.0	-3.4	-8.7	-4.6	-7.5
		W2	9.2	5.4	9.4	6.9	8.3	-0.4	6.8	1.4	-4.5	-10.9	-5.9	-9.4
		W4	8.5	2.9	7.5	4.4	7.9	-0.6	6.1	1.0	-6.9	-12.8	-8.3	-11.2
	C3	*W1*	6.6	3.8	6.5	4.6	4.9	-0.3	3.9	0.2	-3.1	-7.1	-4.0	-6.6
		W2	8.3	5.3	8.6	6.6	6.6	-0.1	5.8	0.8	-3.5	-8.4	-4.0	-7.7
		W4	9.6	5.8	9.6	7.5	7.7	-0.1	6.9	1.5	-3.4	-9.0	-4.0	-7.9
		avg	7.9	4.0	7.8	5.3	6.8	-0.4	5.6	0.9	-5.5	-10.8	-6.5	-9.6
n=30	*C1*	*W1*	12.3	8.1	12.4	9.4	10.0	-0.3	7.7	2.9	-7.5	-16.0	-9.9	-12.8
		W2	11.5	5.6	11.0	6.7	10.2	-0.4	8.5	2.4	-11.4	-19.6	-13.0	-17.0
		W4	9.1	3.0	8.5	3.2	8.4	-0.6	7.3	1.0	-16.4	-22.7	-17.3	-20.9
	C2	*W1*	12.2	8.7	12.4	9.0	10.2	-0.4	8.1	2.1	-4.5	-12.9	-6.6	-10.9
		W2	14.9	10.4	14.4	10.9	13.4	-0.3	10.1	2.8	-4.7	-15.6	-7.7	-13.0
		W4	12.7	5.2	12.0	7.7	12.0	-0.9	10.1	2.2	-9.2	-18.6	-10.8	-15.6
	C3	*W1*	10.3	8.0	10.6	8.1	8.0	-0.2	7.0	1.0	-3.8	-10.3	-4.9	-9.3
		W2	12.5	9.2	13.1	10.1	10.4	-0.1	9.2	1.9	-3.9	-12.3	-5.2	-10.6
		W4	14.1	10.7	14.4	11.0	11.8	-0.1	10.2	2.5	-4.1	-13.4	-5.7	-11.4
		avg	12.2	7.6	12.1	8.5	10.5	-0.4	8.7	2.1	-7.3	-15.7	-9.0	-13.5

TABLE 3.26 – *Valeur de l'objectif* – `BPP petites instances`

			Base Model				Local Reopt A				Local Reopt B			
		Weights	10%		20%		10%		20%		10%		20%	
		Capacity	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%
n=50	*C1*	*W1*	27	23	26	23	17.0	0.0	11.8	7.0	-10.7	-25.0	-15.7	-18.3
		W2	31	25	30	25	15.0	-0.3	12.7	2.0	-17.0	-29.0	-19.0	-26.0
		W4	37	31	36	30	14.0	-1.0	12.0	3.0	-23.0	-35.0	-26.0	-32.0
	C2	*W1*	22	18	22	18	15.8	-1.3	12.6	4.5	-6.7	-21.0	-10.3	-16.0
		W2	27	22	27	22	21.7	0.0	18.7	6.8	-5.8	-24.0	-9.3	-18.0
		W4	32	25	30	25	17.7	0.0	14.0	2.7	-14.3	-29.0	-17.3	-26.0
	C3	*W1*	18	15	18	15	12.5	0.0	11.3	3.8	-6.5	-16.0	-8.7	-12.8
		W2	21	17	21	17	16.8	0.0	14.7	3.5	-5.1	-19.0	-7.6	-16.0
		W4	23	19	24	19	18.3	0.0	18.7	5.7	-6.3	-21.0	-7.0	-17.0
		avg	26	22	26	22	16.5	-0.3	14.1	4.3	-10.6	-24.3	-13.4	-20.2
n=100	*C1*	*W1*	52	43	51	43	39.2	-0.3	28.8	15.7	-11.8	-48.0	-24.5	-33.3
		W2	71	57	69	56	36.0	-1.0	28.0	10.0	-35.0	-64.0	-42.0	-55.0
		W4	79	66	77	62	29.0	-1.0	26.0	5.0	-50.0	-73.0	-53.0	-66.0
	C2	*W1*	45	37	45	37	34.3	-1.0	26.0	9.9	-12.0	-42.0	-20.5	-32.6
		W2	58	46	57	46	44.0	0.0	37.0	14.0	-15.0	-52.0	-22.0	-40.0
		W4	62	50	60	50	35.0	-0.3	32.7	9.7	-29.0	-57.0	-31.0	-49.0
	C3	*W1*	39	32	39	32	38.6	-0.2	26.0	9.9	-8.3	-35.0	-16.0	-26.8
		W2	46	37	46	37	38.6	0.0	34.6	10.6	-10.5	-41.0	-14.5	-32.7
		W4	49	39	50	40	42.2	0.0	37.5	8.5	-9.2	-44.0	-15.2	-37.0
		avg	56	45	55	45	37.4	-0.4	30.7	10.4	-20.1	-50.7	-26.5	-41.4

TABLE 3.27 – *Valeur de l'objectif* – `BPP grandes instances`

à obtenir des solutions négatives avec la formulation (A). Nous améliorons ainsi la solution de référence, notamment lorsque la capacité est augmentée.

Les valeurs négatives sont bien sûr plus importantes lorsque seul le terme quadratique est considéré dans la fonction objectif.

Nous avons ensuite le nombre de colonnes générées avec les différentes formulations (tableaux 3.28 et 3.29). Les nombres de colonnes générées par les différentes approches sont ici globalement comparables.

Enfin, nous avons calculé la distance de Hamming entre la solution de référence et la nouvelle solution pour chaque formulation (tableaux 3.30 et 3.31).

Grâce à la réoptimisation, nous obtenons des solutions beaucoup plus proches de la solution de référence (tableau 3.30), notamment lorsque seul le terme quadratique est considéré dans la fonction objectif.

Ce résultat s'amplifie encore sur les grandes instances (tableau 3.31), surtout lorsque la capacité des boîtes est augmentée.

Capacitated P-Median Problem Le second problème considéré ici est le Capacitated P-median Problem (CPMP) (Reese (2006)), qui est classiquement rencontré dans des problèmes de localisation et qui consiste à localiser p sommets médians dans un réseau de manière à minimiser la somme des distances entre chaque client et le dépôt le plus proche en respectant la capacité des dépôts.

Il existe également de nombreuses formulations pour ce problème, dont la formulation classique comportant un nombre polynomial de variables et de contraintes et une formulation étendue que l'on peut résoudre par génération de colonnes et branch and price.

Formulations mathématiques Soit $I = \{1, \ldots, m\}$ l'ensemble d'indices des clients à allouer et soit $J = \{1, \ldots, n\}$ l'ensemble d'indices des dépôts possibles parmi p ($p < m$). La distance entre un client i et un dépôt j est donnée par la matrice symétrique c_{ij}. Chaque client a une demande ou un poids w_i et chaque dépôt j a une capacité maximale W_j.

			Base Model				Local Reopt A				Local Reopt B			
		Weights	10%		20%		10%		20%		10%		20%	
		Capacity	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%
n=10	*C1*	*W1*	16.8	19.9	17.2	18.5	23.8	13.3	19.9	17.6	15.5	14.5	14.1	16.3
		W2	15.4	18.9	13.9	19.1	14.6	12.3	15.3	15.1	14.3	14.9	14.2	15.8
		W4	16.2	19.1	15.9	17.4	16.9	14.7	16.7	16.6	14.2	16.6	14.3	15.8
	C2	*W1*	17.8	23.7	17.6	21.9	71.9	15.2	62.6	23.7	27.6	14.4	26.9	19.2
		W2	15.4	21.7	14.9	20.9	16.2	11.3	16.2	20.0	15.0	13.9	15.7	16.8
		W4	14.3	16.9	13.3	19.9	15.5	11.8	14.9	19.6	12.0	14.7	12.5	15.1
	C3	*W1*	23.3	26.9	23.1	26.4	119.3	42.7	110.4	45.2	33.7	13.5	37.6	14.5
		W2	19.6	23.7	19.8	24.3	42.4	14.9	36.6	21.4	25.2	13.6	25.7	16.3
		W4	16.8	23.1	17.4	22.2	25.3	10.6	33.7	13.9	20.1	13.4	22.2	15.6
		avg	17.3	21.5	17.0	21.2	38.4	16.3	36.2	21.4	19.7	14.4	20.3	16.1
n=20	*C1*	*W1*	29.9	34.1	28.4	35.7	39.0	22.5	33.2	32.9	26.6	29.8	30.5	32.5
		W2	28.8	34.9	32.0	33.8	32.7	27.2	33.0	29.6	25.0	31.9	27.5	29.8
		W4	32.8	37.3	32.4	35.3	33.0	31.4	34.3	30.8	30.5	34.3	31.8	32.3
	C2	*W1*	27.2	46.0	32.6	52.4	107.3	28.4	115.0	61.1	54.4	29.2	47.0	44.8
		W2	27.4	41.1	30.8	40.6	30.1	23.4	36.9	33.4	23.5	28.9	27.6	30.2
		W4	29.4	37.4	30.2	38.3	29.6	25.3	29.4	29.9	24.3	30.2	27.3	29.8
	C3	*W1*	45.8	61.7	45.0	58.6	319.4	62.5	184.5	92.0	96.7	28.9	68.5	57.2
		W2	33.3	53.2	38.0	54.0	81.8	24.7	64.3	40.2	48.2	29.1	44.5	40.1
		W4	27.8	48.5	35.2	51.1	53.7	20.2	38.8	37.8	37.0	28.1	37.3	34.4
		avg	31.4	43.8	33.8	44.4	80.7	29.5	63.2	43.0	40.7	30.0	38.0	36.8
n=30	*C1*	*W1*	42.5	54.6	44.7	61.7	70.7	39.8	68.3	64.1	44.2	45.4	48.4	55.4
		W2	45.5	51.3	45.8	55.2	51.0	38.9	52.1	47.3	39.5	47.6	42.1	45.8
		W4	48.6	56.4	49.2	56.0	51.8	45.4	52.1	47.3	43.7	50.8	45.7	48.2
	C2	*W1*	52.2	83.2	52.5	86.1	160.4	48.1	201.4	116.7	108.7	45.0	94.8	81.5
		W2	43.9	63.1	49.6	62.7	58.0	32.4	53.7	57.4	34.4	43.7	42.1	49.3
		W4	42.9	57.8	44.4	59.8	39.8	36.6	46.6	48.3	33.8	45.1	37.7	44.2
	C3	*W1*	79.9	87.0	81.8	93.7	369.2	123.3	273.3	185.4	177.3	51.3	142.4	105.2
		W2	59.2	91.7	61.8	102.7	134.1	45.0	143.2	95.1	75.4	44.4	89.3	83.3
		W4	47.1	79.1	53.8	83.2	98.7	32.1	68.4	72.2	53.5	42.3	59.0	57.8
		avg	51.3	69.3	53.7	73.4	114.8	49.1	106.5	81.5	67.8	46.2	66.8	63.4

TABLE 3.28 – *Colonnes générées* – `BPP petites instances`

			Base Model				Local Reopt A				Local Reopt B			
		Weights	10%		20%		10%		20%		10%		20%	
		Capacity	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%
n=50	*C1*	*W1*	2593	2613	2570	2652	4708	2549	2995	2630	2755	2577	2671	2657
		W2	2574	2606	2567	2604	2591	2563	2683	2607	2576	2574	2568	2580
		W4	2584	2594	2572	2570	2579	2567	2590	2567	2563	2580	2575	2577
	C2	*W1*	2545	2683	2626	2655	3355	2797	3965	3231	2768	2575	2693	2637
		W2	2581	2594	2586	2690	2660	2547	3461	2594	2640	2572	2595	2592
		W4	2564	2635	2576	2633	2931	2556	2577	2592	2594	2577	2611	2587
	C3	*W1*	2655	2654	2640	2651	4455	2538	2668	3146	2920	2581	2819	2835
		W2	2884	2779	2784	2807	3536	2538	3593	2712	2773	2594	2750	2692
		W4	2580	3049	2577	3170	2882	2542	2580	2694	2656	2572	2706	2641
		avg	2618	2690	2611	2715	3300	2577	3012	2753	2694	2578	2665	2644
n=100	*C1*	*W1*	10174	10284	10225	10305	10370	10200	10972	10280	10267	10143	10315	10321
		W2	10136	10148	10136	10195	10530	10122	10156	10157	10132	10158	10134	10158
		W4	10153	10160	10176	10178	10159	10140	10436	10164	10141	10167	10148	10159
	C2	*W1*	10122	10345	10265	10292	10470	10472	10419	10709	10324	10174	10315	10747
		W2	10136	10378	10142	10212	10124	10100	10164	10162	10098	10151	10146	10166
		W4	10143	10156	10145	10176	10193	10111	10369	10178	10150	10152	10126	10174
	C3	*W1*	10339	10280	10339	10325	10635	10287	10691	10970	10435	10249	10503	11827
		W2	10245	10704	10289	11526	10481	10080	10550	10557	10260	10159	10317	10451
		W4	10165	10434	10154	10260	10228	10085	10499	10313	10301	10147	10268	10248
		avg	10179	10321	10208	10385	10354	10177	10473	10388	10234	10167	10252	10472

TABLE 3.29 – *Colonnes générées* – `BPP grandes instances`

			Base Model				Local Reopt A				Local Reopt B			
		Weights	10%		20%		10%		20%		10%		20%	
		Capacity	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%
n=10	*C1*	*W1*	6.9	3.5	6.7	7.4	4.4	0.6	4.4	2.6	3.3	0.1	2.8	1.0
		W2	5.8	4.5	6.2	6.5	4.0	0.3	3.5	2.6	3.5	0.0	2.7	1.0
		W4	4.4	4.3	4.4	5.5	3.9	1.4	3.7	3.1	3.3	0.2	3.3	1.5
	C2	*W1*	8.1	4.3	7.7	7.9	5.0	1.3	3.7	3.8	2.9	0.0	2.3	0.9
		W2	7.2	4.4	7.2	6.9	4.4	0.9	3.4	3.6	3.6	0.0	2.8	1.0
		W4	6.6	2.8	6.7	7.8	4.8	0.5	4.5	2.8	4.0	0.1	3.8	1.4
	C3	*W1*	7.8	4.7	8.2	6.3	5.7	1.9	5.9	1.9	2.8	0.0	2.7	0.1
		W2	7.6	3.9	7.9	5.9	4.5	0.8	4.4	1.8	2.7	0.0	2.7	0.4
		W4	8.4	4.6	8.6	4.7	5.1	0.5	3.9	1.2	3.4	0.0	2.9	0.4
		avg	7.0	4.1	7.1	6.5	4.6	0.9	4.1	2.6	3.3	0.0	2.9	0.8
n=20	*C1*	*W1*	12.9	14.6	14.2	18.3	9.8	0.9	7.3	3.5	8.1	0.2	6.1	1.5
		W2	9.6	10.8	10.1	15.4	8.7	1.3	7.5	7.3	8.1	0.3	5.9	3.9
		W4	6.1	7.3	6.8	12.0	5.4	3.1	5.7	8.3	5.2	0.6	4.9	3.9
	C2	*W1*	16.0	16.2	17.6	19.9	10.5	1.6	8.0	5.2	8.0	0.1	5.6	1.6
		W2	12.9	16.0	15.0	19.3	10.5	1.0	8.4	4.5	9.0	0.2	6.7	2.2
		W4	10.0	14.8	11.1	17.6	8.9	2.6	7.3	6.5	8.2	0.8	6.1	3.7
	C3	*W1*	18.6	17.6	19.3	20.8	11.7	1.5	9.1	5.5	7.0	0.0	5.3	0.8
		W2	16.9	16.6	19.1	20.4	11.9	0.9	10.3	3.8	7.0	0.0	6.2	0.8
		W4	17.8	16.1	18.0	19.9	12.0	0.6	10.7	3.9	7.1	0.0	6.4	1.2
		avg	13.4	14.4	14.6	18.2	9.9	1.5	8.3	5.4	7.5	0.2	5.9	2.2
n=30	*C1*	*W1*	20.8	24.7	22.5	28.5	14.7	1.5	10.1	8.2	13.0	0.2	8.7	5.1
		W2	15.8	19.6	17.8	23.8	12.6	1.8	10.3	8.6	11.5	0.4	9.1	4.6
		W4	9.9	15.1	12.3	19.2	8.6	2.4	9.1	10.7	8.6	0.3	7.6	4.5
	C2	*W1*	25.8	29.7	27.4	30.7	17.3	2.8	12.7	9.6	14.8	0.2	9.7	3.1
		W2	22.2	26.9	23.2	29.5	17.9	1.2	12.4	6.4	15.8	0.1	10.2	3.6
		W4	16.5	22.4	19.4	27.3	15.1	2.8	13.2	9.6	13.9	1.1	11.1	5.5
	C3	*W1*	27.7	31.1	29.9	32.0	17.8	2.1	16.2	10.0	12.3	0.0	10.5	1.7
		W2	27.5	29.5	28.9	32.0	18.7	1.9	15.2	8.0	12.8	0.0	10.8	2.6
		W4	26.6	29.2	28.0	30.9	18.8	0.9	14.5	6.4	12.9	0.0	10.7	2.4
		avg	21.4	25.3	23.2	28.2	15.7	1.9	12.6	8.6	12.8	0.3	9.8	3.7

TABLE 3.30 – *H-Distance* – `BPP petites instances`

			Base Model				Local Reopt A				Local Reopt B			
		Weights	10%		20%		10%		20%		10%		20%	
		Capacity	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%
n=50	*C1*	*W1*	34	48	42	44	28	0	17	15	23	0	13	12
		W2	35	43	38	46	24	6	18	15	16	0	14	3
		W4	20	34	26	33	20	2	16	23	20	2	12	7
	C2	*W1*	49	49	48	55	39	13	33	17	32	0	21	9
		W2	42	46	43	53	35	0	22	18	28	0	21	10
		W4	30	45	37	43	24	0	22	11	23	0	17	5
	C3	*W1*	54	57	56	54	24	0	26	14	21	0	16	7
		W2	46	57	48	60	36	0	22	18	24	0	17	4
		W4	48	57	48	57	32	0	33	14	24	0	20	5
		avg	40	48	42	49	29	2	23	16	23	0	17	7
n=100	*C1*	*W1*	90	96	85	96	65	5	41	32	62	0	32	25
		W2	50	79	58	85	44	2	32	20	43	2	31	16
		W4	42	62	42	76	34	2	32	34	33	2	29	15
	C2	*W1*	98	102	91	107	63	3	44	26	59	0	39	17
		W2	80	95	82	100	60	0	44	28	59	0	41	18
		W4	74	96	80	92	42	6	42	20	41	0	38	12
	C3	*W1*	102	114	99	112	102	18	53	53	51	0	37	17
		W2	98	118	96	118	73	0	56	43	45	0	41	12
		W4	97	116	91	102	72	0	58	28	53	0	40	10
		avg	81	98	80	99	62	4	45	32	50	0	36	16

TABLE 3.31 – *H-Distance* – `BPP grandes instances`

La formulation classique du CPMP utilise deux ensembles de variables binaires : x_{ij} qui vaut 1 si le client i est affecté au dépôt j et 0 sinon, et y_j qui prend la valeur 1 si le dépôt j est choisi.

Nous obtenons alors la formulation suivante :

$$
\begin{aligned}
(CPMP) \quad \min \ & \sum_{i\in I}\sum_{j\in J} x_{ij}c_{ij} && && (3.104)\\
& \sum_{j\in J} x_{ij} = 1 && i = 1,\dots,n && (3.105)\\
& \sum_{i\in I} x_{ij}w_i \le W_j y_j && j = 1,\dots,m && (3.106)\\
& \sum_{j\in J} y_j = p && && (3.107)\\
& x_{ij} \in \{0,1\} && i \in I, \quad j \in J && (3.108)\\
& y_j \in \{0,1\} && j \in J && (3.109)
\end{aligned}
$$

dans laquelle nous pouvons de nouveau introduire facilement le terme quadratique induit par la réoptimisation afin d'obtenir une formulation compacte.

Concernant la formulation étendue comportant un nombre exponentiel de variables représentant les sous-ensembles réalisables de clients associés à un dépôt j ; nous introduisons la collection de tous les sous-ensembles réalisables de clients pour un dépôt médian fixé :

$$
{}_m\tilde{S}^{CPM} = \left\{ {}_m\tilde{S}_j^{CPM} = (y_1, y_2, \dots, y_n) : \sum_{i=1}^{n} w_i y_i \le W_m,\ y_i \in \{0,1\} (i = 1,2,\dots,n) \right\}.
$$

Le coût associé au sous-ensemble réalisable ${}_m\tilde{S}_j^{CPM}$ peut alors être obtenu de la manière suivante :

$$
\tilde{C}_{{}_m\tilde{S}_j^{CPM}} = \alpha \sum_{i\in {}_m\tilde{S}_j^{CPM}} \sum_{i'\in {}_m\tilde{S}_j^{CPM}} d_{ii'} - \beta \sum_{i\in {}_m\tilde{S}_j^{CPM}} \sum_{i'\in {}_m\tilde{S}_j^{CPM}} q_{ii'}.
$$

La formulation étendue (C) utilise ainsi des variables binaires ξ_j qui prennent la valeur 1 si le sous-ensemble j est dans la solution optimale :

$$
\begin{aligned}
(C) \quad \min \ & \sum_{j\in {}_m\tilde{S}^{CPM}} \tilde{C}_{{}_m\tilde{S}_j^{CPM}} \xi_j && && (3.110)\\
& \sum_{j\in {}_m\tilde{S}^{CPM} : i\in j} \xi_j = 1 && i = 1,\dots,n && (3.111)\\
& \sum_{j\in {}_m\tilde{S}^{CPM}} \xi_j = p && && (3.112)\\
& \xi_j \in \{0,1\} && j \in_m \tilde{S}^{CPM}. && (3.113)
\end{aligned}
$$

Procédure de séparation Pour la formulation étendue (C), nous obtenons *m sous-problèmes*, un par dépôt potentiel, qui sont encore une fois des problèmes de sac à dos quadratique en 0-1. Nous utilisons ici des variables binaires y_i qui prennent la valeur 1 si le client i est sélectionné dans le sous-ensemble j^* et nous

		Base Model				Local Reopt C				Local Reopt D			
	Distance	10%		20%		10%		20%		10%		20%	
	Weights	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%
n=50	*p=5*	44.3	21.6	36.5	21.9	45.3	26.2	36.4	25.0	44.9	14.2	32.8	13.1
	p=12	314.6	36.2	212.6	19.5	177.8	10.7	114.8	14.7	91.9	15.4	85.1	14.8
	p=16	332.1	223.3	68.0	175.1	144.8	13.6	100.0	15.0	27.7	15.5	25.1	14.9
	p=20	297.4	277.7	104.5	102.4	164.2	8.1	141.1	7.7	262.1	13.8	223.5	13.6

TABLE 3.32 – *Temps d'exécution* – `CPM instances`

obtenons les sous-problèmes suivants :

$$\max \sum_{i=1}^{n}(d_{im} + \pi_i^*)y_i + \sum_{i=1}^{n}\sum_{i'=1}^{n} q_{ii'}y_i y_{i'} \tag{3.114}$$

$$\sum_{i=1}^{n} w_i y_i \leq W_m \tag{3.115}$$

$$y_i \in \{0,1\} \qquad i = 1,\ldots,n. \tag{3.116}$$

Si la solution optimale de ((3.114) - (3.116)) est inférieure ou égale à λ^* (la valeur optimale de la variable duale λ de la contrainte (3.112)) alors toutes les contraintes duales sont satisfaites par la solution π_i^* et λ^*, sinon le sous-ensemble ${}_m\tilde{S}^{CPM^*} = \{i : y_i = 1\}$ définit la plus violée.

Résolution exacte Nous avons développé un algorithme de branch and price pour résoudre optimalement ce problème. Nous avons utilisé SCIP et le langage C++ pour l'implémentation avec une stratégie en profondeur d'abord pour l'exploration de l'arbre.

Instances Les tests ont été effectués sur les instances classiques de la littérature de taille $n = 50$. Nous avons considéré différentes valeurs possibles pour p ($p = 5$, $p = 12$, $p = 16$, $p = 20$) afin d'obtenir 4 instances. Nous avons ensuite modifié aléatoirement les distances et les poids afin d'obtenir des instances différentes sur lesquelles appliquer la réoptimisation.

Tests et résultats Tous les tests ont été effectués sur un Pentium 4 cadencé à 3.2 GHz avec 1 GB RAM, sous Linux Ubuntu 12. Nous avons de plus utilisé Cplex 12.4 avec un thread unique et ses paramètres par défaut comme solveur. Le temps limite a été fixé à 1800 secondes.

Nous ne présenterons pas ici les résultats de la formulation compacte qui prend en compte la réoptimisation puisqu'elle ne permet de trouver une solution optimale que dans peu de cas ($p = 5$), la formulation étendue fournissant de bien meilleurs résultats.

Le modèle de base correspond à la formulation étendue initiale que l'on résout sur la nouvelle instance sans prendre en compte la solution de référence, i.e. sans réoptimiser. Nous avons ensuite les résultats pour la formulation (C) qui prend en compte le terme linéaire de la formulation initiale et le terme quadratique de la réoptimisation dans la fonction objectif, et enfin la formulation (D) qui ne comprend que le terme quadratique de la réoptimisation dans la fonction objectif.

Concernant les temps d'exécution (tableau 3.32), on remarque qu'ils sont comparables lorsque les poids diminuent, mais les formulations (C) et (D) sont résolues

		Base Model				Local Reopt C				Local Reopt D			
	Distance	10%		20%		10%		20%		10%		20%	
	Weights	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%
n=50	*p=5*	747	724	714	690	705	682	674	651	293	242	268	222
	p=12	442	396	421	377	357	298	340	285	-615	-781	-628	-791
	p=16	364	322	343	306	246	188	230	177	-1108	-1302	-1129	-1318
	p=20	354	291	332	275	225	132	206	117	-1726	-2097	-1761	-2143

TABLE 3.33 – *Valeur de l'objectf* – `CPM instances`

		Base Model				Local Reopt C				Local Reopt D			
	Distance	10%		20%		10%		20%		10%		20%	
	Weights	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%
n=50	*p=5*	9255	9273	9086	9290	6354	6793	6275	6824	6821	6841	6496	6730
	p=12	7272	6803	7061	6596	5855	6070	5688	6057	5936	6498	5919	6429
	p=16	6880	6958	6405	6664	5852	6230	5725	6245	5930	6641	5865	6453
	p=20	6324	6589	6121	6460	5402	5936	5425	5968	5697	6282	5636	6232

TABLE 3.34 – *Colonnes générées* – `CPM instances`

plus rapidement lorsque les poids augmentent. Nous remarquons aussi que la formulation (D) semble plus rapide que la formulation (C), sauf pour $p = 20$.

Les valeurs de la fonction objectif (tableau 3.33) sont bien inférieures lorsque le terme quadratique induit par la réoptimisation est pris en compte (formulation (C)) et on constate, pour la formulation (D), que les valeurs deviennent très négatives lorsque p augmente.

Le nombre de colonnes générées (tableau 3.34) par les différentes approches sont du même ordre de grandeur.

Grâce à la réoptimisation, nous obtenons des solutions beaucoup plus proches de la solution de référence (tableau 3.35), notamment lorsque seul le terme quadratique est considéré dans la fonction objectif.

Conclusions

Nous avons présenté une nouvelle approche, la réoptimisation locale, permettant de prendre en compte la réoptimisation et qui s'appuie sur la notion de proximité locale. Cette approche peut s'appliquer sur tous les problèmes de type partitionnement d'ensembles qui sont habituellement résolus par génération de colonnes. Nous avons montré l'intérêt de cette approche sur deux problèmes classiques : le bin packing problem et le capacitated p-median problem.

		Base Model				Local Reopt C				Local Reopt D			
	Distance	10%		20%		10%		20%		10%		20%	
	Weights	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%	-10%	+10%
n=50	*p=5*	9	11	12	14	7	8	9	10	3	0	3	0
	p=12	22	19	23	22	13	4	14	7	4	0	4	0
	p=16	23	18	24	22	11	2	13	3	4	0	4	0
	p=20	25	23	27	25	15	5	15	6	6	0	5	0

TABLE 3.35 – *H-Distance* – `CPM instances`

3.2 PROGRAMMATION QUADRATIQUE

Dans cette section, nous considérons le problème du sac à dos quadratique, avec ou sans contrainte de cardinalité additionnelle. Nous considérons l'introduction de techniques de réoptimisation dans les processus itératifs de résolution induits par les relaxations et décompositions connues pour ce problème. Nous nous intéresserons ensuite à la résolution approchée de ce type de problème via des heuristiques duales fondées sur des relaxations de celui-ci. Nous nous focaliserons enfin sur la reformulation de ces problèmes via des techniques de convexification afin d'améliorer le comportement des solveurs commerciaux standards.

3.2.1 Réoptimisation pour le problème du sac à dos quadratique

Le problème du sac à dos quadratique consiste à maximiser une fonction objectif quadratique sous une contrainte linéaire de capacité. Il peut être formulé de la façon suivante :

$$(QKP) \begin{cases} \max & \sum_{i\in N}\sum_{j\in N} c_{ij}x_i x_j \\ s.c. & \sum_{j\in N} a_j x_j \leq b \\ & x_j \in \{0,1\} \quad j \in N \end{cases} \tag{3.117}$$

avec $N = \{1,\ldots,n\}$, les coefficients a_j $(j \in N)$ et c_{ij} $(i \in N, j \in N)$ sont des entiers positifs et b est un entier tel que $max\{a_j : j \in N\} \leq b < \sum_{j\in N} a_j$.

Le problème (QKP) est une généralisation du problème de sac à dos linéaire en variables 0-1, dans lequel tous les $c_{ij} = 0$, pour tout $i \neq j$ et il est connu comme étant *NP*-difficile. Des états de l'art sur les approches de résolution et les applications sont disponibles dans Gallo et al. (1980) et Pisinger (2007).

Les majorants produits par les meilleures approches exactes sont basés sur des méthodes Lagrangiennes (relaxation et décomposition (Gueye et Michelon (2005), Michelon et Maculan (1993), Michelon et Veilleux (1996), Rodrigues et al. (2009))) ou sur la programmation semidéfinie (Helmberg et Rendl (1998; 2000)).

À notre connaissance, la meilleure approche est celle proposée par Pisinger et al. (2007) (notée *méthode PRS*). Cette approche combine une relaxation Lagrangienne proposée par Caprara et al. (1999) (notée *méthode CPT*) et une décomposition Lagrangienne proposée par Billionnet et al. (1999), Billionnet et Soutif (2004) (notée *méthode BFS*), en tirant avantage que ces deux méthodes, *CPT* et *BFS*, sont complémentaires.

Il apparaît que le calcul de ces majorants Lagrangiens implique la résolution de nombreux sous-problèmes de sac à dos linéaires en 0-1. Ainsi, nous proposons d'introduire des techniques de réoptimisation pour améliorer l'efficacité de la phase de pré-traitement pour la résolution du problème du sac à dos quadratique en 0-1.

Dans cette section, nous proposons d'introduire la *réoptimisation* pour accélérer chaque séquence de résolution des problèmes de sac à dos linéaires en variables continues induites par la relaxation Lagrangienne de Caprara, Pisinger

et Toth (CPT) et par la décomposition Lagrangienne de Billionnet, Faye et Soutif (BFS). En effet, d'une part ces méthodes sont les deux briques de base de la méthode PRS, et d'autre part les deux codes originaux étaient disponibles auprès des auteurs.

Nous rappelons brièvement les principes des méthodes CPT et BFS et introduisons la réoptimisation dans la résolution des problèmes de sac à dos linéaires continus. Nous proposons d'exploiter et d'adapter les techniques de réoptimisation introduites pour améliorer les temps de résolution des algorithmes itératifs (tels que les méthodes de sous-gradient) qui ont été proposées pour la résolution du dual Lagrangien du problème de sac à dos linéaire en 0-1 bidimensionnel Thiongane et al. (2005; 2006). Enfin, de nombreux résultats numériques valident notre approche.

Majorants Lagrangiens pour (QKP)

Les approches Lagrangiennes CPT (Caprara et al. (1999)) et BFS (Billionnet et al. (1999), Billionnet et Soutif (2004)) pour (QKP) combinent des techniques de linéarisation ou de décomposition avec la relaxation ou la décomposition Lagrangienne. Les étapes principales de ces méthodes duales sont décrites ci-dessous. Un très grand nombre de problèmes de sac à dos linéaires en 0-1 doivent être résolus (i.e. environ n problèmes de taille n à chaque itération des procédures de sous-gradient pour CPT par exemple).

La relaxation Lagrangienne de CPT Caprara et al. (1999) proposent un majorant, fondé sur une relaxation Lagrangienne (après une étape de linéarisation), qui est obtenu via la résolution de très nombreux problèmes de sac à dos linéaires en 0-1.

La première étape de leur procédure consiste à utiliser la linéarisation RLT (Reformulation-Linearization Technique) (Adams et Sherali (1986)). Dans un second temps, en relâchant les n^2 constraintes de symétrie, les auteurs prouvent que chaque relaxation Lagrangienne peut être résolue via n sous-problèmes indépendants de sac à dos linéaires en 0-1 plus un problème global de sac à dos linéaire en 0-1 dépendant des solutions des n précédents. Pour accélérer l'algorithme de sous-gradient utilisé pour résoudre le dual Lagrangien, seule la relaxation continue de ces $n+1$ problèmes de sac à dos linéaires en 0-1 est considérée à chaque itération.

Concernant les expérimentations numériques, Caprara et al. (1999) montrent que cette approche donne des majorants de bonne qualité (notés ub_{CPT} par la suite) pour les instances à forte densité et nécessite un temps de résolution assez faible.

La décomposition Lagrangienne de BFS En utilisant une partition préliminaire des données du problème, Billionnet et al. (1999) proposent un majorant ub_{BFS} de $v(QKP)$ basé sur la décomposition Lagrangienne.

De manière synthétique, leur méthode BFS s'effectue en trois étapes :

- *Création des clusters* : partition des variables en m classes disjointes k, $k \in \{1,\ldots,m\}$.
- *Décomposition Lagrangienne* : séparation du problème en m sous problèmes de sac à dos linéaires en 0-1 indépendants, associés aux m clusters.
- *Calcul du majorant* : résolution du dual Lagrangien par un algorithme de sous-gradient.

n_k est le nombre de variables du cluster k, $k = 1,\ldots,m$. À chaque itération de l'algorithme de sous-gradient, 2^{n_k} sacs à dos linéaires en 0-1 sont résolus pour chaque cluster k. De manière à accélérer la résolution, les auteurs proposent de relâcher les conditions d'intégrité sur les variables. Ils ne considèrent ainsi, comme pour CPT, que la relaxation continue des sous problèmes de sac à dos linéaires en 0-1.

Billionnet et Soutif (2004) soulignent que leur approche, imbriquée dans un schéma de séparation et d'évaluation progressive, consomme plus de temps mais produit de meilleurs résultats que la méthode CPT pour les instances de faible et moyenne densité.

Comme la méthode PRS utilise une hybridation de CPT et BFS, nous proposons d'introduire des techniques de réoptimisation spécifiques dans CPT et BFS.

Réoptimisation pour (QKP)

Dans ce cadre, la *réoptimisation* consiste à exploiter la résolution d'une instance donnée (P) d'un problème pour accélérer la résolution d'une autre instance (P') voisine de la précédente (i.e. avec une légère modification d'une partie des données). Ainsi, en résolvant (P), il est important de sauvegarder de l'information qui pourrait être utile pour la résolution de (P'). Les techniques de réoptimisation peuvent être appliquées sur de grandes séquences d'instances (Belgacem et Hifi (2008a;b), Crema (2000), Geoffrion et Nauss (1977), Hifi et al. (2005), Holm et Klein (1984), Libura (1980), Touati (2008), Touati et al. (2009)).

L'un des points critiques dans la résolution de chaque dual Lagrangien de (QKP) est le très grand nombre de *sacs à dos linéaires 0-1 continus* à résoudre. Afin de résoudre plus efficacement les majorants ub_{CPT} et ub_{BFS}, nous proposons de réduire le temps global de résolution des méthodes CPT et BFS en exploitant des techniques de réoptimisation introduites dans la méthode MARIE (Method combining Approximation and Reoptimization for Integer programming Evoluating instances) (Thiongane (2003), Thiongane et al. (2006)), pour la résolution d'une séquence (finie) de problèmes de sac à dos linéaires en 0-1.

Nous détaillons ici les techniques de réoptimisation spécifiques que nous proposons d'intégrer dans CPT et BFS pour l'obtention des majorants et dans la phase de réduction et décrivons le schéma général de réoptimisation proposé pour ces deux méthodes.

Réoptimisation pour les sacs à dos 0-1 continus La méthode MARIE (Thiongane (2003), Thiongane et al. (2006)) combine une phase de fixation de variables

et une phase de recherche locale pour améliorer le minorant de la valeur du sac à dos bidimensionel produit par l'heuristique Lagrangienne. De plus, la séquence de problèmes de sac à dos linéaires 0-1 est résolue en utilisant des techniques de réoptimisation pour réduire le temps total de résolution.

Nous présentons ci-dessous l'approche de réoptimisation spécifique que nous proposons d'intégrer dans (CPT) et (BFS). Pour cela, nous considérons le problème de sac à dos linéaire 0-1 générique suivant :

$$(K(u)) \begin{cases} \max & \sum_{j\in I} c(u)_j x_j \\ s.c. & \sum_{j\in I} a_j x_j \leq cap \\ & x_j \in \{0,1\} \quad j \in I \end{cases}$$

où u est un nombre réel positif, l'ensemble I est un sous-ensemble de N, les coefficients a_j $(j \in I)$ sont des entiers positifs, les coefficients $c(u)_j$ $(j \in I)$ sont des réels, et cap est un entier tel que $max\{a_j : j \in I\} \leq cap < \sum_{j\in I} a_j$. Cette formulation est valide pour tous les problèmes de sac à dos résolus par les algorithmes de sous-gradient (avec un multiplicateur de Lagrange u) pour résoudre les duaux. En effet, plusieurs séquences de sac à dos linéaires 0-1 doivent être résolues tout au long des algorithmes de sous-gradient. Dans chaque séquence, tous les problèmes ont la même contrainte et ne diffèrent que par leur fonction objectif via le paramètre u.

Résoudre la relaxation continue $(\overline{K}(u))$ de $(K(u))$ est équivalent à déterminer un multiplicateur optimal $\mu(u)$ associé à la contrainte de $(K(u))$. Il est bien connu que :

$$\exists\, i \in I \text{ tel que } \mu(u) = \frac{c(u)_i}{a_i}$$

et les deux ensembles $U(u)$ et $L(u)$ forment une tripartition de I qui satisfait les deux propriétés :

$$(i)\ \forall\, j \in U(u)\ \frac{c(u)_j}{a_j} \geq \mu(u) \geq \frac{c(u)_l}{a_l}\ \forall\, l \in L(u)$$

$$(ii)\ \sum_{j\in U(u)} a_j \leq cap < \sum_{j\in U(u)} a_j + a_i$$

La solution $\overline{x}$ est alors obtenue par :

$$\overline{x}_j = \begin{cases} 1 & j \in U(u) \\ 0 & j \in L(u) \\ c(u)_i(\frac{cap - \sum_{l\in U(u)} a_l}{a_i}) & j = i \end{cases}$$

À partir de l'ensemble des ratios $\frac{c(u)_j}{a_j}$, une séquence de tripartitions est réalisée à partir de différentes valeurs cibles pour vérifier si la condition (ii) est satisfaite. Différentes versions de l'algorithme de la figure 3.13 ont été proposées (Balas et Zemel (1980), Fayard et Plateau (1977), Lawler (1979)). La version proposée ici a une complexité moyenne linéaire (Aho et al. (1976.), Fayard et Plateau (1982)) (on peut noter que, dans le cadre de la réoptimisation, l'indice i est sauvegardé même si $\overline{x}$ est une solution optimale de $(K(u))$ (dans ce cas $\overline{x}_j$ est égal à 0 ou 1)).

/* initialisation */
$L \leftarrow I; U \leftarrow \emptyset; \sigma \leftarrow 0$

/* algorithme itératif */
while $\sigma < cap$ **do**
choisir un objet h dans L
$L_1 \leftarrow \{j \in L |\ \mathrm{c(u)}_j/\mathrm{a}_j \geq (c(u)_h/a_h\} \backslash \{h\}$
$L_2 \leftarrow \{j \in L |\ \mathrm{c(u)}_j/\mathrm{a}_j < c(u)_h/a_h\}$
if $(\sigma + \sum_{L_1} a_j > cap)$ **then** $L \leftarrow L_1$
else
$\sigma \leftarrow \sigma + \sum_{j \in L_1} a_j; U \leftarrow U \cup L_1$
if $(\sigma < cap)$ **then**
$U \leftarrow U \cup \{h\}; \sigma \leftarrow \sigma + a_h; L \leftarrow L_2$
endif
endif
end while
$i \leftarrow h$ /*Pour la réoptimisation*/
$\overline{x}_j \leftarrow 1, j \in U; \overline{x}_j \leftarrow 0, j \notin U$
if $\sigma = cap$ **then**
$\overline{x}$ est réalisable et optimal pour $(K(u))$; $v(K(u)) \leftarrow c(u)\overline{x}$
else
$\sigma \leftarrow \sigma - a_i$
$\overline{x}_i \leftarrow (cap - \sigma)/a_i; U(u) \leftarrow U \backslash \{h\}; L(u) \leftarrow N \backslash U; v(\overline{K}(u)) \leftarrow c(u)\overline{x}$
endif

FIGURE 3.13 – *Résolution du problème de sac à dos continu* $(\overline{K}(u))$ *en temps linéaire*

Il est important de rappeler que le temps de résolution de cet algorithme va diminuer puisque chaque valeur cible (appelée aussi *pivot*), choisie pour réaliser la séquence de partitions, satisfait au moins l'une des propriétés suivantes :

- *être proche de la valeur médiane de l'ensemble des ratios concernés* $\frac{c(u)_j}{a_j}$
  Ce choix permet de créer une séquence de partitions équitables de l'ensemble des ratios (voir, e.g., Bourgeois et Plateau (1992), Nagih et Plateau (2000a;b), Sedgewick (1978)). En effet, pour chaque ensemble de séquences, environ la moitié des éléments de l'ensemble précédent est considérée. Cela tend à s'approcher du pire cas de la complexité linéaire, de manière plus simple que la méthode de la médiane (Aho et al. (1976.), Blum et al. (1972)) utilisée dans l'algorithme de Lawler (Lawler (1979)).
- *être proche du multiplicateur optimal* $\mu(u)$
  Cette propriété peut être utilisée pour le premier pivot du processus quand il est possible d'exploiter la distribution des données des instances (voir par exemple, Fayard et Plateau (1979)). Ce choix tend à réduire de manière importante le nombre d'itérations de l'algorithme. Dans le meilleur des cas (le pivot initial est optimal), cela permet de résoudre directement $(\overline{K}(u))$ en une seule itération. Notre processus de réoptimisation tend vers cet objectif.

Dans le contexte du dual Lagrangien, tout au long des algorithmes de sous-gradient, nous devons résoudre une séquence de sacs à dos 0-1 qui ont la même

contrainte et qui ne diffèrent que par leur fonction objectif. Pour accélérer ces procédures itératives, qui déterminent la tripartition optimale de I, nous proposons de réduire le nombre d'échanges en utilisant le multiplicateur optimal trouvé à l'itération précédente de l'algorithme de sous-gradient. Nous considérons ainsi, à chaque itération l, l'indice i^l de la variable de base optimale de la relaxation continue $(\overline{K}(u^l))$. Pour résoudre $(\overline{K}(u^{l+1}))$, nous considérons $c(u^{l+1})_{i^l}/a_{i^l}$ comme *premier pivot*. Nous espérons ainsi que cette valeur cible n'est pas trop éloignée de $\mu(u^{l+1})$ et que la première tripartition de I est proche de la tripartition optimale qui caractérise $\mu(u^{l+1})$ (voir Figure 3.14).

Réoptimisation dans les méthodes CPT et BFS Le calcul du dual Lagrangien des majorants ub_{CPT} et ub_{BFS} implique la résolution de nombreux sous problèmes de sacs à dos linéaires 0-1 continus. Chaque phase de prétraitement des méthodes CPT et BFS consiste en une réduction dynamique des variables du problème (QKP), qui met à jour à la fois le minorant, le majorant et la taille du problème jusqu'à ce que la phase de réduction échoue. Cela implique la résolution d'un très grand nombre de sacs à dos linéaires 0-1 continus dans lesquels nous proposons d'intégrer les techniques de réoptimisation.

Nous proposons un schéma générique de réoptimisation (Figure 3.15) qui exploite le plus possible les codes existants et utilisés par les auteurs de Caprara et al. (1999) pour CPT, et Billionnet et al. (1999) et Billionnet et Soutif (2004) pour BFS. Comme nous ne modifions pas la structure des codes existants, nous pourrions améliorer encore les temps de calcul en utilisant des structures dédiées à la réoptimisation et nous pourrions également améliorer la phase de fixation de variables intégrant la réoptimisation, mais cela impliquerait une réécriture complète de cette partie du code.

Afin de donner une évaluation du nombre de sacs à dos linéaires 0-1 continus résolus lors du calcul des bornes et dans la phase de fixation de variables des deux codes, on appelle *tour* le nombre d'itérations de la phase de prétraitement et *iter* le nombre d'itérations de l'algorithme de sous-gradient pour résoudre le dual Lagrangien (p est le nombre de problèmes de sac à dos résolus à chaque itération). De plus, n est toujours la taille du problème courant (QKP) au début de la phase de prétraitement.

Réoptimisation dans la méthode CPT

La phase itérative de prétraitement s'arrête quand aucune nouvelle variable ne peut être fixée. Ce critère d'arrêt permet d'avoir une valeur pour le paramètre *tour*. Le critère d'arrêt pour la calcul de ub_{CPT} est un nombre fixé d'itérations : *iter* est égal à $n + 200$. Enfin, le paramètre p est égal à $n + 1$ (n sacs à dos linéaires 0-1 continus $(CKP_j(\mu))$ de taille $n - 1$ plus un sac à dos linéaire 0-1 continu (CKP) de taille n (voir Section 3.2.1).

Réoptimisation dans la méthode BFS

Le nombre d'itérations de la phase de prétraitement *tour* dépend d'un critère d'arrêt basé sur une valeur seuil de réduction de la taille du problème. Le nombre d'itérations *iter* pour le calcul de ub_{BFS} provient d'un critère d'arrêt qui combine des valeurs seuil pour le saut de dualité $\frac{upperbound-lowerbound}{lowerbound}$ et pour les multipli-

/* initialisation */
$i^l \leftarrow$ *indice de la variable de base optimale de* $(\overline{K}(u^l))$
$L \leftarrow$ *ensemble d'indices des objets dans l'ordre trouvé à la fin de la résolution de* $(\overline{K}(u^l))$
$h \leftarrow i^l$
$ratio \leftarrow c(u^{l+1})_h / a_h$
$L_1 \leftarrow \{j \in L | c(u^{l+1})_j / a_j \geq ratio\} \backslash \{h\}$
$L_2 \leftarrow \{j \in L | c(u^{l+1})_j / a_j < ratio\}$
if $(\sum_{L_1} a_j > cap)$ **then** $L \leftarrow L_1$
else
  $\sigma \leftarrow \sum_{j \in L_1} a_j$
  if $(\sigma = cap)$ **or** $(\sigma + a_h = cap)$ **then**
    /* $(\overline{K}(u^{l+1}))$ et $(K(u^{l+1}))$ sont résolus */
    if $(\sigma = cap)$ **then** $U(u^{l+1}) \leftarrow L_1$ **else** $U(u^{l+1}) \leftarrow L_1 \cup \{h\}$ **endif**
    $i^{l+1} \leftarrow h\,; \overline{x}_j \leftarrow 1, j \in U(u^{l+1})\,; \overline{x}_j \leftarrow 0, j \notin U(u^{l+1}); v(K(u^{l+1})) \leftarrow c(u^{l+1})\overline{x}$
  else
    $U \leftarrow L_1$
    if $\sigma + a_h > cap$ **then**
      /* $(\overline{K}(u^{l+1}))$ est résolu */
      $i^{l+1} \leftarrow h\,; \overline{x}_h \leftarrow (cap - \sigma)/a_h\,; U(u^{l+1}) \leftarrow U$
      $\overline{x}_j \leftarrow 1, j \in U(u^{l+1})\,; \overline{x}_j \leftarrow 0, j \notin U(u^{l+1}) \cup \{h\}\,; v(\overline{K}(u^{l+1})) \leftarrow c(u^{l+1})\overline{x}$
    else /* $\sigma + a_h < cap$*/
      $\sigma \leftarrow \sigma + a_h; U \leftarrow U \cup \{h\}; L \leftarrow L_2$

/* algorithme itératif */
  voir Figure 3.13 avec $u = u^{l+1}$

    endif
  endif
endif

/* information sauvegardée pour l'instance suivante*/
  sauvegarder i^{l+1} *et l'ordre courant de l'ensemble* L

FIGURE 3.14 – *Réoptimisation pour le problème du sac à dos continu* $(\overline{K}(u^{l+1}))$ *après la résolution de* $(\overline{K}(u^l))$

$tour \leftarrow 1$
repeat
  $l \leftarrow 1$
  /* calcul du majorant */
  repeat
    on applique la *réoptimisation* pour chacun des p sacs à dos linéaires 0-1 continus $(\overline{K_j}(u^{l+1}))$ connaissant la résolution de $(\overline{K_j}(u^l))$ $j \in \{1, \ldots p\}$ (voir Figure 3.14)
    $l \leftarrow l + 1$
  until (critère d'arrêt)
  /* fixation de variables */
  /* *iter* est le nombre d'itérations pour calculer le majorant et
  $\underline{x}$ est la solution associée au meilleur minorant*/
  for each variable x_f **do**
    appliquer la *réoptimisation* pour les p sacs à dos linéaires 0-1 continus $(\overline{K_j}(u^{iter})|x_f$ fixé à $1 - \underline{x}_f)$ connaissant la résolution de $(\overline{K_j}(u^{iter}))$ $j \in \{1, \ldots p\}$ dans l'objectif de fixer x_f à $\underline{x}_f$ (voir Figure 3.14)
  endfor
$tour \leftarrow tour + 1$
until (critère d'arrêt)

FIGURE 3.15 – *Schéma générique de réoptimisation*

cateurs de Lagrange. Le paramètre p est égal à $\sum_{k=1}^{m} 2^{n_k}$. En effet, pour chaque cluster $k \in \{1, \ldots, m\}$, et pour chacune des 2^{n_k} fixations de variables x_i dans I_k, un sac à dos linéaire 0-1 continu de taille $n - n_k$ doit être résolu (voir le problème $(KP^k(x, \lambda, \mu))$.

Résultats

Cette section est dédiée aux résultats expérimentaux sur des instances générées aléatoirement. Les données sont générées selon une loi de distribution uniforme dans les intervalles suivants : $c_{ij} \in [1, 100]$, $a_j \in [1, 50]$ et $b \in [1, \sum_{j \in N} a_j]$ pour $i, j \in \{1, \ldots, n\}$ avec $n \in \{100, \ldots, 600\}$ pour CPT et $n \in \{100, \ldots, 400\}$ pour BFS. Dans les quatre tableaux, δ représente la densité de la matrice quadratique (25 %, 50 %, 75 % et 100 %) et les résultats sont des moyennes sur 10 instances. Tous les tests ont été effectués sur un Intel Xeon bi-processeur dual cœur 3 GHz avec 2 Go de RAM (mais un seul cœur était utilisé).

Ces tests ont été réalisés pour apprécier et évaluer l'impact de la réoptimisation sur les méthodes CPT et BFS. En partant des codes d'origine, nous n'avons ajouté que ce qui a trait à la réoptimisation sans modifier les paramètres fournis par les auteurs. En particulier, nous avons conservé la taille originale (par paquet de 5 variables) des clusters de la méthode BFS (on peut noter que les gains sont les mêmes quelque soient les tailles de clusters utilisés). De plus, concernant le code de la méthode BFS, des problèmes de mémoire sont apparus pour les instances à partir de 500 variables, ce qui explique la limite en taille pour les tests de BFS.

Chaque colonne *Ave.* donne les valeurs moyennes du pourcentage du nombre d'échanges économisés (Tableaux 3.36 et 3.38) et du pourcentage du temps CPU économisé (Tableaux 3.37 et 3.39) obtenus pour les résolutions des sacs à dos continus dans la phase de prétraitement. Les valeurs moyennes des écarts-types

sont également données dans les colonnes Dev.

Les résultats montrent que l'utilisation de la réoptimisation dans la phase de prétraitement permet d'économiser en moyenne 28.53% des échanges pour CPT et 27.73% pour BFS. De plus, on peut noter que malgré l'absence de structures de données dédiées à la réoptimisation, les temps moyens sont réduits de 4.15 % pour CPT et de 21.27 % pour BFS. Les écarts-types montrent que les gains sont réguliers pour CPT, alors que les gains semblent plus chaotiques pour BFS, même si, en moyenne, ils restent positifs dans tous les cas.

n \ δ	25 %		50 %		75 %		100 %	
	Ave.	Dev.	Ave.	Dev.	Ave.	Dev.	Ave.	Dev.
100	14.13	10.31	31.41	12.30	46.97	8.62	56.80	4.33
200	17.13	10.88	24.52	7.95	46.41	9.38	58.05	4.56
300	20.59	11.35	22.19	9.24	18.23	13.66	53.38	2.40
400	12.50	8.09	12.90	9.85	32.76	5.01	53.30	3.86
500	10.90	4.91	16.77	13.36	32.51	14.08	53.52	3.35
600	7.86	2.68	27.27	10.16	27.88	9.93	51.13	3.91

TABLE 3.36 – *Pourcentage du nombre d'échanges économisés pour CPT (%)*

n \ δ	25 %		50 %		75 %		100 %	
	Ave.	Dev.	Ave.	Dev.	Ave.	Dev.	Ave.	Dev.
100	6.11	7.43	7.66	6.73	20.67	15.94	17.60	7.69
200	3.70	7.73	4.81	20.36	12.70	12.36	19.60	8.84
300	8.11	11.73	2.35	6.88	5.12	8.70	11.73	8.60
400	3.31	5.42	1.22	4.11	4.15	4.21	13.70	5.42
500	0.68	6.54	2.70	8.59	7.36	11.14	13.43	6.82
600	1.85	3.59	2.08	4.05	0.80	11.60	11.89	6.95

TABLE 3.37 – *% du temps CPU économisé pour CPT (%)*

n \ δ	25 %		50 %		75 %		100 %	
	Ave.	Dev.	Ave.	Dev.	Ave.	Dev.	Ave.	Dev.
100	33.66	33.65	19.94	32.89	51.09	74.54	74.81	17.86
200	1.19	1.56	67.94	20.12	36.76	54.94	22.67	34.56
300	63.27	32.30	43.69	42.17	0.37	5.62	75.15	58.10
400	41.45	43.80	62.50	31.00	6.87	11.58	0.11	23.31

TABLE 3.38 – *Pourcentage du nombre d'échanges économisés pour BFS (%)*

Conclusion

Caprara et al. (1999) d'une part et Billionnet et al. (1999), Billionnet et Soutif (2004) d'autre part ont proposé des méthodes efficaces fondées sur la dualité Lagrangienne pour résoudre le problème du sac à dos quadratique en variables 0-1. Comme leurs phases de prétraitement impliquent la résolution de nombreux sacs à

n \ δ	25 %		50 %		75 %		100 %	
	Ave.	Dev.	Ave.	Dev.	Ave.	Dev.	Ave.	Dev.
100	8.29	12.64	5.00	4.39	28.59	28.06	37.47	15.57
200	0.78	0.86	34.15	37.97	25.45	44.40	8.26	15.95
300	43.96	29.28	29.54	21.29	1.57	5.29	36.69	33.51
400	27.93	31.39	42.96	20.21	11.56	17.97	2.65	11.68

TABLE 3.39 – *Pourcentage du temps CPU économisé pour BFS (%)*

dos linéaires en 0-1, nous avons proposé d'améliorer l'efficacité de ces méthodes en introduisant des techniques de réoptimisation. Nous avons proposé un schéma générique de réoptimisation valide pour ces deux méthodes Lagrangiennes qui exploite la résolution d'un très grand nombre de sacs à dos linéaires 0-1 continus. En utilisant les codes d'origine des auteurs amendés par nos techniques de réoptimisation, les tests ont permis de valider l'intérêt d'une telle aproche dans les schémas itératifs. Afin d'aller plus loin et de déterminer l'impact maximal de ce type de techniques, il serait intéressant d'adapter les structures de données des deux codes pour intégrer la réoptimisation dans la phase de réduction de variables.

3.2.2 Reformulation, résolution et relaxation pour le problème du sac à dos quadratique avec contrainte de cardinalité

Introduction

Le problème du sac à dos quadratique 0-1 avec contrainte de cardinalité $(E-kQKP)$ consiste à maximiser une fonction quadratique sous deux contraintes linéaires, la première est la contrainte classique de capacité du sac à dos et la seconde est une contrainte de cardinalité en égalité sur le nombre d'objets à insérer dans le sac. Il peut être formulé de la manière suivante :

$$(E-kQKP)\begin{cases} \max & f(x)=\sum_{i=1}^{n}\sum_{j=1}^{n}c_{ij}x_ix_j & \\ \text{s.t.} & \sum_{j=1}^{n}a_jx_j\leq b & (1)\\ & \sum_{j=1}^{n}x_j=k & (2)\\ & x_j\in\{0,1\} \quad j=1,\ldots,n & \end{cases}$$

avec n le nombre d'objets et toutes les valeurs, k (le nombre d'objets à mettre dans le sac), a_j (le poids de l'objet j), c_{ij} (le profit associé à la sélection des objets i et j) et b (la capacité du sac) sont des entiers positifs. Sans perte de généralité, la matrice $C=(c_{ij})$ est supposée symétrique.
De plus, nous supposons que $\max_{j=1,\ldots,n}a_j\leq b<\sum_{j=1}^{n}a_j$ pour éviter les solutions triviales ou les fixations évidentes de variables via la contrainte (1). Notons k_{max} le plus grand nombre d'objets qui peuvent être insérés dans le sac à dos, i.e. avec les plus petits a_j dont la somme n'excède pas b. $k\in\{2,\ldots,k_{max}\}$, avec k_{max} qui peut être trouvé en $O(n)$ (Balas et Zemel (1980), Fayard et Plateau (1982)). Dans le cas contraire, soit la valeur du problème est égale à $\max_{i=1,\ldots,n}c_{ii}$ (pour $k=1$), soit le domaine de $(E-kQKP)$ est vide (pour $k>k_{max}$).
Notons que si l'on supprime la contrainte (1), $(E-kQKP)$ se ramène au problème du k-cluster (Billionnet (2005)), et en supprimant la contrainte (2), on se ramène au problème de sac à dos quadratique (Pisinger (2007)). On en déduit donc que $(E-kQKP)$ est NP-difficile puisqu'il comprend deux sous-problèmes qui le sont

également.

À notre connaissance, $(E-kQKP)$ a très peu été étudié auparavant. Ce problème est une extension du problème du sac à dos linéaire avec contrainte de cardinalité (Caprara et al. (2000), Dudzinski (1989), Kellerer et al. (2004)). Ses applications couvrent celles du k-cluster et du sac à dos quadratique.

Nous montrons que ce problème ne peut pas être résolu optimalement en utilisant des solveurs standards dès 40 objets.

Nous introduisons ensuite une heuristique primale basique, puis nous proposons deux heuristiques duales qui produisent de bons majorants et minorants du problème en temps raisonnable, l'une basée sur une relaxation semi-définie et l'autre sur une relaxation agrégée.

Enfin, nous proposons une heuristique hybride qui combine les trois heuristiques précédentes. Elle intègre l'heuristique primale et l'heuristique duale agrégée dans la phase de réduction de la relaxation semi-définie.

De nombreux résultats expérimentaux, obtenus sur des instances générées aléatoirement, sont présentés et permettent de valider notre approche puisque notre heuristique hybride trouve la solution optimale dans 90% des cas et prouve l'optimalité dans 76% des cas et cela en 100 secondes en moyenne pour des instances de 50 à 200 objets.

Difficulté pratique du problème

Nous montrons dans cette section qu'en utilisant un solveur commercial comme CPLEX pour résoudre exactement notre problème, nous ne pouvons résoudre des instances de taille supérieure à 50 objets (sans reformulation préalable) voire 100 objets (en utilisant des reformulations quadratiques convexes (Billionnet et al. (2013; 2009))) en temps raisonnable (moins d'une heure).

Environnement de test Tous les tests ont été effectués sur un Intel Xeon biprocesseur dual core 3 GHz avec 2 Go de RAM (en utilisant les 4 cœurs quand CPLEX est utilisé, sinon 1 cœur).
Nous présenterons des résultats moyens sur 10 instances. Pour $n \in \{10,20,30,\ldots,200\}$, nous générons $k \in [1,n/4]$, $b \in [50,30k]$, et a_j, $c_{ij} \in [1,100]$. Ces intervalles ont été empiriquement choisis afin d'obtenir des instances difficiles mais qui comportent au moins une solution réalisable.
CPLEX a été utilisé dans sa version 12.1 avec son paramétrage par défaut. Pour résoudre les programmes semi-définis, nous avons utilisé CSDP, qui est intégré à COIN-OR, et qui applique une méthode de points intérieurs développée par Borchers (1999).
Dans tous les tableaux :

δ représente la densité de la matrice quadratique,

les temps CPU sont donnés en secondes,

les sauts de dualité (*gap*) pour chaque majorant (*upper bound*) sont définis comme étant $\frac{upperbound-opt}{opt} \times 100$, avec opt la valeur optimale du problème (si elle est connue), ou le meilleur minorant connu si la valeur optimale est inconnue,

de même, les sauts de dualité (*gap*) pour chaque minorant (*lower bound*) sont

définis comme étant $\frac{opt-lowerbound}{opt} \times 100$, avec opt la valeur optimale du problème (si elle est connue), ou le meilleur majorant connu si la valeur optimale est inconnue.

Tests avec CPLEX 12.1 Les versions actuelles de CPLEX permettent de convexifier automatiquement un problème non convexe en variables $0-1$, il est donc possible d'utiliser directement CPLEX sans reformulation préalable pour résoudre notre problème. Les résultats des tableaux 3.40 et 3.41 permettent de remarquer les très grands sauts de dualité. Les nombres entre parenthèses représentent le nombre d'instances non résolues dans le temps limite imparti (une heure) sur les 10 instances. On peut ainsi en conclure que CPLEX ne permet pas de résoudre en moins d'une heure certaines instances de 50 objets, quelle que soit la densité.

	δ = 25 %		δ = 50 %	
n	Gap %	CPU time (s)	Gap %	CPU time (s)
10	174.24	0.01	251.67	0.01
20	154.87	0.02	316.22	0.07
30	159.93	0.27	293.30	0.77
40	135.56	2.36	295.09	25.68
50	157.28	245.48	214.56	>3600 (1)

TABLE 3.40 – *Résolution exacte avec CPLEX 12.1 pour de faibles densités*

	δ = 75 %		δ = 100 %	
n	Gap %	CPU time (s)	Gap %	CPU time (s)
10	255.52	0.02	321.09	0.02
20	311.45	0.23	353.79	0.58
30	306.56	9.01	725.26	21.36
40	364.78	>3600 (1)	414.04	>3600 (1)
50	752.33	>3600 (1)	726.20	>3600 (2)

TABLE 3.41 – *Résolution exacte avec CPLEX 12.1 pour de fortes densités*

Reformulation quadratique convexe Les travaux de Billionnet et al. (2009) et Billionnet et al. (2013) ont permis de montrer l'intérêt d'une convexification adéquate d'un problème quadratique. En effet, cela améliore nettement les performances des solveurs.

La méthode QCR (Quadratic Convex Reformulation) La méthode consiste en une reformulaion d'un problème quadratique (non convexe) en variables $0-1$ en un problème équivalent en variables $0-1$ avec une fonction quadratique concave (dans le cas d'une maximisation), qui peut ensuite être donné en entrée à un solveur standard utilisant une procédure de séparation et d'évaluation progressive qui utilise la relaxation continue du nouveau problème. Un des objectifs est donc de proposer une reformulation dont la borne fournie par la relaxation continue sera la plus serrée possible.

QCR est une méthode en deux phases :

- **Phase 1**
  On remplace la fonction objectif $f(x)$ par une fonction objectif quadratique concave $f_{\alpha,u}(x)$ qui dépend de deux paramètres α et u appartenant à $\mathbb{R}^n$ pour obtenir un problème convexe en 0-1 équivalent.
- **Phase 2**
  On applique un solveur de problèmes quadratiques convexes en 0-1 au nouveau problème.

La première phase consiste à perturber la fonction objectif en soustrayant deux fonctions, nulles sur le domaine réalisable et qui dépendent de deux paramètres.

Billionnet et al. (2009) remplacent $f(x)$ par :

$$f_{u,\alpha}(x) = f(x) - \sum_{i=1}^{n} u_i \left(x_i^2 - x_i\right) - \sum_{i=1}^{n} \alpha_i x_i \left(\sum_{j=1}^{n} x_j - k\right).$$

avec $u \in \mathbb{R}^n$ et $\alpha \in \mathbb{R}^n$.

D'après un résultat de Faye et Roupin (2007), Billionnet et al. (2013) proposent de remplacer $f(x)$ par :

$$f_{u,v}(x) = f(x) - \sum_{i=1}^{n} u_i \left(x_i^2 - x_i\right) - v \left(\sum_{j=1}^{n} x_j - k\right)^2$$

avec $u \in \mathbb{R}^n$ et $v \in \mathbb{R}$.

L'objectif est alors de déterminer les meilleures valeurs des paramètres $u^* \in \mathbb{R}^n$ et $v^* \in \mathbb{R}$ tels que la fonction $f_{u^*,v^*}(x)$ soit concave et que le majorant obtenu via la relaxation continue soit minimal, ie :

$$(P) \quad \min_{u\in\mathbb{R}^n,\, v\in\mathbb{R}} \quad \max_{x\in[0,1]^n :\ \sum_{j=1}^n a_j x_j \le b,\ \ \sum_{j=1}^n x_j = k} \quad f_{u,v}(x)$$

Pour obtenir ces paramètres optimaux (Billionnet et al. (2013)), il suffit de résoudre la relaxation semi-définie de $(E-kQKP)$ suivante :

$$(E-kQKP_{SDP}) \begin{cases} \max & \sum_{i=1}^n \sum_{j=1}^n c_{ij} X_{ij} & & \\ \text{s.c.} & X_{ii} = x_i & i = 1,\dots,n & (3) \\ & \sum_{i=1}^{n}\sum_{j=1}^{n} X_{ij} - 2k\sum_{j=1}^{n} x_j = -k^2 & & (4) \\ & \sum_{j=1}^n a_j x_j \le b & & \\ & \sum_{j=1}^n x_j = k & & \\ & \begin{pmatrix} 1 & x^t \\ x & X \end{pmatrix} \succeq 0 & & \\ & x \in \mathbb{R}^n,\ X \in R^{n\times n} & & \end{cases}$$

où les contraintes (4) ont été introduites pour améliorer la valeur optimale du problème.

Les valeurs optimales u_i^* $(i = 1, \ldots, n)$ (resp. v^*) de (P) sont données par les valeurs optimales des variables duales des contraintes (3) (resp. (4)) de $(E - kQKP_{SDP})$.

Résultats numériques Nous sommes maintenant en mesure de résoudre des instances contenant jusqu'à 90 objets en utilisant la reformulation QCR et CPLEX 12.1.

	δ = 25 %		δ = 50 %	
n	Gap %	CPU time (s)	Gap %	CPU time (s)
50	41.22	0.69	36.38	0.87
60	149.24	0.58	21.65	1.65
70	46.84	4.08	80.44	11.57
80	42.06	17.38	64.10	40.41
90	129.34	81.72	92.22	145.19
100	82.78	183.17	21.83	>3600 (1)
110	84.61	>3600 (2)	20.49	>3600 (3)

TABLE 3.42 – *Résolution exacte avec CPLEX 12.1 via QCR pour de faibles densités*

	δ = 75 %		δ = 100 %	
n	Gap %	CPU time (s)	Gap %	CPU time (s)
50	114.26	0.52	69.45	1.66
60	150.84	3.91	69.06	3.90
70	63.77	22.96	66.18	31.18
80	92.31	101.41	49.19	240.64
90	42.12	>3600 (1)	29.93	376.36
100	27.22	>3600 (4)	106.57	>3600 (2)
110	148.16	>3600 (3)	24.24	>3600 (5)

TABLE 3.43 – *Résolution exacte avec CPLEX 12.1 via QCR pour de fortes densités*

Une heuristique primale

Nous présentons dans cette section une heuristique primale, H_{pri}, pour $(E - kQKP)$ qui est une adaptation d'une heuristique classique pour le sac à dos quadratique (Billionnet et Calmels (1996)).

Cette approche comprend une phase destructive telle que le nombre d'objets insérés soit inférieur ou égal à k et une phase constructive, qui englobe un algorithme glouton et une recherche locale, qui permet d'améliorer la solution et de satisfaire les deux contraintes.

Cette heuristique est très rapide (moins de 0.04 secondes) et les minorants obtenus sont de bonne qualité (voir la figure 3.17 et le tableau 3.44).

Une heuristique basée sur une relaxation semi-définie

Chaque étape de cette heuristique duale, H_{sdp}, basée sur la programmation semi-définie, consiste à fixer des variables à zéro dans une solution fournie par une relaxation semi-définie du problème avant d'utiliser l'heuristique primale pour réduire le problème.

/*Phase initiale : fournit une solution triviale $\underline{x}^0$ telle que $a\underline{x}^0 \leq b$ et $e\underline{x}^0 = k$*/
$J \leftarrow arg\ k\ smallest\ (a_j)_{j\in\{1,\ldots,n\}}$
for j **from** 1 **to** n **do**
  if $j \in J$ **then** $\underline{x}^0_j \leftarrow 1$ **else** $\underline{x}^0_j \leftarrow 0$ **endif**
enddo
$\underline{x}^{pri} \leftarrow \underline{x}^0\,; v(H_{pri}) \leftarrow f(\underline{x}^0)$
/*Phase destructive : fournit une solution $\underline{x}^1$ telle que $a\underline{x}^1 \leq b$ et $e\underline{x}^1 \leq k$*/
$\underline{x}^1 \leftarrow e = \{1,\ldots,1\}$
$U \leftarrow \{1,\ldots,n\}$
while $\sum_{j=1}^n a_j > b$ **or** $card(U) > k$ **do**
  $i^* \leftarrow arg\ min(\sum_{j\in U} c_{ij}/a_i)$.
  $\underline{x}^1_{i^*} \leftarrow 0$
  $U \leftarrow U - \{i^*\}$
endwhile
if $e\underline{x}^1 = k$ **and** $f(\underline{x}^1) > v(H_{pri})$ **then** $\underline{x}^{pri} \leftarrow \underline{x}^1\,; v(H_{pri}) \leftarrow f(\underline{x}^1)$ **endif**
/*Phase constructive : remplissage et échange afin d'améliorer $\underline{x}^1$ */
$\underline{x}^2 \leftarrow \underline{x}^1$
$bestgain \leftarrow 1$
while $bestgain$ **do**
  $bestgain \leftarrow 0; fillup \leftarrow false; exchange \leftarrow false$
  for all variable x_i telle que $\underline{x}^2_i = 0$ **do**
    if $e\underline{x}^2 < k$ **and** $a\underline{x}^2 + a_i \leq b$ **then**
      $gain \leftarrow \delta f_i$ /*dérivée première de $f = c_i + \sum_{j=1}^n c_{ij}\underline{x}^2_j$*/
      if $gain > bestgain$ **then**
        $bestgain \leftarrow gain; i^{fill} \leftarrow i, fillup \leftarrow true$
      endif
    else /*$e\underline{x}^2 = k$*/
      for all variable x_j telle que $\underline{x}^2_j = 1$ **do**
        if $a\underline{x}^2 - a_j + a_i \leq b$ **then**
          $gain \leftarrow \delta f_i - \delta f_j - c_{ij} - c_{ji}$ /*dérivée seconde de f*/
          if $gain > bestgain$ **then**
            $bestgain \leftarrow gain; (i^{ex}, j^{ex}) \leftarrow (i,j); exchange \leftarrow true$
          endif
        endif
      endfor
    endif
  endfor
  if $exchange$ **then**
    $\underline{x}^2_{i^{ex}} \leftarrow 1\,; \underline{x}^2_{j^{ex}} \leftarrow 0$
  else
    if $fillup$ **then** $\underline{x}^2_{i^{fill}} \leftarrow 1$ **endif**
  endif
endwhile
if $e\underline{x}^2 = k$ **and** $f(\underline{x}^2) > v(H_{pri})$ **then** $\underline{x}^{pri} \leftarrow \underline{x}^2\,; v(H_{pri}) \leftarrow f(\underline{x}^2)$ **endif**

FIGURE 3.16 – *Heuristique primale* H_{pri}

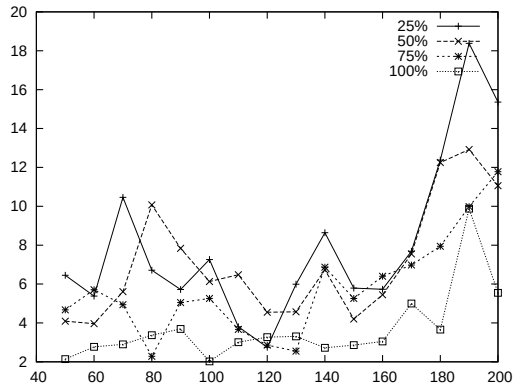


FIGURE 3.17 – *Heuristique H_{pri} : gaps pour chaque densité de la matrice C*

Cette heuristique duale itérative consiste à résoudre la relaxation semi-définie $(E - kQKP_{SDP})$ à chaque itération pour ensuite fixer des variables de la solution x^{sdp} à zéro. L'heuristique primale H_{pri} est ensuite utilisée pour réduire le problème. Nous tentons ensuite d'améliorer, à chaque itération, la solution en appliquant la procédure de remplissage et d'échanges sur le problème global (non réduit) (voir Figure 3.18).

Nous pouvons constater une amélioration importante de la qualité des résultats obtenus par H_{sdp} (voir les tableaux 3.44, 3.45 et 3.46), de l'ordre de trois fois meilleurs, avec des temps de calcul d'au plus 4 secondes en moyenne.

	Gaps %	
δ	H_{pri}	H_{sdp}
25%	8.08	1.69
50%	7.09	1.67
75%	5.76	1.64
100%	3.70	1.04
Global results	6.16	1.51

TABLE 3.44 – *H_{sdp} vs H_{pri} : Gaps*

Une heuristique basée sur une relaxation agrégée de $(E - kQKP)$

L'heuristique H_{sur}, basée sur une relaxation agrégée, permet d'obtenir de très bonnes solutions et de prouver leur optimalité dans 46% des cas en une minute en moyenne.

$x^{sdp} \leftarrow$ solution de la relaxation semi-définie $(E - kQKP_{SDP})$
$\epsilon \leftarrow 0$
$n_{reduce} \leftarrow n$
$\underline{x}^{sdp} \leftarrow 0$
while $n_{reduce} \geq 1$ **do**
  repeat
    $\epsilon \leftarrow \epsilon + 0.01$
  until au moins une variable de x^{sdp} soit inférieure à ϵ
  Mise à zéro des variables de x^{sdp} inférieures à ϵ
  Mettre à jour n_{reduce}
  Appliquer l'heuristique H_{pri} sur le *problème réduit*
  Mettre à jour $\underline{x}^{pri}$ en appliquant la procédure de remplissage et d'échanges sur le *problème global*
  if $f(\underline{x}^{pri}) > f(\underline{x}^{sdp})$
  then
    $\underline{x}^{sdp} \leftarrow \underline{x}^{pri}$
  endif
endwhile
$v(H_{sdp}) \leftarrow f(\underline{x}^{sdp})$

FIGURE 3.18 – *Heuristique* H_{sdp}

Relaxation agrégée Nous effectuons une double relaxation : nous transformons en premier lieu la contrainte de cardinalité (2) en une inégalité $(2')$:

$$(kQKP) \begin{cases} \max & f(x) & \\ \text{s.c.} & ax \leq b & (1) \\ & ex \leq k & (2') \\ & x \in \{0,1\}^n & \end{cases}$$

Ensuite, nous effectuons une relaxation agrégée de ce problème en combinant les contraintes (1) et $(2')$.

Afin de résoudre cette relaxation, nous adaptons l'algorithme $SADE^2$ Fréville et Plateau (1993) qui effectue une recherche dichotomique sur l'intervalle [0,1], revisitant ainsi la méthode de Glover Glover (1965) pour garantir l'optimalité en un nombre fini d'itérations.

Étant donné un multiplicateur $\mu \in [0,1]$, nous déterminons un ordre sur les contraintes (1) et (2') afin de réécrire la relaxation :

$$(S(\mu)) \begin{cases} \max & f(x) \\ \text{s.c.} & (A_1 + \mu A_2)x \leq (b_1 + \mu b_2) \\ & x \in \{0,1\}^n \end{cases}$$

avec soit

$$(A_1, b_1) = (a, b) \quad \text{et} \quad (A_2, b_2) = (e, k)$$

soit

$$(A_1, b_1) = (e, k) \quad \text{et} \quad (A_2, b_2) = (a, b)$$

Pour un μ donné, $x^*(\mu)$ est la solution optimale du problème $(S(\mu))$, si $x^*(\mu)$ satisfait les deux contraintes, alors c'est aussi une solution optimale de $(kQKP)$. Si de plus, $ax^*(\mu) = k$, le problème $(E - kQKP)$ est résolu. Sinon, nous effectuons une recherche dichotomique en utilisant le fait que $x^*(\mu)$ satisfait la

première (resp. la seconde) contrainte et viole la seconde (resp. la première) et en exploitant le ratio $(b_1 - A_1x^*(\mu))/(A_2x^*(\mu) - b_2)$ pour mettre à jour les bornes de l'intervalle (Fréville et Plateau (1993)).

À chaque itération de la procédure, nous résolvons le problème de sac à dos quadratique obtenu via la méthode de Caprara et al. (1999) en l'adaptant à des données réelles.

Afin de contrôler le temps de calcul, nous recherchons une valeur approchée de la valeur du dual (S) de $(kQKP)$:

-approximation 1 (résolution de $(S(\mu))$) :
Nous mettons un temps limite (une minute ici) pour le branch-and-bound de la méthode de Caprara et al. (1999). Ce critère d'arrêt n'intervient que quand leur relaxation Lagrangienne est inefficace, i.e. quand μ est proche de zéro ou pour les grandes instances. Dans ce cas, nous récupérons le majorant $v^{up}(S(\mu))$ de la valeur de $(S(\mu))$, qui est égal à la valeur maximale des nœuds pendants de l'arbre, ainsi que la solution associée $x^{up}(\mu)$ de $(S(\mu))$.

-approximation 2 (recherche dichotomique) :
La méthode de Caprara et al. (1999) n'est pas efficace pour des problèmes du type $(S(0))$, i.e. lorsque tous les coefficients sont identiques (égaux à 1 ici). Quand aucune solution n'est trouvée dans le temps imparti, nous augmentons μ pas à pas à partir de zéro (i.e. nous débutons avec la valeur 0.01 puis nous la doublons à chaque pas). Ce processus est arrêté dès que μ^t permet de trouver une solution optimale de $(S(\mu^t))$ en respectant le temps limite.

Avec l'*approximation 2*, l'ensemble M des multiplicateurs peut ne pas contenir le multiplicateur optimal μ^* pour (S), alors qu'avec l'*approximation 1*, notre algorithme fournit un majorant $\overline{v}(S)$ de $v(S)$ défini par :

$$\overline{v}(S) = min_{\mu \in M}\overline{v}(S(\mu))$$

où $\overline{v}(S(\mu))$ est égal à $v^{up}(S(\mu))$ si l'*approximation 1* est utilisée, ou à $v(S(\mu))$ sinon.

Proposition 2.

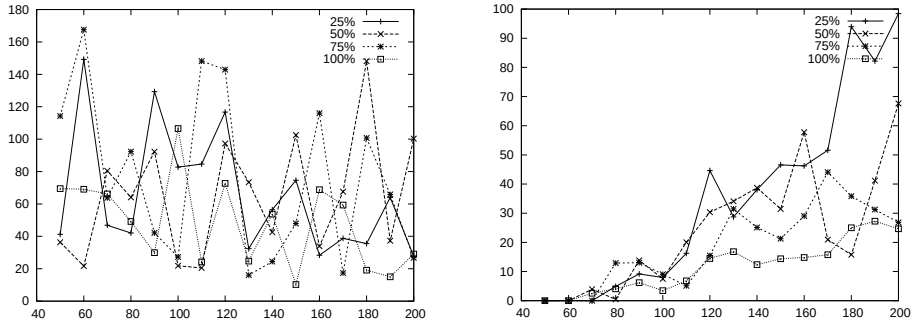
$$v(E-kQKP) \leq v(kQKP) \leq v(S) = \min_{\mu \in [0,1]} v(S(\mu)) \leq \quad \overline{v}(S) = \min_{\mu \in M} \overline{v}(S(\mu))$$

Enfin, lorsque nous utilisons l'*approximation 1*, la résolution de $(S(\mu))$ fournit une solution $x^{up}(\mu)$ qui peut être différente de $x^*(\mu)$. Nous noterons $\overline{x}(\mu)$ cette solution, qui est égale à $x^{up}(\mu)$ si l'*approximation 1* est utilisée ou à $x^*(\mu)$ sinon.

Résultats

Comme pour le cas linéaire, le nombre d'itérations pour converger est faible (au plus 10 itérations ici). Les temps de calcul sont plus importants mais restent

maîtrisés (pas plus de 7 minutes en moyenne) et les majorants sont de très bonne qualité (Figure 3.19 (b)) puisque plus de 75% des gaps sont améliorés par rapport à ceux de la relaxation semi-définie.



(a) Convexification via QCR (b) Dual agrégé

FIGURE 3.19 – *Gaps pour chaque densité de la matrice C*

L'heuristique duale agrégée H_{sur} Notre heuristique duale, H_{sur}, conserve les meilleures solutions $\underline{x}^{sur}$ de $(E-kQKP)$ produites par la recherche dichotomique de la figure 3.20. Elle fournit à la fois un minorant $f(\underline{x}^{sur})$ et un majorant $\overline{v}(S)$ de $v(E-kQKP)$.
À chaque itération, i.e. pour chaque multiplicateur μ de l'ensemble M, nous sauvegardons la meilleure solution réalisable $\underline{x}(\mu)$, fournie par le branch-and-bound, qui satisfait exactement la contrainte de cardinalité (i.e. $a\underline{x}(\mu) \leq b$ et $e\underline{x}(\mu) = k$). Comme nous débutons par l'heuristique H_{pri}, la solution $\underline{x}^{sur}$ produite par H_{sur} est telle que :

$$f(\underline{x}^{sur}) = \max\{f(\underline{x}^{pri}), \max_{\mu \in M} f(\underline{x}(\mu))\}$$

Nous avons ici réduit à 30 secondes le temps limite de résolution de $(S(\mu))$. Cela permet d'accélérer la procédure avec une détérioration négligeable du gap.

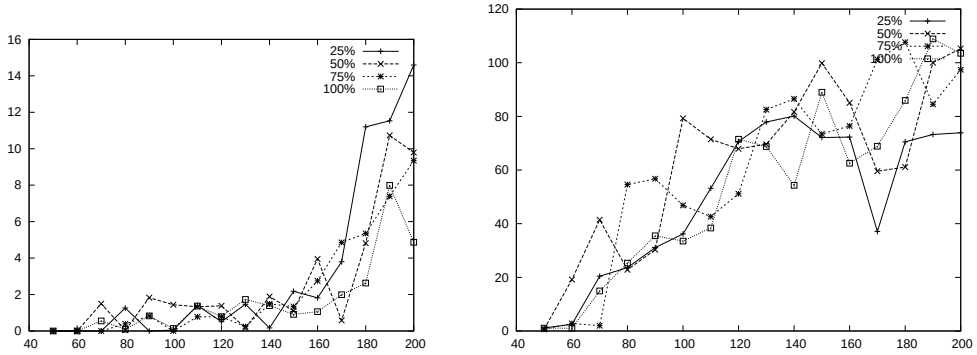
Résultats expérimentaux
Les résultats de la figure 3.21 (et des tableaux 3.45 et 3.46) montrent la qualité des minorants fournis par H_{sur}. 46% des instances sont résolues exactement par cette heuristique duale. Les temps de calculs n'excèdent pas deux minutes et le temps moyen est d'environ une minute.

Heuristique duale hybride

Notre heuristique hybride H_{hybrid} combine les heuristiques précédentes. Elle intègre l'heuristique primale et l'heuristique agrégée dans la phase de réduction de la relaxation semi définie.

Appliquer l'heuristique H_{pri} ; $\underline{x}^{sur} \leftarrow \underline{x}^{pri}$
/*Intervalle de confiance pour la recherche dichotomique*/
$(A_1, b_1) \leftarrow (a, b)$; $(A_2, b_2) \leftarrow (e, k)$; $end \leftarrow false$
Résolution de $(S(1)) : max\ f(x)\ \ s.t.\ \ (a+e)x \leq b+k;\ \ x \in \{0,1\}^n$ /*solution associée $\overline{x}(1)$*/
if $f(\underline{x}(1)) > f(\underline{x}^{sur})$ **then** $\underline{x}^{sur} \leftarrow \underline{x}(1)$ **endif**
if $\overline{x}(1)$ est réalisable
then $v(kQKP) \leftarrow f(\overline{x}(1))$ /*$\overline{v}(S) = v(kQKP)$*/
  if $e\overline{x}(1) = k$ **then** $\underline{x}^{sur} = \overline{x}(1)$ est optimale pour $(E - kQKP)$; $end \leftarrow true$ **endif**
else $\alpha \leftarrow (b - a\overline{x}(1))/(e\overline{x}(1) - k)$
 if $a\overline{x}(1) > b$
 then
  Résolution de $(S(0)) : max\ f(x)\ \ s.t.\ \ ax \leq b;\ \ x \in \{0,1\}^n$ /*solution associée $\overline{x}(0)$*/
  if $f(\underline{x}(0)) > f(\underline{x}^{sur})$ **then** $\underline{x}^{sur} \leftarrow \underline{x}(0)$ **endif**
  if $e\overline{x}(0) \leq k$
  then $v(kQKP) \leftarrow f(\overline{x}(0))$ /*$\overline{v}(S) = v(kQKP)$*/
    if $e\overline{x}(0) = k$ **then** $\underline{x}^{sur} = \overline{x}(0)$ est optimale pour $(E - kQKP)$; $end \leftarrow true$ **endif**
  else $\alpha_l \leftarrow (b - a\overline{x}(0))/(e\overline{x}(0) - k)$; $\alpha_r \leftarrow \alpha$
  endif
 else /*$e\overline{x}(1) > k$*/
  Résolution de $(S(\infty)) : max\ f(x)\ \ s.t.\ \ ex \leq k;\ \ x \in \{0,1\}^n$ /*solution associée $\overline{x}(\infty)$*/
  if $f(\underline{x}(\infty)) > f(\underline{x}^{sur})$ **then** $\underline{x}^{sur} \leftarrow \underline{x}(\infty)$ **endif**
  if $a\overline{x}(\infty) \leq b$
  then $v(kQKP) \leftarrow f(\overline{x}(\infty))$ /*$\overline{v}(S) = v(kQKP)$*/
    if $e\overline{x}(\infty) = k$ **then** $\underline{x}^{sur} = \overline{x}(\infty)$ est optimale pour $(E - kQKP)$; $end \leftarrow true$ **endif**
  else $\alpha_l \leftarrow (e\overline{x}(0) - k)/(b - a\overline{x}(0))$; $\alpha_r \leftarrow 1/\alpha$
    $(A_1, b_1) \leftarrow (e, k)$; $(A_2, b_2) \leftarrow (a, b)$ /*Permutation des deux contraintes*/
  endif
 endif
endif
/*Dichotomie étendue*/
$\mu_l \leftarrow 0; \mu_r \leftarrow 0$
while *not end* **do**
 if $\alpha_l \geq \alpha_r$
 then $\overline{v}(S) \leftarrow min\{f(\overline{x}(\mu_l)), f(\overline{x}(\mu_r))\}$; $end \leftarrow true$
   if $f(\underline{x}^{sur}) = \overline{v}(S)$ **then** $\underline{x}^{sur}$ est optimale pour $(E - kQKP)$ **endif**
 else $\mu \leftarrow (\alpha_l + \alpha_r)/2$; *Résolution de* $(S(\mu))$ /*solution associée $\overline{x}(\mu)$*/
  if $f(\underline{x}(\mu)) > f(\underline{x}^{sur})$ **then** $\underline{x}^{sur} \leftarrow \underline{x}(\mu)$ **endif**
  if $\overline{x}(\mu)$ is réalisable
  then $v(kQKP) \leftarrow f(\overline{x}(\mu))$ /*$\overline{v}(S) = v(kQKP)$*/
    if $e\overline{x}(\mu) = k$ **then** $\underline{x}^{sur} = \overline{x}(\mu)$ est optimale pour $(E - kQKP)$; $end \leftarrow true$ **endif**
  else $\alpha \leftarrow (b_1 - A_1\overline{x}(\mu))/(A_2\overline{x}(\mu) - b_2)$
   if $A_2\overline{x}(\mu) > b_2$ **then** $\alpha_l \leftarrow \alpha; \mu_l \leftarrow \mu$ **else** $\alpha_r \leftarrow \alpha; \mu_r \leftarrow \mu$ /*$A_1\overline{x}(\mu) > b_1$*/ **endif**
  endif
 endif
endwhile
$v(H_{sur}) \leftarrow f(\underline{x}^{sur})$

FIGURE 3.20 – *Heuristique* H_{sur}



(a) Gaps (b) Temps CPU (s)

FIGURE 3.21 – *Heuristique H_{sur} (résultats pour chaque densité de la matrice C)*

L'heuristique H_{hybrid} L'heuristique (figure 3.22) applique H_{sur}, qui inclut H_{pri}, puis applique H_{sdp} dans laquelle H_{sur} est utilisée une seconde fois. Nous obtenons ainsi une solution $\underline{x}^{hybrid}$ au moins aussi bonne que celles fournies par H_{pri}, H_{sdp} et H_{sur}. De plus, les temps de calcul restent raisonnables.

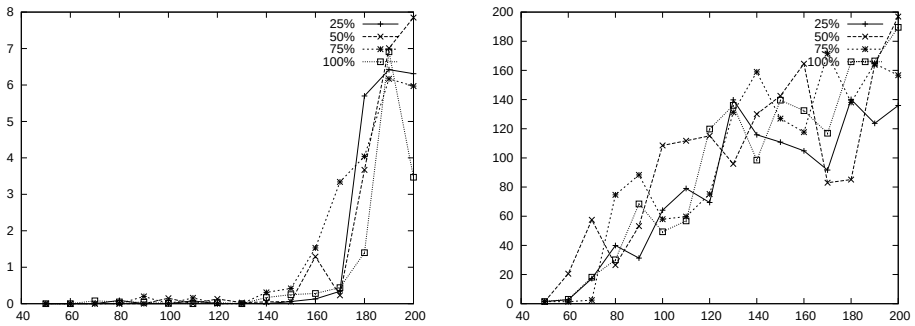
Résultats Le gap moyen de l'heuristique hybride est d'environ 1.20 % avec un temps CPU moyen autour d'une minute trente.
La figure 3.23 et les tableaux 3.45, 3.46 et 3.47 résument les performances de l'heuristique H_{hybrid}. Nous avons de plus appliqué CPLEX 12.1 sur chaque instance avec comme temps limite le temps mis par H_{hybrid} sur l'instance correspondante. Les résultats avec *CPLEX en temps limité*, noté $LT - CPLEX$, figurent dans les tableaux 3.45 et 3.46. La figure 3.24 (a) (gaps) montre clairement la supériorité de H_{hybrid} sur H_{sur} et $LT - CPLEX$.
Le tableau 3.47 compte le pourcentage d'instances pour lesquelles l'optimalité est prouvée (i.e. où minorant et majorant coïncident) ou simplement obtenue (550 instances sur les 640 ont une solution optimale connue). Les nombres entre paranthèses se réfèrent à ces 550 instances.
Pour l'ensemble des 550 instances (resp. 640 instances), l'heuristique H_{hybrid} obtient cette solution optimale dans 90% (resp. 76%) des cas et le prouve dans 63% (resp. 49%) des cas, en moins de 100 secondes en moyenne. Pour les instances comprenant de 50 à 100 objets, H_{hybrid} obtient une solution optimale dans 96% des cas (i.e. 232 sur 240 instances).
Enfin, la figure 3.24 (b) montre que les temps CPU de H_{hybrid} augmentent avec la taille des instances, mais n'excèdent jamais deux minutes trente en moyenne.

Conclusion

Cette section fournit des minorants et majorants pour le problème du sac à dos quadratique avec contrainte de cardinalité en égalité. Nous avons résolu des

Appliquer l'heuristique H_{sur}
/*la valeur de H_{sur} = $f(\underline{x}^{sur})$ est au moins égale à la valeur de H_{pri}=$f(\underline{x}^{pri}_{first})$ */
$\underline{x}^{hybrid} \leftarrow \underline{x}^{sur}$
$x^{sdp} \leftarrow$ solution de la relaxation semi-définie $(E - kQKP_{SDP})$ du problème
$\epsilon \leftarrow 0$
$n_{reduce} \leftarrow n$
while $n_{reduce} \geq 1$ **do**
  repeat
    $\epsilon \leftarrow \epsilon + 0.01$
  until une variable soit inférieure à ϵ dans x^{sdp}
  Supprimer les variables dont la valeur est inférieure à ϵ dans x^{sdp}
  Mettre à jour n_{reduce}
  Appliquer l'heuristique H_{pri} sur le *problème réduit*
  if ($\underline{x}^{hybrid}$ et $\underline{x}^{pri}$ ne sont pas meilleurs que $\underline{x}^{pri}_{first}$) et $\epsilon = 0.05$
  then /*L'heuristique H_{sur} est appliquée pour la seconde fois*/
    Appliquer l'heuristique H_{sur} sur le *problème réduit*
    if $f(\underline{x}^{sur}) > f(\underline{x}^{pri})$
    then
      $\underline{x}^{pri} \leftarrow \underline{x}^{sur}$
    endif
  endif
  Mettre à jour $\underline{x}^{pri}$ en appliquant la procédure de remplissage et d'échange sur le *problème global*
  if $f(\underline{x}^{pri}) > f(\underline{x}^{hybrid})$
  then
    $\underline{x}^{hybrid} \leftarrow \underline{x}^{pri}$
  endif
endwhile
$v(H_{hybrid}) \leftarrow f(\underline{x}^{hybrid})$

FIGURE 3.22 – *Heuristique H_{hybrid}*



(a) Gaps (b) Temps CPU (s)

FIGURE 3.23 – *Heuristique H_{hybrid} (résultats pour chaque densité de la matrice C)*

	Heuristique H_{sdp}		Heuristique H_{sur}		Heuristique H_{hybrid}		$LT-CPLEX$
δ	Gap %	CPU (s)	Gap %	CPU (s)	Gap %	CPU (s)	Gap %
25 %	0.60	1.55	0.22	19.15	**0.01**	26.14	1.62
50 %	0.43	1.70	0.80	32.28	**0.03**	44.64	2.11
75 %	0.22	1.46	0.20	27.27	**0.03**	37.71	2.28
100 %	0.35	1.51	0.27	18.55	**0.03**	28.43	1.27
Global results	0.40	1.56	0.37	25.06	**0.03**	34.23	1.82

TABLE 3.45 – *H_{hybrid} vs H_{sdp}, H_{sur} et $LT-CPLEX$: valeurs moyennes pour $n = 50$ à 100*

	Heuristique H_{sdp}		Heuristique H_{sur}		Heuristique H_{hybrid}		$LT-CPLEX$
δ	Gap %	CPU (s)	Gap %	CPU (s)	Gap %	CPU (s)	Gap %
25 %	2.42	6.91	4.95	68.10	**1.98**	111.11	3.99
50 %	2.41	6.12	3.57	80.16	**2.03**	128.87	4.41
75 %	2.51	6.14	3.43	80.36	**2.19**	130.05	4.15
100 %	1.45	6.66	2.47	75.17	**1.30**	132.19	2.65
Global	2.20	6.46	3.61	75.95	**1.88**	125.56	3.80

TABLE 3.46 – *H_{hybrid} vs H_{sdp}, H_{sur} et $LT-CPLEX$: valeurs moyennes pour $n = 110$ à 200*

			Optimalité	
δ	Gap %	CPU (s)	Prouvée %	Obtenue %
25 %	1.25	79.25	47.95 (58.82)	80.14 (92.13)
50 %	1.28	97.28	51.33 (65.25)	77.33 (89.92)
75 %	1.38	95.43	47.02 (61.74)	72.85 (88.00)
100 %	0.82	93.28	51.32 (66.67)	74.34 (88.28)
Global	1.18	91.31	49.42 (63.11)	76.13 (89.59)

TABLE 3.47 – *Heuristique H_{hybrid} : performance pour toutes les tailles*

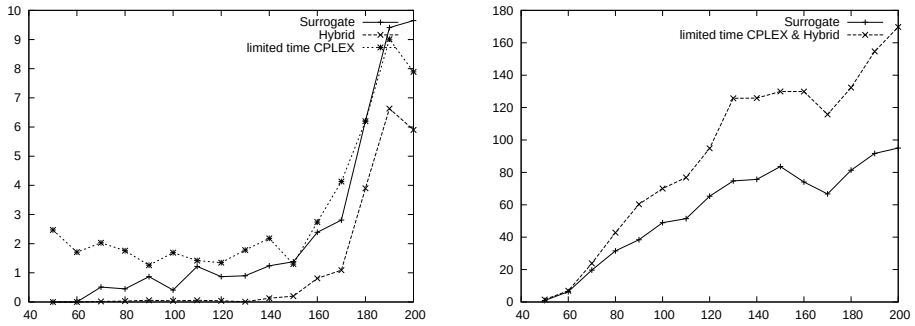
instances de 200 objets quelle que soit la densité de la matrice de la fonction objectif. Les résultats montrent l'efficacité de notre heuristique hybride qui permet d'obtenir la solution optimale (quand elle est connue) dans 90% des cas et de la prouver dans 76% des cas en 100 secondes en moyenne. Nous élaborons actuellement un branch and cut basé notamment sur des relaxations semi-définies et sur les heuristiques proposées dans cette section.

3.2.3 Convexification et application au problème du sac à dos quadratique avec contrainte de cardinalité

Dans cette section, nous montrons comment améliorer la méthode QCR et ses extensions pour des problèmes quadratiques en variables 0-1 et nous appliquons cette convexification au problème du sac à dos quadratique avec contrainte de cardinalité $(E-kQKP)$ étudié dans la section précédente.

Introduction

L'objectif ici est de présenter une algorithmique dédiée pour gérer le nombre de variables et de contraintes des méthodes de convexification tout en maîtrisant le



(a) Gaps (%) (b) Temps CPU (s)

FIGURE 3.24 – H_{hybrid} *vs* H_{sur} *et* $CPLEX$ *en temps limité*

temps de convexification. En effet, en partant du principe de QCR (Billionnet et al. (2009), Plateau (2008)), des travaux de Billionnet et al. (2013) et Ji et al. (2012) permettent d'améliorer la borne de la relaxation continue du problème convexifié. Nous appliquons cette approche au problème $(E-kQKP)$. La convexification, reprenant les travaux de Billionnet et al. (2013) et Ji et al. (2012), sera présentée et l'approche algorithmique et les résultats validant celle-ci seront ensuite donnés.

Convexification pour les problèmes quadratiques en 0-1

Nous reprenons les travaux de Billionnet et al. (2013) et Ji et al. (2012) pour construire une reformulation quadratique convexe. Nous effectuons une décomposition de la matrice de la fonction objectif et montrons que la programmation semi-définie permet d'obtenir une telle décomposition.

Soient P et N deux ensembles de matrices $n \times n$, respectivement positives et négatives, i.e.,

$$\mathcal{P} := P = (P_{ij}) \in \mathbb{R}^{n\times n} | P \geq 0, \mathcal{N} := N = (N_{ij}) \in \mathbb{R}^{n\times n} | N \leq 0.$$

Nous considérons la décomposition suivante de $f(x)$:

$$f(x) = x^T(C-M)x + x^T Mx,$$

où $C - M \preceq 0$ et $M = Diag(\rho) + P + N \in \mathcal{S}$ avec $\rho \in \mathbb{R}^n$, $P \in \mathcal{P}$ et $N \in \mathcal{N}$.

Pour tout $x_i, x_j \in \{0,1\}$, nous avons que

$$x_i x_j = min(x_i, x_j),\ x_i x_j = max(0, x_i + x_j - 1).$$

Ainsi, pour tout $x \in \{0,1\}^n$, nous pouvons réécrire $f(x)$ comme

$$\begin{aligned} f(x) &= x^T(C - Diag(\rho) - P - N)x + x^T(Diag(\rho) + P + N)x \\ &= x^T(C - Diag(\rho) - P - N)x + \rho^T x + \sum_{i,j=1}^{n}[P_{ij}x_i x_j + N_{ij}x_i x_j] \\ &= x^T(C - Diag(\rho) - P - N)x + \rho^T x + \sum_{i,j=1}^{n}[P_{ij}s_{ij} + N_{ij}t_{ij}] \end{aligned}$$

où $s_{ij} = min(x_i, x_j)$ et $t_{ij} = -max(0, x_i + x_j - 1)$.

Nous obtenons donc la reformulation équivalente :

$$\begin{array}{lll} \max & f(x) = x^T(C - Diag(\rho) - P - N)x + \rho^T x + \sum_{i,j=1}^{n}[P_{ij}s_{ij} + N_{ij}t_{ij}] & \\ \text{s.c.} & \sum_{j=1}^{n} a_j x_j \leq b & (1) \\ & \sum_{j=1}^{n} x_j = k & (2) \\ & s_{ij} = min(x_i, x_j), t_{ij} = -max(0, x_i + x_j - 1) \;\; i, j = 1, \ldots, n & \\ & x_j \in \{0, 1\} \quad j = 1, \ldots, n & \end{array}$$

où $C - Diag(\rho) - P - N \preceq 0$, $P \in \mathcal{P}$ et $N \in \mathcal{N}$. Comme $P_{ij} \geq 0$, la contrainte $s_{ij} = min(x_i, x_j)$ peut être relâchée pour obtenir les deux inégalités suivantes : $s_{ij} \leq x_i$ et $s_{ij} \leq x_j$ sans que la solution optimale n'en soit affectée. De même, nous pouvons relâcher $t_{ij} = -max(0, x_i + x_j - 1)$ en $t_{ij} \leq 0$ et $t_{ij} \leq 1 - x_i - x_j$.

La reformulation est donc équivalente au programme convexe suivant :

$$\begin{array}{lll} \max & f(x) = x^T(C - Diag(\rho) - P - N)x + \rho^T x + \sum_{i,j=1}^{n}[P_{ij}s_{ij} + N_{ij}t_{ij}] & \\ \text{s.c.} & \sum_{j=1}^{n} a_j x_j \leq b & (1) \\ & \sum_{j=1}^{n} x_j = k & (2) \\ & s_{ij} \leq x_i, \; s_{ij} \leq x_j \quad i, j = 1, \ldots, n & \\ & t_{ij} \leq 0, \; t_{ij} \leq 1 - x_i - x_j \;\; i, j = 1, \ldots, n & \\ & x_j \in \{0, 1\} \quad j = 1, \ldots, n & \end{array}$$

En combinant QCR avec cette reformulation, la fonction objectif peut alors être réécrite :

$$\begin{aligned} f(x, \rho, \alpha, P, N) = x^T[C - Diag(\rho) - (\alpha e^T + e\alpha^T) - P - N]x + (\rho + \\ k\alpha)^T x) + \sum_{i,j=1}^{n}[P_{ij}s_{ij} + N_{ij}t_{ij}] \end{aligned}$$

avec $C - Diag(\rho) - (\alpha e^T + e\alpha^T) - P - N \preceq 0$.

En remplaçant $f(x)$ par $f(x, \rho, \alpha, P, N)$ dans $(E - kQKP)$, nous obtenons une reformulation quadratique convexe de $(E - kQKP)$. Les paramètres optimaux

(ρ, α, P, N) peuvent être obtenus en résolvant la relaxation semi-définie suivante :

$$
(E-kQKP_{SDP2}) \left\{ \begin{array}{lll}
\max & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}X_{ij} & \\
\text{s.c.} & X_{ii} = x_i & i=1,\ldots,n \quad (3)\\
& -kx_i + \sum_{j=1}^{n} X_{ij} = 0 & i=1,\ldots,n \quad (4)\\
& \sum_{j=1}^{n} a_j x_j \leq b & \\
& \sum_{j=1}^{n} x_j = k & \\
& X_{ij} \leq x_i,\ X_{ij} \leq x_j & i,j=1,\ldots,n \quad (5)\\
& X_{ij} \geq x_i + x_j - 1,\ X_{ij} \geq 0 & i,j=1,\ldots,n \quad (6)\\
& \begin{pmatrix} 1 & x^t \\ x & X \end{pmatrix} \succeq 0 & \\
& x \in \mathbb{R}^n,\ X \in S_n &
\end{array} \right.
$$

où S_n représente l'ensemble des matrices réelles symétriques $n \times n$.

Mise en œuvre et résultats

La relaxation semi-définie $(E-kQKP_{SDP2})$ contient un grand nombre de contraintes et il est malheureusement impossible, même pour des instances de taille moyenne, de résoudre exactement de telles relaxations avec les solveurs actuels en des temps raisonnables (même quelques heures). Nous avons donc opté pour une stratégie itérative où nous résolvons en premier lieu la relaxation $(E-kQKP_{SDP})$, i.e. celle correspondant à QCR, puis nous détectons les contraintes (5) et (6) violées, nous les trions et nous n'ajoutons à la relaxation que les t plus violées, t dépendant de la taille du problème mais n'excédant pas quelques milliers (entre 1500 pour 50 objets et 3000 pour 200 objets pour $(E-kQKP)$ par exemple). Nous répétons ce processus, en gérant le pool de contraintes (suppression des contraintes inutiles, ajout de nouvelles), tant que nous ajoutons plus de r contraintes (r étant fixé à 30 pour ce problème). Empiriquement, nous remarquons que, pour $(E-kQKP)$, cette procédure n'itère jamais plus de 5 fois avant de se terminer. Nous pouvons remarquer que pour de tels problèmes en maximisation, très peu de contraintes (6) sont violées et que l'ajout de ces contraintes n'améliorent que très peu la valeur de la relaxation mais ralentissent du coup sa résolution ; nous avons donc décidé de ne pas considérer ces contraintes violées et les variables t_{ij} associées. Concernant les contraintes (5), il est inutile ici de gérer les deux séries de contraintes à la fois puisque la matrice X est symétrique. Nous n'avons donc géré ici que les contraintes $X_{ij} \leq x_i\ i,j=1,\ldots,n$ et les variables s_{ij} associées. Nous notons notre approche MQCR.

Résultats Les résultats pour $E-kQKP$ sont donnés dans le tableau 3.48 en utilisant les instances de Ji et al. (2012).

Les gaps correspondent aux gaps obtenus après résolution avec CPLEX en utilisant la convexification correspondante (QCR, Ji et al. (2012) et MQCR). On a ensuite le temps de résolution des relaxations semi-définies ; notons que pour MQCR, il s'agit du temps de résolution de la procédure itérative décrite ci-dessus, qui comprend donc plusieurs résolutions SDP. Enfin, le temps de résolution total

n	GAP QCR	Temps (s) $(E-kQKP_{SDP})$	Temps (s) QCR	GAP Ji et al. (2012)	Temps SDP (s) Ji et al. (2012)	Temps (s) Ji et al. (2012)	GAP MQCR	Temps (s) $(E-kQKP_{SDP2})$	Temps (s) MQCR
60	0	1,00	33.30	0	6,00	33,30	0	3,02	23,29
70	0	2,00	220,8	0	7,20	167,60	0	4,95	87,33
80	0	2,00	1106,80	0	9,40	751,00	0	8,60	297,92
90	4,30	2,20	1787,60	1,50	12,00	1697,80	0,64	12,01	1429,24
100	6,20	3,00	1800	5,00	15,20	1800	2,83	15,74	1800

TABLE 3.48 – *Résultats pour $E-kQKP$ ($k=\lfloor\frac{n}{8}\rfloor$, Temps limite=1800 sec)*

est indiqué pour chaque cas, le temps limite étant fixé à 30 minutes pour ces instances. Les convexifications Ji et al. (2012) et MQCR permettent bien d'améliorer QCR (le majorant obtenu est de meilleur qualité ce qui permet de mieux résoudre les instances, i.e. avec un temps inférieur et un gap inférieur). Les temps de résolution des SDP pour Ji et al. (2012) et MQCR sont comparables, mais nous obtenons de meilleurs résultats en moins de temps avec MQCR. Le code de Billionnet et al. (2013) n'étant pas disponible, nous n'avons pas effectué de comparaison avec cette approche qui utilise un branch and bound spécifique.

Conclusion

Nous avons proposé, dans cette section, une algorithmique permettant d'appliquer en pratique les dernières méthodes de convexification, et celle-ci a été validée expérimentalement.

CONCLUSION DU CHAPITRE

Nous nous sommes intéressés dans ce chapitre à des approches issues de la programmation mathématique afin d'élaborer des méthodes de résolution exactes et/ou approchées pour des problèmes difficiles. Nous avons considéré différentes relaxations et décompositions de modèles connus et nous avons introduit également le concept de réoptimisation. Nous nous sommes particulièrement focalisés sur la décomposition de Dantzig-Wolfe et sur le schéma de génération de colonnes, pour lequel nous avons proposé différentes techniques de diversification, nous avons aussi considéré le cas de la réoptimisation et étudié son intérêt dans le cadre de la génération de colonnes. Cela nous a permis notamment de proposer des méthodes de résolution pour des problèmes de tournées de véhicules et de localisation routage. Nous avons également considéré des problèmes de type sac à dos quadratique pour lesquels nous avons introduit des techniques de réoptimisation ainsi que des méthodes de résolution et des reformulations. Ces travaux permettent d'envisager de nombreuses perspectives, parmi lesquelles nous pouvons citer par exemple :

- l'élaboration d'approches de type "Branch and Cut and Price" intégrant des techniques de diversification et de réoptimisation pour la résolution de problèmes difficiles de localisation et routage ;
- l'extension du principe de réoptimisation locale à d'autres types de problèmes ;
- l'élaboration d'approches exactes de résolution pour des problèmes de type sac à dos quadratique avec différentes contraintes additionnelles.

CONCLUSION GÉNÉRALE 4

BILAN

Depuis ma nomination en tant que Maître de Conférences à l'université Paris 13, j'effectue mes recherches au sein de l'équipe AOC (Algorithmes et Optmisation Combinatoire) du LIPN (Laboratoire d'Informatique de Paris Nord). Mes travaux de recherche comportent plusieurs volets et s'articulent autour de la résolution de problèmes d'optimisation combinatoire en nombres entiers, avec des aspects théoriques, algorithmiques et applicatifs.

Au cours de ce mémoire, nous avons présenté différents travaux de recherche autour de la reformulation, des relaxations et de la réoptimisation.

Nous nous sommes notamment intéressés aux problèmes de type (multi-)flots pour lesquels nous avons proposé quelques résultats de complexité ainsi que des algorithmes polynomiaux dédiés dans le cas des graphes en anneaux. Nous avons également reformulé des problèmes complexes issus d'autres domaines (traitement d'images, fouille de données, bioinformatique) comme des problèmes de flots, ce qui nous a permis de proposer des approches de résolution innovantes et efficaces. Nous avons enfin présenté une approche alternative pour la programmation dynamique qui a permis de diminuer considérablement les opérations de dominance pour des problèmes de type plus court chemin avec contraintes de ressources.

Nous nous sommes ensuite intéressés à la résolution, via la programmation mathématique, des problèmes de type routage et sac à dos. Nous nous sommes appuyés sur la méthode de décomposition de Dantzig-Wolfe et sur la génération de colonnes. Nous avons notamment introduit la notion de diversification ainsi que la réoptimisation et nous avons proposé des approches exactes et approchées de résolution, en utilisant parfois des reformulations.

D'autres travaux effectués à l'université Paris 13 auraient pu également être présentés dans ce mémoire mais ne l'ont pas été par soucis de concision, d'homogénéité et de clarté. On peut citer notamment :

- une approche de résolution utilisant une décomposition de Danzig-Wolfe ainsi que des inégalités valides pour le problème du voyageur de commerce asymétrique avec contraintes de saut (Hop asymmetric Traveling Salesman Problem : HTSP) ;
- des méthodes pour la négociation en temps réel des fenêtres de temps dans le cadre du problème du voyageur de commerce ;

- une approche heuristique fondée sur la génération de colonnes pour la planification des arrêts des centrales nucléaires ;
- une métaheuristique hybride (recuit simulé et recherche à voisinages variables) parallèle pour le problème de réaffectation des processus à des machines ;
- une matheuristique pour la planification et le routage des personnels pour l'hospitalisation à domicile ;
- des relaxations compactes pour des problèmes de programmation polynomiale ;
- des approches de résolution exactes et approchées pour les problèmes de type affectation quadratique ;

PERSPECTIVES

Le large spectre des travaux présentés dans ce mémoire implique un grand nombre de perspectives. En plus des perspectives évoquées dans les conclusions des chapitres 2 et 3, comme la recherche d'autres topologies de graphes pour lesquelles les problèmes de multiflots et/ou de multicoupes seraient polynomiaux et la recherche d'algorithmes dédiés efficaces pour les résoudre ; l'élaboration de nouvelles approches de réduction et de résolution des problèmes de traitement d'images, comme la segmentation, dans le cas multi-labels (segmentation de plusieurs objets) qui induisent alors des problèmes de type "multiway cut" ; l'adaptation et le développement d'approches efficaces pour la recherche de communautés dans les réseaux sociaux ; l'extension de la programmation dynamique par blocs aux cas à plus de deux ressources ; l'élaboration d'approches de type "Branch and Cut and Price" intégrant des techniques de diversification et de réoptimisation pour la résolution de problèmes difficiles de localisation et routage ; l'extension du principe de réoptimisation locale à d'autres types de problèmes ; ou encore l'élaboration d'approches exactes de résolution pour des problèmes de type sac à dos quadratique avec différentes contraintes additionnelles ; nous pouvons ajouter les différents travaux en cours qui concernent :

- l'élaboration de branch and cut and price pour la résolution de problèmes particuliers de localisation et routage ;
- l'introduction de techniques de réoptimisation pour faire face à l'ajout et à la suppression de certains services de soins pour la planification et le routage des personnels pour l'hospitalisation à domicile ;
- la résolution exacte via une approche "additive bounding" utilisant la génération de colonnes pour un problème de type "inventory routing" ;
- une approche heuristique basée sur une double génération de colonnes pour la planification et le routage des trains en gare ;
- l'élaboration d'un branch and cut utilisant des relaxations semi-définies pour le problème du sac à dos quadratique avec contrainte de cardinalité.

Ces travaux ont permis également de mettre en exergue des perspectives à plus long terme et plus ambitieuses.

La première concerne le concept de réoptimisation locale qui pourrait être étendu en utilisant d'autres notions de distance pour modéliser la proximité locale et pourrait ainsi fournir un cadre théorique intéressant à la problématique de la

prise en compte de la modification partielle des données. Il serait également opportun dans ce cadre de collaborer avec des spécialistes d'apprentissage afin de développer des approches hybrides innovantes.

La deuxième concerne l'élaboration de méthodes de décomposition automatique. En effet, il existe depuis peu quelques tentatives pour développer des logiciels capables de reformuler de manière automatique les problèmes d'optimisation combinatoire, notamment via la génération de colonnes. Malgré ces quelques avancées, il reste encore plusieurs aspects à la fois théoriques et algorithmiques à explorer. Nous souhaitons en particulier aborder les différentes problématiques sous-jacentes suivantes qui sont encore partiellement voire complètement ouvertes :

- Un des points importants dans la reformulation concerne les techniques de réarrangement de la matrice des contraintes pour identifier et faire ressortir des structures classiques nécessaires pour appliquer une décomposition et/ou une relaxation.
- Un deuxième point connexe au précédent consiste à définir quelle reformulation permet d'appliquer la meilleure décomposition parmi celles possibles. Les principales sont la décompostion de Dantzig-Wolfe, la décomposition lagrangienne, la décomposition de Benders. Il faut donc d'abord identifier des critères selon lesquels comparer les reformulations : borne obtenue, temps de calcul, etc.
- Une fois le problème reformulé automatiquement, il faut considérer la phase de résolution. Encore une fois, un choix doit être opéré entre les algorithmes approchés (heuristiques) et les algorithmes exacts. Parmi ces derniers on peut distinguer ceux classiques de type branch-and-cut-and-price et la méthode fondée sur "l'additive bounding".

La troisième perspective à long terme concerne l'élaboration de nouvelles méthodes de convexification pour les programmes non linéaires mixtes en utilisant l'éventail des linéarisations existantes. En effet, les dernières approches de convexification proposées utilisent en fait les contraintes de linéarisation classiques dans la phase de convexification, ce qui implique des variables et des contraintes additionnelles dans le problème convexifié. Il semble donc judicieux d'étudier la possibilité d'utiliser d'autres linéarisations connues pour proposer de nouvelles approches de convexification. Comme pour les techniques de linéarisation, il conviendra de limiter le nombre de variables et de contraintes ajoutées afin de maîtriser les temps de calcul.

Enfin la dernière perspective à long terme concerne la généralisation de la décomposition de Dantzig-Wolfe et de la génération de colonnes pour la programmation non linéaire (non convexe) mixte. La reformulation de Dantzig-Wolfe peut être généralisée au cas non linéaire convexe mais ne peut pas s'appliquer dans le cas non convexe. Dans un premier temps, nous pourrions nous intéresser aux méthodes de convexification qui pourraient alors être utilisées comme une phase de prétraitement. L'efficacité des méthodes de convexification réside notamment dans le fait que le problème convexifié fournit une bonne relaxation continue. Cette relaxation pourrait encore être améliorée en appliquant une reformulation de Dantzig-Wolfe. Une des questions est de déterminer comment reconvexifier les termes quadratiques de la fonction objectif après l'ajout d'une ou de plusieurs variables. Même

si théoriquement il est possible d'appliquer les méthodes de convexification après chaque ajout de variables, au détriment du temps d'exécution, il serait plutôt intéressant de proposer des techniques ad-hoc pour reconvexifier la fonction objectif. Dans un deuxième temps, nous souhaiterions considérer la possibilité d'étendre le principe de la décomposition de Dantzig-Wolfe aux problèmes non linéaires non convexes, en faisant tout d'abord le lien avec la décomposition simpliciale et la décomposition primale. Nous pourrions ensuite par exemple envisager de générer, dans le cas de la programmation quadratique en variables binaires, des matrices et non plus des colonnes. Cela suppose une étude sur la décomposition matricielle de tels problèmes et la résolution, via des approches de type semi-définies ou similaires, des sous-problèmes afin de générer ces nouvelles matrices à intégrer dans le problème maître issu de la décomposition.

ANNEXES

A

SOMMAIRE

CURRICULUM-VITAE

A.1 ÉTAT CIVIL ET DIPLÔMES

Nom	LÉTOCART
Prénom	Lucas
Date et lieu de naissance	16 juin 1976 à Montmorency
Nationalité	Française
Situation familiale	Marié, 2 enfants
Adresse personnelle	6 allée des tilleuls, 95600 Eaubonne, France
Téléphone	06 20 61 37 29
Adresse professionnelle	LIPN - Université Paris 13, Sorbonne Paris Cité 99, avenue J-B. Clément 93430 Villetaneuse
Téléphone	01 49 40 35 95
Fax	01 48 26 07 12
E-mail	lucas.letocart@lipn.univ-paris13.fr
Web	`http://www-lipn.univ-paris13.fr/~letocart`

Fonction actuelle	Maître de Conférences à l'université Paris 13 depuis le 1er septembre 2003 Titulaire de la prime d'excellence scientifique (PES) depuis octobre 2012
Fonctions antérieures	Conservatoire National des Arts et Métiers, Paris (France) . 2002 - 2003 : ATER en Informatique . 1999 - 2002 : Allocataire de recherche - Moniteur

1999 : **DEA** (Informatique et Recherche Opérationnelle)
Université Pierre et Marie Curie Paris 6 - CNAM (France)
Mention Très Bien (Major)
Stage : Différenciation des routes aériennes. Eurocontrol - Brétigny-sur-Orge (France)

2002 : **Doctorat** en Informatique (soutenu le 19 décembre 2002)

Sujet	Optimisation combinatoire
Titre	Problèmes de multicoupes minimales et de multiflots maximaux en nombres entiers
Mention	Très honorable
Établissement	Conservatoire National des arts et Métiers, Paris (France)
Membres du jury	M-C. Costa (Directrice de thèse) et F. Roupin (co-encadrant) A. Billionnet (Président), M. Minoux et G. Plateau (Rapporteurs), V. Gabrel et A. Lisser (Examinateurs)

A.2 ENSEIGNEMENT

A.2.1 Conservatoire National des Arts et Métiers : 1999 - 2003

Moniteur de l'enseignement supérieur en informatique (64h/an de 1999 à 2002)
ATER (Attaché Temporaire d'Enseignement et de Recherche) en informatique (192h en 2002-2003)

Cycle probatoire

- **VARI** : Valeur d'Accueil et de Reconversion à l'Informatique.
  (Système et architecture des machines, programmation (en Ada), structures de données, algorithmique, théorie des graphes et complexité)
  - CM : 40h en 2002-2003.
  - TD : 20h/an de 1999 à 2002 et 45h en 2002-2003.
  - TP : 45h en 1999-2000 et 25h/an de 2000 à 2002.
- **CDI** : Conception et Développement Informatique.
  - TD : 15h/an de 2000 à 2002 et 30h en 2002-2003.
  - TP : 5h/an de 2000 à 2002 et 10h en 2002-2003.
- **MOCA** : Modélisation, Optimisation, Complexité et Algorithmes.
  - TD : 30h en 2002-2003.

A.2.2 Université Paris 13 - Institut Galilée : depuis septembre 2003

Maître de Conférences en Informatique (192h/an)

École d'ingénieurs Sup Galilée

- **Cursus Préparatoire Intégré Ingénieurs** :
  - Algorithmique
    - 2012-2014 : CM (12h/an)
    - 2012-2014 : TD (12h/an)
    - 2012-2014 : TP (12h/an)
    - 2013-2014 : Colles (26h/an)
  - Architecture, Systèmes et Réseaux
    - 2009-2011 : CM (12h/an)
    - 2009-2011 : TD (12h/an)
    - 2009-2011 : TP (12h/an)
    - 2009-2011 : Colles (12h en 2009-2010 et 10h en 2010-2011)
  - Oraux d'admission au concours GEIPI-Polytech
    - 2008-2011 et 2012-2013 : 1 à 2 demi-journées par an
- **1ère année toutes spécialités** :
  - Informatique de base (programmation C)
    - 2005-2010 : CM (10.5h/an)
    - 2005-2009 : TP (21h/an)
- **1ère année spécialité Informatique, parcours Informatique et Réseaux en apprentissage** :
  - Informatique de base (programmation C)
    - 2010-2011 : CM (10.5h)

- **2ème année spécialité Informatique, parcours Informatique et Réseaux en apprentissage** :
  - Algorithmique de graphes
    - 2012-2014 : CM (18h)
- **1ère année spécialité Informatique** :
  - Paradigme impératif
    - 2004-2005 : CM (12h)
    - 2003-2005 : TD (20h/an)
  - Unix
    - 2005-2010 : CM-TD-TP (12h/an)
  - Architecture des ordinateurs
    - 2006-2007 : CM (20h)
    - 2006-2007 : TD (27h)
    - 2006-2008 : TP (12h/an)
  - Suivi de stages de découverte de l'entreprise
    - 2006-2007 : 7 stages
    - 2007-2011 : 4 stages/an
  - Jury d'admission au concours Archimède
    - 2003-2007 : 1 à 2 demi-journées par an
- **2ème année spécialité Informatique** :
  - Algorithmique de Graphes
    - 2003-2004 et 2007-2010 : CM (16h en 2003-2004 et 18h/an depuis 2007)
    - 2003-2007 : TD (20h/an de 2003 à 2005 et 18h/an de 2005 à 2007)
  - Optimisation Combinatoire
    - 2007-2008 et 2012-2014 : CM (18h en 2007-2008 puis 15h en 2012-2014)
    - 2003-2007 : TD (24h/an de 2003 à 2005 et 18h/an de 2005 à 2007)
    - 2004-2007 : TP (12h/an)
  - Programmation Linéaire
    - 2010-2011 : TP (12h)
  - Encadrement de projets (groupes de 4 étudiants)
    - 2005-2009 et 2010-2011 : 1 projet/an
  - Encadrement et/ou suivi de stages de technicien
    - 2006-2011 et 2012-2013 : 2 à 3 stages/an
- **3ème année spécialité Informatique** :
  - Optimisation multi-critères
    - 2003-2004 : CM (15h)
    - 2003-2004 : TP (6h)
  - Aide à la décision
    - 2004-2008 : CM (9h/an)
    - 2004-2008 : TP (3h/an)
  - Optimisation et logiciels
    - 2006-2007 et 2008-2010 : CM (6h en 2006-2007 et 4h30 en 2008-2010)
    - 2005-2006 et 2008-2010 : TP (6h/an)
  - Résolution de problèmes de grande taille
    - 2010-2011 : CM (7.5h)
    - 2010-2011 : TP (3h)
  - Recherche Opérationnelle
    - 2012-2014 : CM (4.5h en 2012-2013 et 6h en 2013-2014)

  - 2012-2014 : TP (4.5h en 2012-2013 et 6h en 2013-2014)
- Encadrement de projets de fin d'études (groupes de 1 à 4 étudiants)
  - 2007-2008 : 6 projets en 2007-2008 et 2 projets en 2008-2009
- Encadrement et/ou suivi de stages de fin d'études
  - 2003-2011 et 2012-2013 : 2 stages/an
- **3ème année spécialité Mathématiques Appliquées et Calcul Scientifique** :
- Optimisation combinatoire
  - 2006-2010 : CM-TD-TP (24h/an de 2006 à 2008 et 21h/an en 2008-2010)

Licence

- **Licence 1 Sciences, Technologies et Santé** :
- Éléments d'informatique
  - 2012-2014 : CM (2*18h en 2012-2013 et 18h+9h en 2013-2014)
  - 2012-2014 : TD (36h en 2012-2013 et 18h en 2013-2014)
  - 2012-2014 : TP (33h en 2012-2013 et 29h en 2013-2014)
- **Deug MASS2** : Informatique
- 2003-2005 : CM (18h/an)
- **Licence 3 Informatique** :
- Algorithmique de Graphes
  - 2007-2011 : CM (18h/an)
  - 2003-2005 : TD (21h/an)
- Architecture des ordinateurs
  - 2003-2006 : TD (12h/an)
  - 2003-2006 : TP (12h/an)
- Bases de Données
  - 2009-2011 : TD (19,5h/an)
  - 2009-2011 : TP (19,5h/an)
- Systèmes
  - 2003-2004 : TP (20h)
- **Classe Préparatoire à l'Enseignement Supérieur** : Architecture, Systèmes et Réseaux
- 2007-2009 : CM (15h/an)
- 2007-2009 : TD (15h/an)
- 2007-2009 : TP (15h/an)

Master

- **Master Informatique 1ère année** :
- Encadrement de projets (groupes de 3 étudiants)
  - 2003-2009 : 4 projets en tout
- Bases de Données
  - 2008-2009 : TD (10,5h)
  - 2008-2009 : TP (15h)
- **Master Mathématiques et Informatique 1ère année** : Encadrement de projets (groupes de 3 étudiants)
- 2003-2004 : 1 projet
- **Master Mathématiques et Informatique 2ème année, spécialité Optimisation, Modélisation et Calculs** :

  - Programmation mathématique approfondie
    - 2009-2010 : CM (6h/an)
  - Optimisation et Logiciels
    - 2009-2010 : CM (4h30/an)
    - 2009-2010 : TP (6h/an)
  - Aide à la décision
    - 2004-2008 : CM (9h/an)
    - 2004-2008 : TP (3h/an)
- **Master Informatique 2ème année, spécialité Exploration Informatique des Données et spécialité Modélisation Informatique des Connaissances et du Raisonnement** :
  - Aide à la décision
    - 2004-2008 : CM (9h/an)
    - 2004-2008 : TP (3h/an)
- **DEA IA&OC (Intelligence Artificielle & Optimisation Combinatoire) puis Master Mathématiques et Informatique 2ème année, spécialité Optimisation, Modélisation et Calculs** :
  - Méthodes avancées en optimisation combinatoire
    - 2003-2009 : CM (6h/an)
    - 2003-2008 : TP (3h/an)
  - Co-encadrement de stages (6 mois entre avril et septembre)
    - 2003-2009 : 9 stages

A.2.3 Récapitulatif

Voir Tableau A.1.

	Algorithmique et Programmation	Architecture, Systèmes, Réseaux et Bases de Données	Recherche Opérationnelle	Total
CM	291.5	96	331.5	719
TD	222.5	237.5	340	800
TP	235	220	155.5	610.5
Total	749	553.5	827	2129.5

TABLE A.1 – *Tableau récapitulatif des enseignements (heures)*

A.3 ENCADREMENT

Titulaire d'un contrat de prime d'encadrement doctoral et de recherche (PEDR), d'octobre 2008 à septembre 2012, puis de la prime d'excellence scientifique (PES) depuis octobre 2012

A.3.1 Post-Doctorat

1. Co-encadrement (50%) avec Roberto Wolfler Calvo (Professeur, LIPN - Université Paris 13) du Post-Doctorat d'Emiliano Traversi (LIPN - Université Paris 13) de septembre 2013 à août 2014, "Bulk distribution optimization within VMI industrial" financé par un contrat de recherche avec Air Liquide.
2. Co-encadrement (50%) avec Roberto Wolfler Calvo (Professeur, LIPN - Université Paris 13) du Post-Doctorat de Fabio Furini (LIPN - Université Paris 13) d'octobre 2012 à septembre 2013, "Reoptimization and Column Generation" financé par une bourse de Google Grant Program on Mathematical and Combinatorial Optimization.
3. Suivi du Post-Doctorat de Sonia Cafieri (LIX - École Polytechnique) de mars 2008 à mars 2009, "Recherche de reformulations automatiques", encadrée par Leo Liberti (Professeur, LIX - École Polytechnique) et financée par l'ANR Jeunes Chercheurs ARS (Automatic Reformulation Search).

A.3.2 Doctorat

1. Suivi de la thèse de doctorat en co-tutelle de Marco Casazza sur la période juin 2013 - juin 2015, co-encadré par Alberto Ceselli (Maître de conférences, Università degli Studi di Milano, Italie) et Roberto Wolfler Calvo (Professeur, LIPN - Université Paris 13) et financé par une allocation italienne.
2. Co-encadrement (33%) avec Sylvie Borne (Maître de conférences, LIPN, Université Paris 13) et Roberto Wolfler Calvo (Professeur, LIPN - Université Paris 13) de la thèse de doctorat d'Hanane Allaoua (octobre 2011 - décembre 2014), "Optimisation des services aux personnes dépendantes" financée par une allocation ministérielle.
3. Co-encadrement (33%) avec Laurent Alfandari (Professeur, ESSEC) et Roberto Wolfler Calvo (Professeur, LIPN - Université Paris 13) de la thèse de doctorat de Paolo Gianessi (mai 2011 - septembre 2014), "Optimisation de problèmes de planification stratégique et tactique en logistique urbaine" financé par l'ANR Villes Durables MODUM (Mutualisation et Optimisation de la Distribution Urbaine de Marchandises).
4. Co-encadrement (50%) avec Céline Rouveirol (Professeur, LIPN - Université Paris 13) de la thèse de doctorat de Karima Mouhoubi (octobre 2009 - septembre 2013), "Recherche de motifs contraints dans des données bruitées" financée par une allocation ministérielle. Soutenue le 30/09/2013 à l'Université Paris 13.
5. Co-encadrement (50%) avec François Malgouyres (Professeur, IMT - Toulouse) de la thèse de doctorat de Nicolas Lermé (octobre 2008 - décembre 2011), "Réduction de graphes et application à la segmentation de tumeurs pulmonaires" financé par une allocation ministérielle. Soutenue le 07/12/2011 à l'Université Paris 13.

6. Co-encadrement (50%) avec Anass Nagih (Professeur, LITA - Université de Metz) de la thèse de doctorat de Nora Touati (octobre 2005 - décembre 2008), “Sur l’amélioration des performances de la génération de colonnes” financée par une allocation ministérielle. Soutenue le 03/12/2008 à l’Université Paris 13.

A.3.3 Monitorat

1. Tuteur de Manisha Pujari dans le cadre de sa mission d’enseignement de son contrat doctoral à l’Université Paris 13 (Institut Galilée) sur la période juin 2012 - juin 2013.

A.3.4 Master

Master 2ème année / DEA

Stages de 6 mois entre avril et septembre *Les stages ci-dessous ont été / sont co-encadrés avec différents collègues :*
N. Achir (L2TI), L. Alfandari (ESSEC-LIPN), S. Borne (LIPN), A. Ceselli (Milan, Italie), D. De Almeida (SNCF), F. Malgouyres (LAGA), A. Nagih (LITA), G. Plateau (LIPN), M-C. Plateau (GDF-Suez), C. Rouveirol (LIPN)

1. Déploiement des stations de base dans un réseau de capteurs (A. Sahbani, L2TI 2014)
2. Heuristiques pour le problème du voyageur de commerce avec fenêtres de temps en temps réel (Suivi de L. Bussi Roncalini, Milan, Italie 2011)
3. Branch and Cut pour le problème du voyageur de commerce asymétrique avec contraintes de saut (F. Ait Salaht, LIPN 2010)
4. Algorithmes de graphes pour la recherche de motifs bruités (K. Mouhoubi, LIPN 2009)
5. Graph Cuts pour la restauration d’images (N. Lermé, LIPN-LAGA 2008)
6. Résolution exacte du problème du voyageur de commerce asymétrique (S. Abdallah, LIPN 2008)
7. Développement d’un outil intranet d’aide à la décision pour la conception des roulements des agents sédentaires (O. Berraho, SNCF 2007)
8. Méthodes lagrangiennes pour la résolution du sac à dos quadratique avec contrainte de cardinalité (A. Benouali, LIPN 2007)
9. Méthodes de décomposition et génération de colonnes (Z. Beddar, LIPN 2007)
10. Réoptimisation du sac à dos quadratique (T. Radakovic, LIPN 2006)
11. Réoptimisation au sein d’un schéma de génération de colonnes (N. Touati, LIPN 2005)
12. Réoptimisation du sac à dos quadratique (R. Ouaras, LIPN 2005)
13. Modèles réalistes de déplacement d’acteurs virtuels (G. Elain, Thalès 2005)

Master 1ère année / Maîtrise

Projets de 4 mois entre mars et juin *Certains projets ci-dessous ont été co-encadrés avec A. Nagih (LITA) ou N. Touati (LIPN)*

1. Application web pour une plateforme d'optimisation en transport, 2009
2. Problème d'EDT : optimisation de l'affectation des salles de TP - Approche exacte, 2007
3. Étude de l'approche de génération de colonnes pour la résolution de problèmes classiques de recherche opérationnelle, 2005.
4. Problème d'EDT : optimisation de l'affectation des salles de TP - Approche par génération de colonnes, 2004
5. Problème d'EDT : optimisation de l'affectation des salles de TP - Approche heuristique, 2004
6. Résolution du sac à dos quadratique, 2004

A.3.5 École d'ingénieurs Sup Galilée

École d'ingénieurs Sup Galilée 3ème année

Projets de fin d'études de 4 mois entre novembre et février

1. Monitoring de process Windows (en collaboration avec EADS), 2009
2. Optimisation de l'outil CamStudio 2 (en collaboration avec EADS), 2009
3. Facturation Buena Vista International via la plateforme OGONE (en collaboration avec Distribution Service), 2008
4. Analyse de l'ergonomie du logiciel Spider (en collaboration avec Nokia Siemens Networks), 2008
5. Développement d'un logiciel de mesure de sondes thermiques sur pale d'hélicoptère (en collaboration avec Eurocopter), 2008
6. Développement d'une bibliothèque de classes pour la conversion des états crystal report (en collaboration avec PHD), 2008
7. Planification du personnel et création d'emplois du temps pour des centres d'amincissement (en collaboration avec PRO CD Informatique), 2008
8. Optimisation de l'outil CamStudio (en collaboration avec EADS), 2008

Stages de fin d'études de 6 mois entre avril et septembre *Les stages ci-dessous ont été co-encadrés avec différents collègues :*
J. André (GDF), D. De Almeida (SNCF), A. Nagih (LITA), G. Plateau (LIPN)

1. Développement d'un outil intranet d'aide à la décision pour la conception des roulements des agents sédentaires (O. Berraho, SNCF 2007)
2. Renforcement des réseaux de transport de gaz (M. Antunes, GDF 2005)
3. Modèles réalistes de déplacement d'acteurs virtuels (G. Elain, Thalès 2005)

École d'ingénieurs Sup Galilée 2ème année

Projets de 8 mois entre novembre et juin *Certains projets ci-dessous ont été co-encadrés avec V. Mogbil (LIPN), C. Rouveirol (LIPN), A. Rozenknop (LIPN), R. Wolfler-Calvo (LIPN)*

1. Site web officiel d'une cycliste de haut niveau, 2014
2. Application web pour une plateforme de gestion d'énergie, 2011
3. Gestion de perturbations dans le domaine aérien, 2009
4. Application web pour l'évaluation des enseignements, 2008
5. Développement d'une application web pour l'AEIG (Association des Élèves de l'Institut Galilée), 2007
6. Système informatique d'affectation des salles de TP, 2006

Stages de 2 mois entre juillet et août *Le stage ci-dessous a été co-encadré avec C. Rouveirol (LIPN)*

1. Développement de l'application web du BDE (Bureau Des Étudiants), 2007

École d'ingénieurs Sup Galilée, parcours en apprentissage

Tuteur d'apprentis

1. Tuteur de Bastien Duval, apprenti chez Capgemini Telecom Media Defense, septembre 2010 – septembre 2013

A.3.6 Récapitulatif

Voir Tableau A.2.

Post-Doctorats		Doctorats		Stages		Projets	Tutorats
Suivi	Encadrement	Suivi	Encadrement	Suivi	Encadrement	Suivi	Suivi
1	2	1	5	57	16	20	2

TABLE A.2 – *Tableau récapitulatif des encadrements et suivis (nombre)*

A.4 ACTIVITÉS DE RECHERCHE

Ma recherche, depuis 1999, concerne la poursuite de travaux fondamentaux et appliqués en optimisation en nombres entiers.

Depuis ma nomination en tant que Maître de Conférences à l'université Paris 13, j'effectue mes recherches au sein de l'équipe AOC (Algorithmes et Optimisation Combinatoire) du LIPN (Laboratoire d'Informatique de Paris Nord).

J'ai bénéficié d'un accueil en délégation CNRS au LIPN (Laboratoire d'Informatique de Paris Nord) UMR 7030, Université Paris 13 en 2011-2012. J'ai été titulaire d'une prime d'encadrement doctoral et de recherche (PEDR), d'octobre 2008 à septembre 2012 et je bénéficie d'une prime d'excellence scientifique (PES) depuis octobre 2012.

A.4.1 Bilan

Les travaux au sein du LIPN concernent actuellement :

- les méthodes de décompositions et la réoptimisation pour la résolution de programmes de grande taille (application : planification en transport, en énergie et en santé), en collaboration avec : Laurent Alfandari (Professeur, LIPN - ESSEC), Hanane Allaoua (Doctorante, LIPN - Université Paris 13), Roberto Baldacci (Associate Professor, DEI, Univertità di Bologna, Italie), Sylvie Borne (Maître de Conférences, LIPN - Université Paris 13), Marco Casazza (Doctorant en co-tutelle, DTI, Univertità degli Studi di Milano, Italie et LIPN - Université Paris 13), Alberto Ceselli (Associate Professor, DTI, Univertità degli Studi di Milano, Italie), Fabio Furini (Maître de Conférences, LAMSADE - Université Paris Dauphine), Paolo Gianessi (Doctorant, LIPN - Université Paris 13), Antoine Rozenknop (Maître de Conférences, LIPN - Université Paris 13) et Emiliano Traversi (Post-Doctorant, LIPN - Université Paris 13) et Roberto Wolfler Calvo (Professeur, LIPN - Université Paris 13) ;
- les relaxations et la convexification pour la résolution de problèmes quadratiques en variables binaires, en collaboration avec Monique Guignard-Spielberg (Full Professor, DOIM - University of Pennsylvania, the Wharton School, USA), Gérard Plateau (Professeur émérite, LIPN - Université Paris 13) et Angelika Wiegele (Associate Professor, Math. Dep., Alpen-Adria-Universität Klagenfurt, Autriche).

Les travaux antérieurs au sein du LIPN concernaient :

- les méthodes de décompositions, leur accélération et la réoptimisation pour la résolution de programmes de grande taille, en collaboration avec : Anass Nagih (Professeur, LITA - Université de Metz) dans le cadre de la thèse de doctorat de Nora Touati (allocation de recherche 2005-2008) ;
- la reformulation et l'introduction d'inégalités valides pour le problème du voyageur de commerce asymétrique avec contraintes de saut, en collaboration avec Laurent Alfandari, Sylvie Borne (Maître de Conférences, LIPN - Université Paris 13) et Pierre Pesneau (IMB - Université de Bordeaux 1) ;
- la ré-optimisation en nombres entiers et la dualité lagrangienne pour la résolution du problème du sac à dos quadratique, en collaboration avec Anass Nagih et Gérard Plateau (Professeur, LIPN - Université Paris 13) ;

- les heuristiques, les relaxations (semi-définie, agrégée) et la convexification pour la résolution du problème du sac à dos quadratique avec contrainte de cardinalité, en collaboration avec Gérard Plateau (Professeur émérite, LIPN - Université Paris 13) et Marie-Christine Plateau (Ingénieur de recherche, GDF-Suez) ;
- la recherche de reformulations automatiques, en collaboration avec Leo Liberti (Professeur, LIX - École Polytechnique), Frédéric Messine (Maître de Conférences, IRIT - ENSEEIHT), Pierre Hansen (Professeur, GERAD -HEC Montréal) et Sonia Cafieri (Maître de conférences, École Nationale de l'Aviation Civile) dans le cadre d'un projet ANR Jeunes Chercheurs ;
- l'algorithmique de graphes, pour le traitement d'images, en collaboration avec François Malgouyres (Professeur, IMT - Université de Toulouse), dans le cadre de la thèse de doctorat de Nicolas Lermé (allocation de recherche 2008-2011) ;
- l'algorithmique de graphes, pour la recherche de motifs bruités en bioinformatique, en collaboration avec Céline Rouveirol (Professeur, LIPN - Université Paris 13) dans le cadre de la thèse de Karima Mouhoubi (allocation de recherche 2009-2012).

Les collaborations avec Marie-Christine Costa (Professeur, CEDRIC - CNAM), Frédéric Roupin (Professeur, LIPN - Université Paris 13) et Cédric Bentz (Maître de Conférences, CEDRIC - CNAM) s'articulaient principalement autour de :

– la complexité des problèmes de multiflots maximaux et de multicoupes minimales dans les structures particulières, tels que les arbres et les anneaux ;
– la détermination d'algorithmes polynomiaux efficaces pour ces problèmes dans de telles structures.

Mots clés

– Méthodes :
  Relaxation lagrangienne, relaxation agrégée, décomposition lagrangienne, décomposition de Dantzig-Wolfe, génération de colonnes, programmation dynamique, branch and bound/cut/price, réoptimisation, reformulation et modélisation, algorithmes et complexité, heuristiques et matheuristiques ;
– Problèmes :
  Planification en transport, tournées de véhicules avec contraintes de ressources, localisation et routage, inventory routing, voyageur de commerce (asymétrique avec contraintes de saut, temps réel avec fenêtres de temps), multiflots et multicoupes, plus courts chemins contraints, énergie, sac à dos en 0-1 (linéaire, quadratique, avec contrainte de cardinalité), affectation quadratique, sous-graphes denses ;
– Applications :
  Systèmes de production et gestion des flux dans les réseaux de transport, logistique urbaine, hospitalisation à domicile, planification ferroviaire, planification et dimensionnement des ressources, traitement d'images, analyse de données d'expression de gènes, dimensionnement de réseaux gaziers, planification des arrêts des centrales thermiques/nucléaires, réaffectation de processus à des machines.

A.5 Contrats et valorisation scientifique

A.5.1 Contrats de recherche et coopérations industrielles

- GDF SUEZ : Participation à une prestation de conseil pour *GDF SUEZ* en 2013 ; titre de la prestation : Optimisation des capacités d'un réseau de transport de gaz ; Autres membres de l'équipe : Frédéric Roupin (Paris 13-LIPN), Roberto Wolfler Calvo (Paris 13-LIPN). Partenaire industriel : GDF SUEZ.
- Air Liquide : Participation à un projet *Air Liquide* pour 2013-2014 ; titre du projet : Bulk distribution optimization within VMI industrial ; Autre membre de l'équipe : Roberto Wolfler Calvo (Paris 13-LIPN). Partenaire industriel : Air Liquide.
- BQR Université Paris 13 : Projet interdisciplinaire : Participation à un projet interdisciplinaire dans le cadre du Bonus Qualité Recherche de l'Université Paris 13 pour l'année 2012 ; titre du projet : Méthodes automatiques et adaptatives de segmentation pour la détection et le suivi de l'évolution de pathologies par analyse d'images médicales ; Autres membres de l'équipe : Françoise Dibos (Paris 13-LAGA), Nicolas Lermé (Paris 13-LAGA/LIPN), Sébastien Li-Thiao-té (Paris 13-LAGA), Marie Luong (Paris 13-L2TI), Jean-Marie Rocchisani (Hôpital Avicenne, UFR SMBH), Quang Tung Thieu (Paris 13-L2TI), Dinh Hoan Trinh (Paris 13-LAGA/L2TI), Emmanuel Viennet (Paris 13-L2TI).
- Google : Participation à un projet *Google Focused Grant Program on Mathematical Optimization and Combinatorial Optimization in Europe* pour l'année 2012 ; titre du projet : Reoptimization and Column Generation ; Autre membre de l'équipe : Roberto Wolfler Calvo (Paris 13-LIPN). Partenaire industriel : Google.
- ANR (Agence Nationale de la Recherche) : Participation à un projet *ANR Ville Durable* pour une durée de quatre ans de décembre 2010 à décembre 2014 ; titre du projet : Mutualisation et Optimisation de la Distribution Urbaine de Marchandises (MODUM) ; Autres membres de l'équipe : Nabil Absi (EMSE-ARMINES-CMP), Laurent Alfandari (ESSEC-LIPN), Christian Ambrosini (Lyon 2-LET), Stéphane Dauzère-Pérès (EMSE-ARMINES-CMP), Dominique Feillet (EMSE-ARMINES-CMP), Jesus Gonzales-Feliu (Lyon 2-LET), Frédéric Meunier (ENPC-LVMT), Joëlle Morana (Lyon 2-LET), Jean-Louis Routhier (Lyon 2-LET), Florence Toilier (Lyon 2-LET), Roberto Wolfler Calvo (Paris 13-LIPN, responsable du projet). Partenaires académiques : ARMINES-CMP (Ecole des Mines de Saint-Etienne), ESSEC (Cergy), LET (Université Lyon 2), LVMT (Ecole Nationale des Ponts et Chaussées).
- Projet Relations Internationales de l'Université Paris 13 : Participation à un projet Relations Internationales de l'Université Paris 13 pour l'année 2010 ; titre du projet : Modèles et algorithmes pour résoudre les problèmes de conception de réseaux de distribution à deux niveaux ; Autres membres de l'équipe : Roberto Baldacci (Universita di Bologna), Daniel Chemla (Paris 13-LIPN), Vittorio Maniezzo (Universita di Bologna), Aristide Mingozzi (Universita di Bologna), Roberto Roberti (Universita di Bologna), Sophie Toulouse (Paris 13-LIPN), Roberto Wolfler Calvo (Paris 13-LIPN, responsable du projet). Partenaire académique : Universita di Bologna (Italie).
- PEPS : Interactions Mathématiques-Informatiques-Ingénierie du CNRS : Participation à un projet scientifique exploratoire dans le cadre des Pro-

jets Exploratoires PluridisciplinaireS du CNRS pour l'année 2010 ; titre du projet : DisCut : Segmentation de tumeurs par graph-cuts distribués ; Autres membres de l'équipe : Franck Butelle (Paris 13-LIPN), Lucian Finta (Paris 13-LIPN), François Malgouyres (Paris 13-LAGA, responsable du projet), Nicolas Lermé (Paris 13-LAGA/LIPN), Jean-Marie Rocchisani (Paris 13-BTI).

- BQR Université Paris 13 : Projets scientifiques interdisciplinaires : Participation à un projet scientifique interdisciplinaire dans le cadre du Bonus Qualité Recherche de l'Université Paris 13 pour l'année 2010 ; titre du projet : SinToSC : Segmentation 3d INteractive d'images TOmographiques : application au Suivi évolutif des Cancers pulmonaires par volumétrie tumorale ; Autres membres de l'équipe : Camille Guillerminet (Paris 13-BTI), Françoise Dibos (Paris 13-LAGA), François Malgouyres (Paris 13-LAGA, responsable du projet), Jean-François Morère(Paris 13-Oncologie), Nicolas Lermé (Paris 13-LAGA/LIPN), Jean-Marie Rocchisani (Paris 13-BTI).
- Projet Relations Internationales de l'Université Paris 13 : Participation à un projet Relations Internationales de l'Université Paris 13 pour l'année 2009 ; titre du projet : Méthodes et modèles pour la résolution exacte de nouveaux problèmes complexes émergeant dans le domaine de la logistique et du transport de marchandises. ; Autres membres de l'équipe : Roberto Baldacci (Universita di Bologna), Enrico Bartolini (Universita di Bologna), Sylvie Borne (Paris 13-LIPN), Vittorio Maniezzo (Universita di Bologna), Aristide Mingozzi (Universita di Bologna), Roberto Roberti (Universita di Bologna), Sophie Toulouse (Paris 13-LIPN), Roberto Wolfler Calvo (Paris 13-LIPN, responsable du projet). Partenaire académique : Universita di Bologna (Italie).
- GDR Recherche Opérationnelle du CNRS : Porteur et participation à un projet du GDR RO pour l'année 2009 ; titre du projet : Projections, heuristiques, relaxations et méthodes de coupes pour le problème du voyageur de commerce asymétrique avec contraintes de saut ; Autres membres de l'équipe : Laurent Alfandari (ESSEC-LIPN), Sylvie Borne (Paris 13-LIPN), Pierre Pesneau (Bordeaux 1- IMB). Partenaires académiques : ESSEC (Cergy), IMB (Bordeaux).
- Projet LIPN : Participation à un projet LIPN (*Laboratoire d'Informatique de Paris Nord*) pour l'année 2009 ; titre du projet : Algorithmes de graphes pour la recherche de motifs fréquents flous ; Autre membre de l'équipe : Céline Rouveirol (Paris 13-LIPN).
- BQR Université Paris 13 : Projet scientifique émergeant : Participation à un projet scientifique émergeant dans le cadre du Bonus Qualité Recherche de l'Université Paris 13 pour l'année 2009 ; titre du projet : Étude croisée des méthodes d'optimisation combinatoire et de leurs applications en segmentation et recalage pour l'analyse et le traitement des images. ; Autres membres de l'équipe : Françoise Dibos (Paris 13-LAGA), François Malgouyres (Paris 13-LAGA) et Ghilès Mostafaoui (Paris 13-L2TI).
- SNCF DI&R : Participation à un contrat d'*Étude et de veille technologique* avec la Direction de l'Innovation et de la Recherche de la SNCF, janvier 2005 - décembre 2007.
- ANR (Agence Nationale de la Recherche) : Participation à une *ANR Jeunes Chercheuses et Jeunes Chercheurs* pour une durée de trois ans d'octobre 2007 à septembre 2010 ; titre du projet : Automatic Reformulation Search ;

Autres membres de l'équipe : Leo Liberti (École Polytechnique-LIX, responsable du projet), Philippe Baptiste (École Polytechnique-LIX), Christophe Durr (École Polytechnique-LIX), Frédéric Messine (ENSEEIHT-IRIT), Marie-Christine Plateau (GDF) ; Partenaire industriel : GDF ; Partenaire académique : Pierre Hansen (Montréal-GERAD), Frank Plastria (Bruxelles-MOSI).

- Ministère de la recherche : Participation à une *ACI Jeunes Chercheuses et Jeunes Chercheurs* (Action Concertée Incitative) pour une durée de trois ans de septembre 2004 à août 2007 ; titre du projet : Planification et gestion optimisée des ressources en transport : réoptimisation et hybridation de la génération de colonnes et des métaheuristiques ; Autres membres de l'équipe : Anass Nagih (Metz-LITA, responsable du projet), Laurent Alfandari (ESSEC-LIPN), Agnès Plateau (CNAM-CEDRIC) et Sophie Toulouse (Paris 13-LIPN) ; Partenaire industriel : SNCF ; Partenaire académique : GERAD (Canada).

A.5.2 Brevets et Logiciels

- FMR Programme de calcul de ré-affectation de processus FMR (Fast Machine Reassignment) développé dans le cadre du challenge Roadef 2011-2012 ; Licence : GPL3 ; `http://www-lipn.univ-paris13.fr/butelle/s26.tgz` ; Autres participants : Laurent Alfandari (ESSEC-LIPN), Franck Butelle (Paris 13-LIPN, responsable du logiciel), Camille Coti (Paris 13-LIPN), Lucian Finta (Paris 13-LIPN), Gérard Plateau (Paris 13-LIPN), Frédéric Roupin (Paris 13-LIPN), Antoine Rozenknop (Paris 13-LIPN), Roberto Wolfler-Calvo (Paris 13-LIPN).
- Reduction of VISION GRAPHS Brevet Reduction of VISION GRAPHS en janvier 2010, Patent No. FR2955408 (A1) ; Autres participants : Nicolas Lermé (Paris 13-LAGA/LIPN), François Malgouyres (Toulouse-IMT, responsable du dépôt).
- COGiTO Dépôt du logiciel COGiTO (Column Generation in Transportation Optimization) en 2009 auprès de l'APP, IDDN No. FR.001.050008.000.S.P.2009.000.30805 ; Autres participants : Laurent Alfandari (ESSEC-LIPN), Jérôme Bier (Université de Metz-LITA), Nathan Godard (Université de Metz-LITA), Olivier Laval (Paris 13-LIPN), Anass Nagih (Université de Metz-LITA, responsable du dépôt), Agnès Plateau, (CNAM-CEDRIC), Alexandre Quivet (Paris 13-LIPN), Jalila Sadki Fenzar (Paris 13-LIPN), Nora Touati (Paris 13-LIPN), Sophie Toulouse (Paris 13-LIPN).

A.5.3 Challenges

- EURO/ROADEF 2012 16ème du challenge EURO/ROADEF 2012 (9ème de la catégorie Open Source) : Machine Reassignment ; Autres membres de l'équipe : Laurent Alfandari (ESSEC-LIPN), Franck Butelle (Paris 13-LIPN, responsable), Camille Coti (Paris 13-LIPN), Lucian Finta (Paris 13-LIPN), Gérard Plateau (Paris 13-LIPN), Frédéric Roupin (Paris 13-LIPN), Antoine Rozenknop (Paris 13-LIPN), Roberto Wolfler-Calvo (Paris 13-LIPN).
- EURO/ROADEF 2010 4ème du challenge EURO/ROADEF 2010 : A large-scale energy management problem with varied constraints ; Autres membres

de l'équipe : Laurent Alfandari (ESSEC-LIPN), Daniel Chemla (Paris 13-LIPN), Antoine Rozenknop (Paris 13-LIPN), Guillaume Turri (Paris 13-LIPN), Roberto Wolfler-Calvo (Paris 13-LIPN, responsable).

A.5.4 Expertises

- Referee d'articles pour les revues Computers & Industrial Engineering, Computers & Operations Research, Discrete Applied Mathematics, European Journal of Operational Research, RAIRO - Operations Research et TOP et pour les conférences STACS et ROADEF.
- Évaluation d'une demande de subventions à la découverte par le conseil de recherches en sciences naturelles et en génie du Canada (NSERC) en 2014.
- Évaluation d'une demande de subventions par le Fonds Québécois de la Recherche sur la Nature et les Technologies (FQRNT) en 2014.
- Membre d'un comité d'évaluation d'une équipe associée INRIA en 2010-2011.
- Évaluation d'un projet ANR Blanc SIMI 2 en 2011.

A.5.5 Organisation et animation

- Membre du comité d'organisation de l'école d'hiver sur la génération de colonnes, qui s'est déroulée du 10 au 14 mars 2014 à l'IHP à Paris.
- Membre du comité scientifique de la conférence ROADEF (Société Française de Recherche Opérationnelle et Aide à la Décision) 2014, qui s'est déroulée du 26 au 28 février 2014 à l'université Bordeaux Segalen.
- Participation à l'organisation de la conférence internationale CO 2002 : International Symposium on Combinatorial Optimization, qui s'est déroulée du 8 au 10 avril 2002 au CNAM à Paris.
- Membre du comité d'organisation des Journées Franciliennes de Recherche Opérationnelle de septembre 2003 à septembre 2006. Organisation de 8 journées (et webmaster du site).
- Présentation de démonstrations "Courses et jeux de fourmis" et "Êtes-vous plutôt cigale ou fourmi : la Recherche Opérationnelle expliquée par les jeux vidéos" aux manifestations Savante Banlieue, Science en fête, Villetaneuse, en 2005 et 2008.
- Membre du Groupe de Travail "Knapsack et Optimisation" de la ROADEF depuis sa création en 2007.
- Membre du Groupe de Travail "Images" commun aux universités Paris 5 et Paris 13 et à l'ONERA depuis sa création en 2008.
- Membre du Groupe de Travail "Apprentissage et Optimisation Combinatoire" du LIPN, Université Paris 13 depuis sa création en 2009.

A.5.6 Invitations

- Invitation d'une semaine par l'Università degli Studi di Milano, Italie, en juillet 2011.

A.5.7 Membre des sociétés savantes

- ROADEF : Société Française de Recherche Opérationnelle et Aide à la Décision, depuis 2000.

A.6 Activités administratives et responsabilités collectives

A.6.1 Au niveau national

- Commissions de spécialistes et comités de sélection :
  - Membre d'un comité de sélection pour un poste profilé Recherche Opérationnelle - 27ème section du CNU de l'université de Valenciennes et du Hainaut-Cambrésis en 2013.
  - Membre d'un comité de sélection pour un poste profilé Recherche Opérationnelle - 27ème section du CNU de l'université de Picardie Jules Verne en 2012.
  - Membre d'un comité de sélection pour un poste profilé Méthodes et modèles de conception et évaluation de systèmes interactifs - 27ème section du CNU de l'université de Valenciennes et du Hainaut-Cambrésis en 2012.
  - Membre d'un comité de sélection pour un poste profilé Recherche Opérationnelle - 27ème section du CNU de l'université de Valenciennes et du Hainaut-Cambrésis en 2011.
  - Membre de la commission de spécialistes - 27ème section du CNU de l'université de Blaise-Pascal (Clermont-Ferrand) entre septembre 2006 et septembre 2008 (titulaire).
  - Membre de la commission de spécialistes - 27ème section du CNU de l'université de Cergy-Pontoise entre septembre 2004 et septembre 2008 (suppléant).
- Jury de thèse :
  - Membre examinateur : Amel Benhamiche (LAMSADE - Paris Dauphine, 12/12/2013).
- Jury de baccalauréat :
  - Président d'un jury de baccalauréat S en juin 2012 à Enghien les bains (95).

A.6.2 Au niveau de l'Université

- Conseil de l'Institut Galilée :
  - Membre élu du conseil (d'administration) de l'Institut Galilée de l'université Paris 13 de mai 2009 à mai 2013.
- Commission des moyens de l'Institut Galilée :
  - Membre élu de la commission des moyens de l'Institut Galilée de l'université Paris 13 de mai 2009 à mai 2013.
- Commission de spécialistes, comité d'experts et comités de sélection :
  - Membre du comité de recrutement des ATER de l'université Paris 13 en 2013.
  - Membre d'un comité de sélection (27) de l'université Paris 13 en 2013.
  - Membre de 2 comités de sélection (27 et 27/61) de l'université Paris 13 en 2011.
  - Membre du comité de recrutement des ATER de l'université Paris 13 en 2011.
  - Membre élu du comité d'experts de la 27ème section de l'université Paris 13 de janvier 2009 à janvier 2013.

  - Membre élu de la commission de spécialistes 27ème section de l'université Paris 13, suppléant entre septembre 2006 et septembre 2008.
- Jurys de thèse :
  - Co-encadrant : Karima Mouhoubi (LIPN, 20/09/13), Nicolas Lermé (LAGA-LIPN, 07/12/11), Nora Touati (LIPN, 03/12/08).

A.6.3 Au niveau du laboratoire

- Conseil de laboratoire :
  - Membre élu du conseil de laboratoire du LIPN (Laboratoire d'Informatique de Paris Nord) depuis juillet 2013. Représentant de l'équipe AOC (Algorithmes et Optimisation Combinatoire).
  - Membre élu du conseil de laboratoire du CEDRIC (Centre d'Étude et De Recherche en Informatique du Cnam) de 2001 à août 2003. Représentant des doctorants et ATER.
- Membre du jury de pré-soutenance de thèse de : Paolo Gianessi (LIPN, 09/10/2013), Hanane Allaoua (LIPN, 09/10/2013), Karima Mouhoubi (LIPN, 06/07/2012) et Nora Touati (LIPN, 20/11/07).

A.6.4 Au niveau du département

- Département d'informatique :
  - Vice-Président du département d'informatique (Institut Galilée) depuis octobre 2012.
  - Membre du bureau du département d'informatique (Institut Galilée) de septembre 2005 à décembre 2009.
- École d'ingénieurs Sup Galilée :
  - Directeur des études de la spécialité *Informatique* de septembre 2006 à septembre 2010.
  - Responsable de la 2ème année de la spécialité *Informatique* en 2005-2006.
  - Responsable de la 3ème année de la spécialité *Informatique* en 2006-2007.
  - Responsable de l'option "Aide à la Décision et Optimisation" de la 3ème année de la spécialité *Informatique* de septembre 2009 à septembre 2011.
  - Participation à deux campagnes d'habilitation de l'École par la CTI (Commission des Titres d'Ingénieur) en 2007 et 2010.
  - Participation au montage, à l'habilitation et à la mise en place de la spécialité *Informatique et Réseaux* en apprentissage ouverte en septembre 2010.
  - Membre du Conseil de l'École d'ingénieurs de septembre 2006 à septembre 2010.
  - Membre des Conseils de perfectionnement de l'École d'ingénieurs de septembre 2006 à septembre 2010.
  - Président des jurys de :
    - 2ème année de la spécialité *Informatique* en 2006.
    - 3ème année de la spécialité *Informatique* en 2007.
  - Membre nommé des jurys de :
    - 1ère année de la spécialité *Informatique* de 2006 à 2010.
    - 1ère année de la spécialité *Télécommunication* de 2006 à 2008.
    - 2ème année de la spécialité *Informatique* de 2007 à 2011 et en 2013.

  - 3ème année de la spécialité *Informatique* de 2005 à 2011.
  - 3ème année de la spécialité *Informatique et Réseaux* en 2013.
  - 3ème année de la spécialité *Mathématiques Appliquées et Calcul Scientifique* de 2006 à 2010.
- Licence et Master :
  - Membre nommé des jurys de :
    - Licence 1 Sciences et Technologies en 2013 et 2014 ;
    - DEUG MASS en 2004 et 2005.
    - Licence 3 Informatique en 2004, 2008 et 2009 ;
    - Master Mathématiques et Informatique 2ème année de 2006 à 2011 ;
- Classes préparatoires :
  - Membre du groupe de travail sur la mise en place du programme des enseignements d'informatique de la classe préparatoire à l'enseignement supérieur mixte Lycée Feyder - Institut Galilée - Sup Méca en 2005-2006.
  - Membre du groupe de travail sur la mise en place du programme des enseignements d'informatique de la classe préparatoire intégrée à l'École d'ingénieurs Sup Galilée en 2007-2008.

A.7 PUBLICATIONS

A.7.1 Revues internationales à comité de lecture

1. Antoine Rozenknop, Roberto Wolfler-Calvo, Laurent Alfandari, Daniel Chemla, **Lucas Létocart** (2013) "Solving the electricity production planning problem by a column generation based heuristic", *Journal of Scheduling*, Vol. 16 (6), pp 585-604.
2. **Lucas Létocart**, Anass Nagih, Gérard Plateau (2012) "Reoptimization in Lagrangian methods for the quadratic knapsack problem", *Computers and Operations Research*, Vol. 39 (1), pp. 12-18.
3. **Lucas Létocart**, Anass Nagih, Nora Touati Moungla (2012) "Dantzig-Wolfe and Lagrangian decompositions in integer linear programming", *International Journal of Mathematics in Operational Research*, Vol. 4 (3), pp. 247-262.
4. Nora Touati Moungla, **Lucas Létocart**, Anass Nagih (2010) "Solutions diversification in a column generation algorithm", *Algorithmic Operations Research*, Vol. 5, pp. 86-95.
5. Cédric Bentz, Marie-Christine Costa, **Lucas Létocart**, Frédéric Roupin (2009) "Multicut and integral multiflow in rings", *European Journal of Operational Research*, Vol. 196 (3), pp. 1251-1254.
6. Marie-Christine Costa, **Lucas Létocart**, Frédéric Roupin (2005) "Multicut and integral multiflow : a survey", *European Journal of Operational Research*, Vol. 162 (1), pp. 55-69.
7. Marie-Christine Costa, **Lucas Létocart**, Frédéric Roupin (2003) "A greedy algorithm for multicut and integral multiflow in rooted trees", *Operations Research Letters*, Vol. 31, pp. 21-27.

A.7.2 Revues nationales à comité de lecture

8. **Lucas Létocart**, Marie-Christine Plateau, Gérard Plateau (2014) "A fast hybrid heuristic for the 0-1 exact k-item quadratic knapsack problem", *Pesquisa Operacional*, Vol. 34 (1), pp. 1-22.

A.7.3 Pré-publications en soumission/révision

9. Paolo Gianessi, Laurent Alfandari, **Lucas Létocart**, Roberto Wolfler-Calvo "The Multicommodity-Ring Location Routing Problem" en révision pour *Transportation Science* (25 pages).
10. Franck Butelle, Laurent Alfandari, Camille Coti, Lucian Finta, **Lucas Létocart**, Gérard Plateau, Frédéric Roupin, Antoine Rozenknop, Roberto Wolfler Calvo "Fast Machine Reassignment" en révision pour *Annals of Operations Research* (20 pages).
11. Marco Casazza, Alberto Ceselli, **Lucas Létocart** "Dynamically negotiating time slots in attended home delivery" en soumission à *Omega* (19 pages).

A.7.4 Conférences internationales avec actes (sur articles)

12. Hanane Allaoua, Sylvie Borne, **Lucas Létocart**, Roberto Wolfler-Calvo "A matheuristic approach for solving a home health care problem" *INOC 2013 : International Network Optimization Conference*, Tenerife, Espagne, mai 2013, *Electronic Notes in Discrete Mathematics*, Vol. 41, pp. 471-478.
13. Hanane Allaoua, Sylvie Borne, **Lucas Létocart**, Roberto Wolfler-Calvo "Combining routing and rostering for the home health care problem" *TRISTAN VIII : 8th TRIennal Symposium on Transportation ANalysys*, San Pedra de Atacama, Chilie, juin 2013, 4 pages.
14. Karima Mouhoubi, **Lucas Létocart**, Céline Rouveirol "A knowledge-driven bi-clustering method for mining noisy datasets", *ICONIP 2012 : 19th International Conference on Neural Information Processing*, Doha, Qatar, novembre 2012, pp. 585-593, *Lecture Notes in Computer Science*.
15. Sonia Cafieri, Pierre Hansen, **Lucas Létocart**, Leo Liberti, Frédéric Messine "Compact relaxations for polynomial programming problems", *SEA 2012 : 11th International Symposium on Experimental Algorithms*, Bordeaux, France, juin 2012, pp. 75-86, *Lecture Notes in Computer Science*.
16. Laurent Alfandari, Paolo Gianessi, **Lucas Létocart**, Roberto Wolfler-Calvo "Solving network design and routing problems in urban freight distribution issues" *ODYSSEUS 2012 : 5th International Workshop on Freight Transportation and Logistics*, Mykonos, Grèce, mai 2012, pp. 564-567.
17. Karima Mouhoubi, **Lucas Létocart**, Céline Rouveirol "Itemset mining in noisy contexts : a hybrid approach", *ICTAI 2011 : 23rd IEEE International Conference on Tools with Artificial Intelligence*, Boca Raton, Florida, USA, novembre 2011, pp. 33-40.
18. Nicolas Lermé, **Lucas Létocart**, François Malgouyres "Reduced graphs for min-cut/max-flow approaches in image segmentation", *LAGOS'11 : VI Latin-American Algorithms, Graphs, and Optimization Symposium*, Bariloche, Argentine, mars 2011, *Electronic Notes in Discrete Mathematics*, Vol. 37, pp. 63-68.
19. Nicolas Lermé, François Malgouyres , **Lucas Létocart** "Reducing graphs in graph cut segmentation", *ICIP 2010 : International Conference on Image Processing*, Hong Kong, Chine, septembre 2010, pp. 3045-3048.
20. Nora Touati Moungla, **Lucas Létocart**, Anass Nagih "An improving dynamic programming algorithm to solve the shortest path problem with time windows", *ISCO 2010 : International Symposium on Combinatorial Optimization*, Hammamet, Tunisie, mars 2010, *Electronic Notes in Discrete Mathematics*, Vol. 36, pp. 931-938.
21. Nora Touati, **Lucas Létocart**, Anass Nagih (2009) "Diversification and reoptimization procedures in column generation for the resolution of the acyclic vehicle routing problem with time windows", *INOC 2009 : International Network Optimization Conference*, Pise, Italie, avril 2009, 6 pages.

A.7.5 Conférences nationales avec actes (sur articles)

22. Karima Mouhoubi, **Lucas Létocart**, Céline Rouveirol "Extraction de motifs ensemblistes dans des contextes bruités", *CAP 2011 : Conférence francophone d'APprentissage*, Chambéry, France, mai 2011, 16 pages.

23. Nicolas Lermé, François Malgouyres, **Lucas Létocart**, Jean-Marie Rocchisani "Une méthode de réduction exacte pour la segmentation par graph cuts", *ORASIS 2011 : Congrès des jeunes chercheurs en vision par ordinateur*, Praz-sur-Arly, France, juin 2011, 8 pages.
24. Karima Mouhoubi, **Lucas Létocart**, Céline Rouveirol "Heuristique pour l'extraction de motifs ensemblistes bruités", *EGC 2011 : 11ème Conférence Internationale Francophone sur l'Extraction et la Gestion des Connaissances*, Brest, France, janvier 2011, pp. 467-472.

A.7.6 Conférences invitées

25. Monique Guignard-Spielberg, Michael Bussieck, **Lucas Létocart** (2014) "Using SDP-based convexification for quadratic MIP problems", *INFORMS 2014 : INFORMS Annual Meeting*, San Fransisco, États-Unis, novembre 2014.
26. **Lucas Létocart**, Paolo Gianessi, Alberto Ceselli, Roberto Wolfler-Calvo (2014) "A Branch&Price Approach for the Vehicle Routing Problem with Intermediate Replenishment Facilities" *IFORS 2014 : International Federation of Operational Research Societies Conference*, Barcelone, Espagne, juillet 2014.
27. Roberto Wolfler-Calvo, Paolo Gianessi, **Lucas Létocart** (2014) "A new exact approach for the Vehicle Routing Problem with Intermediate Replenishment Facilities" *ROUTE 2014 : An international workshop on vehicle routing, intermodal transportation and related areas*, Comwell Borupgaard, Danemark, juin 2014.
28. **Lucas Létocart**, Fabio Furini, Roberto Wolfler-Calvo (2014) "Local Reoptimization via column generation and quadratic programming", *SIAM Conference on Optimization*, San Diego, États-Unis, mai 2014.
29. Roberto Wolfler-Calvo, Fabio Furini, **Lucas Létocart** (2013) "Local Reoptimization for set partitioning problem", *PGMO's days : Conference of the Gaspard Monge program for optimization and operations research*, Paris, France, octobre 2013.
30. Monique Guignard-Spielberg, Aykut Ahlatcioglu, Michael Bussieck, Peter Hahn, **Lucas Létocart** (2012) "Quadratic combinatorial optimization models : why they are needed, and a few approaches to solve them", *CO 2012 : International Symposium on Combinatorial Optimization*, Oxford, Royaume-Uni, septembre 2012.
31. Monique Guignard-Spielberg, **Lucas Létocart**, Gérard Plateau (2012) "0-1 quadratic optimization problems : convexification and solution", *EURO 2012 : 25th European conference on operational research*, Vilnius, Lituanie, juillet 2012.
32. **Lucas Létocart**, Marie-Christine Plateau, Gérard Plateau (2012) "SDP reformulation within a surrogate dual heuristic for the 0-1 exact k-item quadratic knapsack problem", *ECCO 2012 : 25th Conference of European Chapter on Combinatorial Optimization*, Antalya, Turquie, avril 2012.
33. Monique Guignard-Spielberg, Aykut Ahlatcioglu, **Lucas Létocart**, Gérard Plateau (2012) "A primal heuristic for several quadratic pure 0-1 models", *ECCO 2012*, Antalya, Turquie, avril 2012.

34. Roberto Wolfler Calvo, Antoine Rozenknop, Daniel Chemla, Laurent Alfandari, **Lucas Létocart** (2011) "A column generation approach for scheduling nuclear power plants refueling", *Workshop following the challenge EURO/ROADEF 2010*, EDF, Clamart, France, mars 2011.
35. Nicolas Lermé, François Malgouyres, **Lucas Létocart** (2010) "Segmentation d'images par une coupe dans un graphe", *Journée Mathématiques-Biologie* du Laboratoire d'Analyse, Géométrie et Applications, Université Paris 13, février 2010.
36. **Lucas Létocart** (2008) "Markov Random Fields minimization and minimal cuts in image restoration", *ARS08 Workshop*, École Polytechnique, Palaiseaux, octobre 2008.
37. **Lucas Létocart** (2007) "Inégalités valides pour le problème de multiflot maximum", *2ème journée scientifique Polyèdres et Optimisation Combinatoire*, Paris, mars 2007.
38. **Lucas Létocart**, Anass Nagih, Gérard Plateau (2007) "Réoptimisation dans les méthodes lagrangiennes pour le sac à dos quadratique", *1ère journée KnapSack et Optimisation*, Paris, mars 2007.

A.7.7 Conférences internationales avec comité de sélection (sur résumé)

39. Alberto Ceselli, Marco Casazza, **Lucas Létocart** "Optimizing time slot allocation in single operator home delivery problems" *OR 2014 : International conference on operations research*, Aix La Chapelle, Allemagne, Septembre 2014.
40. **Lucas Létocart**, Marco Casazza, Antoine Rozenknop, Emiliano Traversi, Roberto Wolfler-Calvo (2014) "Solving the ROADEF/EURO 2014 Challenge by a double column generation based heuristic" *IFORS 2014 : International Federation of Operational Research Societies Conference*, Barcelone, Espagne, juillet 2014.
41. Monique Guignard-Spielberg, **Lucas Létocart**, Michael Bussieck (2014) "Progress and issues with SDP-based convexification and convex hull matheuristic for quadratic 0-1 problems with linear constraints" *MINLP 2014 : Mixed-Integer Nonlinear Programming workshop*, Pittsburg, États-Unis, juin 2014.
42. **Lucas Létocart**, Fabio Furini, Roberto Wolfler-Calvo (2013) "Reoptimization and Column Generation for Bin Packing Related Problems", *EURO XXVI : 26th European conference on operational research*, Rome, Italie, juillet 2013.
43. Gérard Plateau, Monique Guignard-Spielberg, **Lucas Létocart** (2012) "Improved quadratic convex reformulation for variants of the 0-1 quadratic knapsack problem", *XVI CLAIO (Congreso Latino-Iberoamericano de Investigacion Operativa) / XLIV SBPO (Simposio Brasileiro de Pesquisa Operacional)*, Rio de Janeiro, Brésil, septembre 2012.
44. Hanane Allaoua, Sylvie Borne, **Lucas Létocart**, Roberto Wolfler-Calvo (2012) "Combining routing and rostering for the home health care problem" *ORAHS 2012 : 38th annual meeting of the EURO working group on OR Applied to Health Services*, Enschede, Pays-Bas, juillet 2012.

45. Monique Guignard-Spielberg, **Lucas Létocart**, Gérard Plateau (2012) "The Generalized Quadratic Assignment Problem : convexification and solution", *INFORMS 2012 : INFORMS International*, Pékin, Chine, juin 2012.
46. Roberto Wolfler-Calvo, Antoine Rozenknop, Daniel Chemla, Laurent Alfandari, **Lucas Létocart**, Guillaume Turri (2010) "A column generation approach for scheduling nuclear power plants refueling", *EURO XXIV : 24th European conference on operational research*, Lisbonne, Portugal, juillet 2010.
47. Gérard Plateau, **Lucas Létocart**, Anass Nagih (2010) "0-1 knapsack problems and reoptimization", *CIRO'10 : Cinquième Conférence Internationale en Recherche Opérationnelle*, Marrakech, Maroc, mai 2010.
48. **Lucas Létocart**, Marie-Christine Plateau, Gérard Plateau (2009) "A surrogate dual heuristics for the 0-1 exact k-item quadratic knapsack problem", *ISMP'09 : The 20th International Symposium on Mathematical Programming*, Chicago, Etats-Unis, août 2009.
49. Nora Touati, **Lucas Létocart**, Anass Nagih (2008) "Solutions diversification in a column generation scheme", *IFORS 2008 : International Federation of Operational Research Societies Conference*, Sandton, Afrique du Sud, juillet 2008.
50. Nora Touati, **Lucas Létocart**, Anass Nagih (2008) "Reoptimization techniques in a column generation scheme", *ECCO XXI : European Chapter on Combinatorial Optimization*, Dubrovnik, Croatie, mai 2008.
51. Laurent Alfandari, **Lucas Létocart** (2007) "Linear and quadratic formulations for the asymmetric traveling salesman problem", *NCP 07 : The international conference on NonConvex Programming : local and global approaches, Theory, Algorithms and Applications*, Rouen, France, décembre 2007.
52. **Lucas Létocart**, Marie-Christine Plateau, Gérard Plateau (2007) "Lagrangian and convexification methods for the 0-1 exact k-item quadratic knapsack problem", *NCP 07*, Rouen, France, décembre 2007.
53. **Lucas Létocart**, Anass Nagih, Gérard Plateau (2007) "Reoptimization in the resolution of the 0-1 quadratic knapsack problem", *EURO XXII : 22nd European conference on operational research*, Prague, République tchèque, juillet 2007.
54. **Lucas Létocart**, Marie-Christine Plateau, Gérard Plateau (2007) "New bounds for the 0-1 exact k-item quadratic knapsack problem", *ECCO XX*, Limassol, Chypre, mai 2007.
55. Marie-Christine Costa, **Lucas Létocart**, Frédéric Roupin (2003) "Minimal multicut and maximal integer multiflow in rings", *ISMP'03 : The 18th International Symposium on Mathematical Programming*, Copenhague, Danemark, août 2003.
56. **Lucas Létocart**, Frédéric Roupin (2002) "A semidefinite approach to solve multicut in trees", *JOPT'02 : Journées de l'optimisation - Optimization Days*, Montréal, Canada, mai 2002.
57. Marie-Christine Costa, **Lucas Létocart**, Frédéric Roupin (2001) "Multicut and integral multiflow : a survey", *ECCO XIV*, Bonn, Allemagne, juin 2001.

58. Marie-Christine Costa, **Lucas Létocart**, Frédéric Roupin (2001) "A greedy algorithm for multicut and integral multiflow in rooted trees", *JOPT'01*, Québec, Canada, mai 2001.

A.7.8 Conférences nationales avec comité de sélection (sur résumé)

59. **Lucas Létocart**, Monique Guignard-Spielberg, Gérard Plateau, Frédéric Roupin, Angelika Wiegele (2014) "Approches exactes pour le problème du sac à dos quadratique avec contrainte de cardinalité" *ROADEF'2014 : 15ème congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision*, Bordeaux, février 2014.
60. Paolo Gianessi, **Lucas Létocart**, Roberto Wolfler-Calvo (2014) "New Branch&Cut Approaches for the Vehicle Routing Problem with Intermediate Replenishment Facilities" *ROADEF'2014*, Bordeaux, février 2014.
61. Monique Guignard, Peter Hahn, Aykut Ahlatcioglu, **Lucas Létocart**, Michael Bussieck (2014) "Quelques approches pour résoudre des problèmes quadratiques en 0-1, en particulier de type affectation, avec application à un problème de crossdock" *ROADEF'2014*, Bordeaux, février 2014.
62. Hanane Allaoua, Sylvie Borne, **Lucas Létocart**, Roberto Wolfler Calvo (2014) "Planification et routage des personnels pour l'hospitalisation à domicile" *ROADEF'2014*, Bordeaux, février 2014.
63. Emiliano Traversi, Mehdi Lamiri, Roberto Wolfler Calvo, **Lucas Létocart**, Jean André (2014) "Inventory Routing Problem with rational objective function" *ROADEF'2014*, Bordeaux, février 2014.
64. Fabio Furini, **Lucas Létocart**, Roberto Wolfler-Calvo (2013) "Local Reoptimization for Bin Packing Related Problems" *ROADEF'2013 : 14ème congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision*, Troyes, février 2013.
65. Hanane Allaoua, Sylvie Borne, **Lucas Létocart**, Roberto Wolfler-Calvo (2013) "Planification des personnels pour l'hospitalisation à domicile" *ROADEF'2013*, Troyes, février 2013.
66. Paolo Gianessi, Laurent Alfandari, **Lucas Létocart**, Roberto Wolfler-Calvo (2012) "Planification et routage pour l'optimisation de la distribution urbaine de marchandises" *ROADEF'2012 : 13ème congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision*, Angers, avril 2012.
67. Karima Mouhoubi, **Lucas Létocart**, Céline Rouveirol (2011) "Une approche heuristique hybride pour l'extraction de motifs ensemblistes dans des contextes bruités", *JFGG'2011 : Journée Fouille de Grands Graphes*, Grenoble, octobre 2011.
68. Nicolas Lermé, **Lucas Létocart**, François Malgouyres (2011) "Réduction de graphes et flot maximum pour la segmentation et le débruitage d'images", *ROADEF'2011 : 12ème congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision*, Saint-Étienne, mars 2011.
69. Karima Mouhoubi, **Lucas Létocart**, Céline Rouveirol (2011) "Heuristique pour l'extraction de motifs ensemblistes bruités", *ROADEF'2011*, Saint-Étienne, mars 2011.

70. Nicolas Lermé, François Malgouyres, **Lucas Létocart** (2010) “Réduction de graphes pour la segmentation d’images par graph cuts”, *CANUM 2010 : 40ème Congrès national d’ANalyse NUMérique*, Bordeaux, juin 2010.
71. Nora Touati, **Lucas Létocart**, Anass Nagih (2009) “Décomposition lagrangienne et génération de colonnes” *JPOC 6 : Journées Polyèdres et Optimisation Combinatoire*, Bordeaux, juin 2009.
72. **Lucas Létocart**, Marie-Christine Plateau, Gérard Plateau (2009) “Une heuristique duale pour le sac à dos quadratique avec contrainte de cardinalité”, *ROADEF’2009* , Nancy, février 2009.
73. Nicolas Lermé, **Lucas Létocart**, François Malgouyres (2009) “Restauration d’images par coupes minimales”, *ROADEF’2009*, Nancy, février 2009.
74. Sylvie Borne, Laurent Alfandari, **Lucas Létocart** (2009) “Le problème du voyageur de commerce asymétrique avec contraintes de saut”, *ROADEF’2009*, Nancy, février 2009.
75. Nora Touati, **Lucas Létocart**, Anass Nagih (2009) “Programmation dynamique par blocs”, *ROADEF’2009*, Nancy, février 2009.
76. Nora Touati, **Lucas Létocart**, Anass Nagih (2008) “Sur la qualité des colonnes générées dans un schéma de génération de colonnes”, *JPOC 5*, Rouen, juin 2008.
77. **Lucas Létocart**, Laurent Alfandari, Sylvie Borne (2008) “Modèles linéaires et quadratiques pour le problème du voyageur de commerce asymétrique”, *ROADEF’2008*, Clermont-Ferrand, février 2008.
78. Nora Touati, **Lucas Létocart**, Anass Nagih (2008) “Décomposition lagrangienne et génération de colonnes”, *ROADEF’2008*, Clermont-Ferrand, février 2008.
79. Nora Touati, **Lucas Létocart**, Anass Nagih (2007) “Diversification des solutions et réoptimisation pour la résolution de problèmes de plus court chemin avec contraintes de ressources dans un schéma de génération de colonnes”, *ROADEF’2007*, Grenoble, février 2007.
80. **Lucas Létocart**, Anass Nagih, Rachid Ouaras, Gérard Plateau (2006) “Réoptimisation du sac à dos quadratique”, *ROADEF’2006*, Lille, février 2006.
81. Marie-Christine Costa, **Lucas Létocart**, Frédéric Roupin (2003) “Multicoupe minimale et multiflot entier maximal dans un anneau”, *Les Journées Graphes, Réseaux et Modélisation*, Paris, décembre 2003
82. Marie-Christine Costa, **Lucas Létocart**, Frédéric Roupin (2003) “Multicoupes minimales et multiflots maximaux en nombres entiers dans les anneaux”, *ROADEF’2003*, Avignon, février 2003.
83. Marie-Christine Costa, **Lucas Létocart**, Frédéric Roupin (2002) “Multiflots entiers et multicoupes : analyse de leur difficulté”, *ROADEF’2002*, Avignon, février 2002.
84. **Lucas Létocart**, Pierre Loubières (2000) “Différenciation des routes aériennes”, *ROADEF’2000*, Nantes, février 2000.

A.7.9 Séminaires - Diffusion de la connaissance - Vulgarisation

85. **Lucas Létocart**, Nicolas Lermé, François Malgouyres (2014) “Réduction de graphes pour la segmentation d’images”, *Séminaire*, du Pôle MathSTIC, Université Paris 13, mars 2014.

86. **Lucas Létocart**, Nicolas Lermé, François Malgouyres, Karima Mouhoubi, Céline Rouveirol (2013) “Les algorithmes de flot à la rescousse”, *Séminaire*, du Pôle MathSTIC, Université Paris 13, avril 2013.
87. Nicolas Lermé, François Malgouyres, **Lucas Létocart**, Jean-Marie Rocchisani (2011) “Graph reduction and application of graph cut to lung tumor segmentation”, *Séminaire*, du Groupe de travail Images LAGA-LIPN-L2TI, Université Paris 13, mars 2011.
88. **Lucas Létocart** (2009) “Optimisation Combinatoire”, *Séminaire* du Groupe de travail Apprentissage et Optimisation Combinatoire du LIPN, Université Paris 13, novembre 2009.
89. Gérard Plateau, **Lucas Létocart**, Marie-Christine Plateau (2009) “Dual heuristics for the 0-1 exact k-item quadratic knapsack problem”, *Séminaire*, Rio de Janeiro, Brésil, mars 2009.
90. **Lucas Létocart** (2009) “(Multi)-flots et (multi)-coupes : formulations, complexité et méthodes de résolution”, *Séminaire*, du Groupe de travail Images LAGA-LIPN-L2TI, Université Paris 13, mars 2009.
91. **Lucas Létocart**, Laurent Alfandari, Roberto Wolfler Calvo (2009) “L’algorithmique et l’optimisation combinatoire au LIPN”, *Séminaire*, Équipe OSIRIS de la Direction de la Recherche d’EDF, Clamart, février 2009.
92. **Lucas Létocart** (2008) “Relaxations et décompositions pour la programmation en nombres entiers”, *Séminaire*, Université Paris Dauphine, LAMSADE, décembre 2008.
93. **Lucas Létocart** (2008) “Relaxations, Décompositions et Réoptimisations”, *Tutoriel*, Pôle Simulation et Optimisation de la Direction de la Recherche de Gaz De France, Saint-Denis, février 2008.
94. **Lucas Létocart**, Marie-Christine Costa, Frédéric Roupin (2003) “Résolution des problèmes de multicoupes minimales et de multiflots maximaux en nombres entiers dans les arbres et les anneaux”, *Séminaire*, Université Paris 13, LIPN, mars 2003.
95. Marie-Christine Costa, **Lucas Létocart**, Frédéric Roupin (2003) “Étude et résolution des problèmes de multicoupes et de multiflots entiers”, *Séminaire*, École Polytechnique Fédérale de Lausanne, Suisse, mars 2003.
96. Marie-Christine Costa, **Lucas Létocart**, Frédéric Roupin (2002) “Multicoupes, multiflots entiers et problèmes connexes”, *Séminaire*, Université Libre de Bruxelles, Belgique, juin 2002.
97. Marie-Christine Costa, **Lucas Létocart**, Frédéric Roupin (2002) “Multicoupes et multiflots entiers”, *Séminaire*, Université Paris 11, LRI, juin 2002.

A.7.10 Posters

98. Karima Mouhoubi, **Lucas Létocart**, Céline Rouveirol “Extraction de biclusters contraints dans des contextes bruités”, *CAP 2012 : Conférence francophone d’APprentissage*, Nancy, France, mai 2012.
99. Sonia Cafieri, Leo Liberti, **Lucas Létocart**, Frédéric Messine, Marie-Christine Plateau (2010) “ ARS : Automatic Reformulation Search. Searching automatic reformulations in Mathematical Programming”, *Colloque ANR STIC*, Cité des Sciences, Paris, janvier 2010.

100. **Lucas Létocart** (2003) “Utilisation de la programmation semi-définie pour la résolution des problèmes de multicoupe minimale”, *comité CNRS d'évaluation du LIPN*, Université Paris 13, Villetaneuse, novembre 2003.

A.7.11 Rapports de recherche

101. Nora Touati, **Lucas Létocart**, Anass Nagih “Méthodes de décomposition pour l'optimisation discrète”, *Rapport LIPN 2007* (30 pages).
102. Nora Touati, **Lucas Létocart**, Anass Nagih “Sur l'accélération de la convergence de la génération de colonnes”, *Rapport LIPN 2006-4* (29 pages).
103. Cédric Bentz, Marie-Christine Costa, **Lucas Létocart**, Frédéric Roupin “A bibliography on multicut and integer multiflow problems”, *Rapport CEDRIC 2004-654* (7 pages).

A.7.12 Rapports de contrats

104. Emiliano Traversi, **Lucas Létocart**, Roberto Wolfler Calvo “Inventory Routing and Scheduling Problem : problem formulation and solution methods”, *Rapport de contrat Air Liquide–LIPN 2013* (23 pages).
105. **Lucas Létocart**, Frédéric Roupin, Roberto Wolfler Calvo “Prestation de conseil sur une problématique d'optimisation des capacités d'un réseau de transport de gaz”, *Rapport de contrat GDF-SUEZ–LIPN 2013* (21 pages).
106. Laurent Alfandari, **Lucas Létocart**, Anass Nagih, Agnès Plateau, Sophie Toulouse “Décomposition lagrangienne et génération de colonnes pour la planification optimale de locomotives”, *Rapport de contrat SNCF-LIPN 2007* (46 pages).
107. Laurent Alfandari, **Lucas Létocart**, Anass Nagih, Agnès Plateau, Sophie Toulouse “Méthodes de décomposition pour l'optimisation discrète”, *Rapport de contrat SNCF-LIPN 2007* (48 pages).
108. Laurent Alfandari, **Lucas Létocart**, Anass Nagih, Agnès Plateau, Sophie Toulouse “Méthodes de génération de colonnes pour la planification optimale des locomotives”, *Rapport de contrat SNCF-LIPN 2006* (54 pages).
109. Laurent Alfandari, **Lucas Létocart**, Anass Nagih, Agnès Plateau, Sophie Toulouse “Techniques de résolution approchée : heuristiques et métaheuristiques”, *Rapport de contrat SNCF-LIPN 2006* (64 pages).
110. Laurent Alfandari, **Lucas Létocart**, Anass Nagih, Agnès Plateau, Sophie Toulouse “Modélisation et résolution de problèmes de couverture de tâches : logiciels du domaine libre, logiciel du commerce et heuristiques”, *Rapport de contrat SNCF-LIPN 2005* (56 pages).
111. Laurent Alfandari, **Lucas Létocart**, Anass Nagih, Agnès Plateau, Sophie Toulouse “Modélisation et approches de résolution”, *Rapport de contrat SNCF-LIPN 2005* (20 pages).

BIBLIOGRAPHIE

W. P. Adams et H. D. Sherali. A tight linearization and an algorithm for zero-one quadratic programming problems. *Management Science*, 32 :1274–1290, 1986. (Cité page 125.)

A. V. Aho, J. E. Hopcroft, et J. D. Ullman. *The design and analysis of computer algorithms*. Addison-Wesley Pub. Company, 1976. (Cité pages 127 et 128.)

Z. Akca, R. T. Berger, et T. K. Ralphs. A Branch-and-Price algorithm for Combined Location and Routing Problems under capacity restrictions. *Operations Research*, 47 :309–330, 2009. (Cité pages 91 et 93.)

S. Almoustafa, S. Hanafi, et N. Mladenovic. New exact method for large asymmetric distance-constrained vehicle routing problem. *European Journal of Operational Research*, 226(3) :386–394, 2013. (Cité page 84.)

D. Ambrosino et M. G. Scutellà. Distribution network design : New problems and related models. *European Journal of Operational Research*, 165 :610–624, 2001. (Cité page 90.)

J. An, A. Wee-Chung Liew, et C. Nelson. Seed-based biclustering of gene expression data. *PLoS ONE*, 7(8), August 2012. URL `http://eprints.qut.edu.au/56976/`. (Cité page 21.)

D. Applegate, R. E. Bixby, V. Chvátal, et W. Cook. Concorde TSP solver. `http://www.tsp.gatech.edu/concorde.html`, 2006. (Cité page 100.)

C. Archetti, L. Bertazzi, et M. Grazia Speranza. Reoptimizing the traveling salesman problem. *Networks*, 42(3) :154–159, 2003. (Cité page 110.)

C. Archetti, L. Bertazzi, et M. Grazia Speranza. Reoptimizing the 0-1 knapsack problem. *Discrete Applied Mathematics*, 158(17) :1879–1887, 2010. (Cité page 110.)

E. Balas et E. Zemel. Solving large zero-one knapsack problems. *Operations Research*, 28 :1130–1154, 1980. (Cité pages 127 et 133.)

R. Baldacci, A. Mingozzi, et R. Wolfler Calvo. An exact method for the capacitated location-routing problem. *Operations Research*, 59(5) :1284–1296, 2011. (Cité page 91.)

J. F. Bard, L. Huang, M. Dror, et P. Jaillet. A branch and cut algorithm for the vrp with satellite facilities. *IIE Transactions*, 30(9) :821–834, 1998. (Cité page 84.)

J. J. Bartholdi, J. B. Orlin, et H. D. Ratliff. Cyclic scheduling via integer programs with circular ones. *Operations Research*, 28(5) :1074–1085, 1980. (Cité pages 8 et 9.)

V. Bélanger, G. Plateau, A. Ruiz, P. Soriano, et R. Wolfler Calvo. A re-optimization approach applied to the single source capacitated location problem. Dans *ROADEF*, 2010a. (Cité page 109.)

V. Bélanger, A. Ruiz, P. Soriano, et R. Wolfler Calvo. A re-optimization approach applied to the single source capacitated location problem. Dans *Journées de l'Optimization, Montréal*, 2010b. (Cité page 109.)

V. Bélanger, A. Ruiz, P. Soriano, et R. Wolfler Calvo. Une approche de réoptimisation appliquée à la gestion des services préhospitaliers d'urgence. Dans *GISEH*, 2010c. (Cité page 109.)

V. Bélanger, A. Ruiz, P. Soriano, et R. Wolfler Calvo. An approach for dynamic redeployment of emergency medical vehicles. Dans *INFORMS Healthcare*, 2011. (Cité page 109.)

J. M. Belenguer, E. Benavent, C. Prins, C. Prodhon, et R. Wolfler Calvo. A Branch-and-Cut method for the Capacitated Location-Routing Problem. *Computers & Operations Research*, 38(6) :931–941, 2011. (Cité page 91.)

T. Belgacem et M. Hifi. Sensitivity analysis of the knapsack sharing problem : perturbation of the weight of an item. *Computers and Operations Research*, 35 : 295–308, 2008a. (Cité page 126.)

T. Belgacem et M. Hifi. Sensitivity analysis of the optimum to perturbation of the profit of a subset of items in the binary knapsack problem. *Discrete Optimization*, 5 :755–761, 2008b. (Cité page 126.)

R. Bellman. *Dynamic programming*. Princeton university press, NJ, USA, 1957. (Cité page 60.)

A. Ben-Tal, L. El Ghaoui, et A. Nemirovski. *Robust Optimization*. Princeton Series in Applied Mathematics, Princeton University Press, 2009. (Cité page 109.)

R. T. Berger. *Location-routing Models for Distribution System Design*. Northwestern University, 1997. (Cité pages 90, 91, 93 et 97.)

R. T. Berger, C. R. Coullard, et M. S. Daskin. Location-Routing problems with distance constraints. *Transportation Science*, 41(1) :29–43, 2007. (Cité pages 90, 91, 93 et 97.)

S. Bergmann, J. Ihmels, et N. Barkai. Iterative signature algorithm for the analysis of large-scale gene expression data. *Phys Rev E Stat Nonlin Soft Matter Phys*, 67(3 Pt 1) :031902, 2003. (Cité page 21.)

A. Bhattacharjee, W.G. Richards, J. Staunton, C. Li, S. Monti, P. Vasa, C. Ladd, J. Beheshti, R. Bueno, M. Gillette, M. Loda, G. Weber, E.J. Mark, E.S. Lander, W. Wong, B.E. Johnson, T.R. Golub, D.J. Sugarbaker, et M. Meyerson. Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses. *Proc Natl Acad Sci U S A*, 98(24) :13790–13795, 2001. (Cité pages 30 et 36.)

A. Billionnet. Different formulations for solving the heaviest k-subgraph problem. *Information Systems and Operational Research*, 43(3) :171–186, 2005. (Cité page 133.)

A. Billionnet et F. Calmels. Linear programming for 0-1 knapsack problem. *European Journal of Operational Research*, 92 :310–325, 1996. (Cité page 137.)

A. Billionnet, S. Elloumi, et A. Lambert. Extending the QCR method to general mixed-integer programs. *Mathematical Programming*, 131(1-2) :381–401, 2013. (Cité pages 134, 135, 136, 147 et 150.)

A. Billionnet, S. Elloumi, et M.-C. Plateau. Improving the performance of standard solvers via a tighter convex reformulation of constrained quadratic 0-1 programs : the QCR method. *Discrete Applied Mathematics*, 157 :1185–1197, 2009. (Cité pages 134, 135, 136 et 147.)

A. Billionnet, A. Faye, et E. Soutif. A new upper bound for the 0-1 quadratic knapsack problem. *European Journal of Operational Research*, 112 :664–672, 1999. (Cité pages 124, 125, 129 et 132.)

A. Billionnet et E. Soutif. Using a mixed integer programming tool for solving the 0-1 quadratic knapsack problem. *INFORMS Journal on Computing*, 16 : 188–197, 2004. (Cité pages 124, 125, 126, 129 et 132.)

J. Birge et F. Louveaux. *Introduction to stochastic programming*. Springer Series in Operations Research, New York : Springer, 2011. (Cité page 109.)

E. Birmele, M. Elati, C. Rouveirol, et C. Ambroise. Identification of functional modules based on transcriptional regulation structure. *BMC Proc.*, 2(Suppl 4) : S4, 2008. ISSN 1753-6561. (Cité page 29.)

M. Blum, R. W. Floyd, V. R. Pratt, R. L. Rivest, et R. E. Tarjan. Time bounds for selection. *Journal of Computer and Systems Sciences*, 7(4) :448–461, 1972. (Cité page 128.)

M. Boccia, T. G. Crainic, A. Sforza, et C. Sterle. A metaheuristic for a two echelon location-routing problem. Dans Paola Festa, éditeur, *SEA*, volume 6049 de *Lecture Notes in Computer Science*, pages 288–301. Springer, 2010. ISBN 978-3-642-13192-9. (Cité page 91.)

M. Boccia, T. G. Crainic, A. Sforza, et C. Sterle. Location-Routing models for designing a Two-Echelon Freight Distribution System. Rapport technique, CIRRELT, Université de Montréal, 2011. (Cité page 91.)

H. J. Bockenhauer, J. Hromkovic, R. Kralovic, T. Momke, et P. Rossmanith. Reoptimization of steiner trees : Changing the terminal set. *Theoretical computer science*, 410(36) :3428–3435, 2009. (Cité page 110.)

S. Borchers. CSDP, a C library for semidefinite programming. *Optimization methods and Software*, 11(1) :613–623, 1999. (Cité page 134.)

C. Borgelt, X. Yang, R. Nogales-Cadenas, P. Carmona-Saez, et A. Pascual-Montano. Finding closed frequent item sets by intersecting transactions. Dans *Proceedings of the 14th International Conference on Extending Database Technology*, EDBT/ICDT '11, pages 367–376, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0528-0. URL `http://doi.acm.org/10.1145/1951365.1951410`. (Cité page 17.)

P. Bourgeois et G. Plateau. Selected algorithmic tools for the resolution of the 0-1 knapsack problem. Dans *EURO XII - TIMS XXXI Joint International Conference*, Helsinki, 1992. (Cité page 128.)

Y. Boykov et M-P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. Dans *ICCV*, volume 1, pages 105–112, 2001. (Cité pages 13 et 15.)

Y. Boykov et V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on PAMI*, 26 (9) :1124–1137, 2004. (Cité pages 11 et 13.)

A. Caprara, H. Kellerer, U. Pferschy, et D. Pisinger. Approximate algorithms for knapsack problems with cardinality constraints. *European Journal of Operational Research*, 123 :333–345, 2000. (Cité page 134.)

A. Caprara, D. Pisinger, et P. Toth. Exact solution of the quadratic knapsack problem. *INFORMS Journal on Computing*, 11 :125–137, 1999. (Cité pages 124, 125, 129, 132 et 141.)

J. M. Cherry, C. Adler, C. Ball, S. A. Chervitz, et S. S. Dwight. Sgd : Saccharomyces genome database. *Nucleic Acids Research*, 26(1) :73–79, 1998. (Cité pages 29, 34 et 36.)

R. J. Cho, M. J. Campbell, E. A. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T. G. Wolfsberg, A. E. Gabrielian, D. Landsman, D. J. Lockhart, et et al. A genome-wide transcriptional analysis of the mitotic cell cycle. *Molecular Cell*, 2(1) :65–73, 1998. URL `http://www.ncbi.nlm.nih.gov/pubmed/9702192`. (Cité page 28.)

C. Cigla et A.A. Alatan. Region-based image segmentation via graph cuts. Dans *ICIP*, pages 2272–2275, 2008. (Cité page 12.)

S. Cosares et I. Saniee. An optimization problem related to balancing loads on sonet rings. *Telecommunication Systems*, 3 :165–181, 1994. (Cité page 7.)

M-C. Costa, L. Létocart, et F. Roupin. A greedy algorithm for multicut and integral multiflow in rooted trees. *Operations Research Letters*, 31(1) :21–27, 2003. (Cité page 8.)

A. Crema. An algorithm for the multiparametric 0-1 integer linear programming problem relative to objective function. *European Journal of Operational Reshearch*, 125 :18–24, 2000. (Cité page 126.)

B. Crémilleux et A. Soulet. Discovering knowledge from local patterns with global constraints. Dans *Proc. Int. Conf. Comp. Sci. Its App., Part II*, pages 1242–1257, 2008. (Cité page 21.)

B. Crevier, J.-F. Cordeau, et G. Laporte. The multi-depot vehicle routing problem with inter-depot routes. *European Journal of Operational Research*, 176(2) : 756–773, 2007. (Cité page 83.)

E. Dalhaus, D.S. Johnson, C.H. Papadimitriou, P.D. Seymour, et M. Yannakakis. The complexity of multiway cuts. *SIAM J. Comput.*, 23 :864–894, 1994. (Cité page 7.)

GB. Dantzig et P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8 :101–111, 1960. (Cité pages 51 et 53.)

A. Delong et Y. Boykov. A scalable graph-cut algorithm for N-D grids. Dans *CVPR*, pages 1–8, 2008. (Cité page 12.)

M. Deodhar, G. Gupta, J. Ghosh, H. Cho, et I. S. Dhillon. A scalable framework for discovering coherent co-clusters in noisy data. Dans *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009*, volume 382 de *ACM International Conference Proceeding Series*, page 31. ACM, 2009. (Cité pages 28 et 38.)

M. Desrochers. An algorithm for the shortest path problem with resource constraints. Rapport Technique G-88-27, Les cahiers du GERAD, 1988. (Cité page 41.)

M. Desrochers et F. Soumis. A reoptimization algorithm for the shortest path problem with time windows. *European Journal of Operational Research*, 35 : 242–254, 1988. (Cité pages 71 et 78.)

J. Desrosiers et F. Soumis. A generalized permanent labelling algorithm for the schortest path problem with time window. *INFOR*, 26(3) :191–212, 1988. (Cité page 41.)

R. Dionne. Etude and extension d'un algorithme de Murchland. *INFOR*, 16 : 132–146, 1978. (Cité pages 71 et 110.)

K. Dudzinski. On a cardinality constrained linear programming knapsack problem. *Operations Research Letters*, 8 :215–218, 1989. (Cité page 134.)

M. Elati, P. Neuvial, M. Bolotin-Fukuhara, E. Barillot, F. Radvanyi, et C. Rouveirol. Licorn : learning cooperative regulation networks from gene expression data. *Bioinformatics*, 23(18) :2407–2414, 2007. (Cité page 31.)

D. Fayard et G. Plateau. Reduction algorithm for single and multiple constraints 0-1 linear programming problems. Dans *Methods of Mathematical Programming, Zakopane (Poland)*, 1977. (Cité page 127.)

D. Fayard et G. Plateau. Contribution à la résolution des programmes mathématiques en nombres entiers. Dans *Thèse d'état, Université de Lille 1 (France)*, 1979. (Cité page 128.)

D. Fayard et G. Plateau. An algorithm for the solution of the 0-1 knapsack problem. *Computing*, 28 :269–287, 1982. (Cité pages 127 et 133.)

A. Faye et F. Roupin. Partial Lagrangian relaxation for general quadratic programming. *4OR*, 5(1) :75–88, 2007. (Cité page 136.)

M. Fischetti, J. J. Salazar Gonzalez, et P. Toth. A branch-and-cut algorithm for the symmetric generalized traveling salesman problem. *Operations Research*, 45(3) :378–394, 1997. (Cité page 96.)

A. Fréville et G. Plateau. An exact search for the solution of the surrogate dual of the 0-1 bidimensional knapsack problem. *European Journal of Operational Research*, 68 :413–421, 1993. (Cité pages 140 et 141.)

G. Gallo, P.L. Hammer, et B. Simeone. Quadratic knapsack problems. *Mathematical Programming Study*, 12 :132–149, 1980. (Cité page 124.)

N. Garg, V.V. Vazirani, et M. Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. *SIAM J. Comput.*, 2(2) :235–251, 1996. (Cité page 8.)

N. Garg, V.V. Vazirani, et M. Yannakakis. Primal-dual approximation algorithms for integral flow and multicut in trees. *Algorithmica*, 18(1) :3–20, 1997. (Cité page 7.)

A.P. Gasch, P.T. Spellman, C.M. Kao, O. Carmel-Harel, M.B. Eisen, G. Storz, D. Botstein, et P.O. Brown. Genomic expression programs in the response of yeast cells to environmental changes. *Mol. Biol. Cell*, 11(12) :4241–57, 2000. (Cité pages 28 et 36.)

A. M. Geoffrion et R. Nauss. Parametric and postoptimality analysis in integer linear programming. *Management Science*, 23 :453–466, 1977. (Cité page 126.)

P. Gianessi, L. Alfandari, L. Létocart, et R. Wolfler Calvo. The multicommodity-ring location routing problem. *Transportation Science*, en révision. (Cité pages 103 et 104.)

F. Glover. A multiphase dual algorithm for the 0-1 integer programming problem. *Operations Research*, 13(6) :879–919, 1965. (Cité page 140.)

A V Goldberg et R E Tarjan. A new approach to the maximum flow problem. Dans *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, STOC '86, pages 136–146, New York, NY, USA, 1986. ACM. ISBN 0-89791-193-8. URL `http://doi.acm.org/10.1145/12130.12144`. (Cité page 24.)

D. M. Greig, B. T. Porteous, et A. H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society*, 51(2) :271–279, 1989. (Cité page 11.)

S. Gueye et Ph. Michelon. Miniaturized linearizations for quadratic 0/1 problems. *Annals of Operations Research*, 140 :235–261, 2005. (Cité page 124.)

M. Guignard et S. Kim. Lagrangian decomposition : a model yielding stronger langrangian bounds. *Mathematical programming*, 32 :215–228, 1987a. (Cité pages 52 et 53.)

M. Guignard et S. Kim. Lagrangian decomposition for integer programming : theory and applications. *RAIRO*, 21 :307–324, 1987b. (Cité pages 52 et 53.)

R. Gupta, G. Fang, B. Field, M. Steinbach, et V. Kumar. Quantitative evaluation of approximate frequent pattern mining algorithms. Dans *KDD*, pages 301–309, 2008. (Cité pages 26 et 27.)

B. Hanczar et M. Nadif. Using the bagging approach for biclustering of gene expression data. *Neurocomputing*, 74(10) :1595–1605, 2011. (Cité pages 26, 28, 29, 38 et 40.)

R. Hassin et A. Tamir. Improved complexity bounds for location problems on the real line. *Operations Research Letters*, 10 :395–402, 1991. (Cité pages 8 et 9.)

C. Helmberg et F. Rendl. Solving quadratic (0,1) problems by semidefinite programs and cutting planes. *Mathematical Programming*, 82 :291–315, 1998. (Cité page 124.)

C. Helmberg et F. Rendl. A semidefinite programming approach to quadratic knapsack problems. *Journal of Combinatorial Optimization*, 4 :197–215, 2000. (Cité page 124.)

G. Hifi, H. Mhalla, et S. Sadfi. Sensitivity of the optimum to perturbations of the profit or weight of an item in the binary knapsack problem. *Journal of Combinatorial Optimization*, 10 :239–260, 2005. (Cité page 126.)

S. Holm et D. Klein. Three methods for post-optimal analysis in integer linear programming. *Mathematical Programming*, 21 :97–109, 1984. (Cité page 126.)

T. Hu, S. Y. Sung, H. Xiong, et Q. Fu. Discovery of maximum length frequent itemsets. *Inf. Sci.*, 178(1) :69–87, Janvier 2008. ISSN 0020-0255. URL `http://dx.doi.org/10.1016/j.ins.2007.08.006`. (Cité page 17.)

P. Hupé, N. Stransky, J.-P. Thiery, F. Radvanyi, et E. Barillot. Analysis of array CGH data : from signal ratio to gain and loss of DNA regions. *Bioinformatics*, 20(18) :3413–3422, 2004. (Cité page 31.)

S. Ji, X. Zheng, et X. Sun. An improved convex 0-1 quadratic program reformulation for quadratic knapsack problems. *Pacific Journal of Optimization*, 8(1) : 75–87, 2012. (Cité pages 147, 149 et 150.)

M. Kanehisa et S. Goto. Kegg : kyoto encyclopedia of genes and genomes. *Nucleic Acids Res*, 28(1) :27–30, 2000. (Cité pages 34 et 36.)

I. Kara. Arc based integer programming formulations for the distance constrained vehicle routing problem. Dans *Logistics and Industrial Informatics (LINDI), 2011 3rd IEEE International Symposium on*, pages 33–38, Aug 2011. (Cité page 84.)

H. Kellerer, U. Pferschy, et D. Pisinger. *Knapsack Problems*. Springer-Verlag Berlin Heidelberg, New York, 2004. (Cité pages 115 et 134.)

J.E. Kelley. The cutting-plane method for solving convex programs. *SIAM Journal on Optimization*, 8 :703–712, 1960. (Cité pages 51 et 52.)

P. Kohli, V. Lempitsky, et C. Rother. Uncertainty driven multi-scale optimization. *Pattern Recognition*, 6376 :242–251, 2010. (Cité page 12.)

V. Kolmogorov et R. Zabih. What energy functions can be minimized via graph cuts ? *IEEE Transactions on PAMI*, 26(2) :147–159, 2004. (Cité pages 13 et 14.)

P. Kouvelis et G. Yu. *Robust Discrete Optimization and Its Applications*. Kluwer, 1997. (Cité page 109.)

P. Lacomme. *Algorithmes de graphes*. EYROLLES, 2003. (Cité page 41.)

G. Laporte. Location-routing problems. Dans B. Golden et A. Assad, éditeurs, *Vehicle Routing : Methods and Studies*, pages 163–196. North Holland, Amsterdam, 1988. (Cité page 90.)

G. Laporte. A survey of algorithms for location-routing problems. *Investigación Operativa*, 1 :93–123, 1989. (Cité page 90.)

G. Laporte et Y. Nobert. An exact algorithm for minimizing routing and operating costs in depot location. *European Journal of Operational Research*, 6(2) :224–226, 1981. (Cité page 90.)

G. Laporte, Y. Nobert, et D. Arpin. An exact algorithm for solving a capacitated location-routing problem. *Annals of Operations Research*, 6(9) :291–310, 1986. (Cité page 90.)

O. Laval, A. Nagih, et S. Toulouse. Rapport de recherche sur le problème du plus court chemin contraint. Rapport Technique 2006-05, LIPN - CNRS UMR 7030 - Université Paris 13, 2006. (Cité page 43.)

E. L. Lawler. Fast approximation algorithms for knapsack problems. *Mathematics of Operations Research*, 4 :339–356, 1979. (Cité pages 127 et 128.)

I. Lee, S.V. Date, A.T. Adai, et E.M. Marcotte. A probalistic functionnal network of yeast genes. *Science*, 306(5701) :1555–1558, 2004. (Cité page 36.)

C. Lemaréchal. The omnipresence of lagrange. *4OR*, 1(1) :7–25, 2003. (Cité pages 51 et 54.)

V. Lempitsky et Y. Boykov. Global optimization for shape fitting. Dans *CVPR*, pages 1–8, 2007. (Cité pages 11 et 12.)

Y. I. Leon-Suematsu et K. Yuta. Framework for fast identification of community structures in large-scale social networks. Dans *Data Mining for Social Network Data*, pages 149–175. 2010. (Cité page 17.)

N. Lermé, F. Malgouyres, et L. Létocart. Reducing graphs in graph cut segmentation. Dans *ICIP*, pages 3045–3048, 2010. (Cité page 15.)

L. Létocart, A. Nagih, et G. Plateau. Reoptimization in lagrangian methods for the quadratic knapsack problem. *Computers and Operations Research*, 39(1) : 12–18, 2012. (Cité page 110.)

Y. Li, J. Sun, CK. Tang, et HY. Shum. Lazy Snapping. *ACM Transactions on Graphics*, 23(3) :303–308, 2004. (Cité page 12.)

M. Libura. Integer programming problems with inexact objective function. *Control and Cybernetics*, 9 :189–202, 1980. (Cité page 126.)

C. K. Y. Lin, C. K. Chow, et A. Chen. A location-routing-loading problem for bill delivery services. *Comput. Ind. Eng.*, 43(1-2) :5–25, 2002. (Cité page 90.)

S. Lin et B. W. Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21(2) :498–516, 1973. (Cité page 99.)

D. Lindgren, A. Frigyesi, S. Gudjonsson, G. Sjödahl, C. Hallden, G. Chebil, S. Veerla, T. Ryden, W. Månsson, F. Liedberg, et al. Combined gene expression and genomic profiling define two intrinsic molecular subtypes of urothelial carcinoma and gene signatures for molecular grading and outcome. *Cancer research*, 70(9) :3463, 2010. (Cité pages 30 et 31.)

S. Martello et P. Toth. Generalized assignment problems. *Lecture Notes, In Computer Science*, 660 :351–369, 1992. (Cité page 57.)

Ph. Michelon et N. Maculan. Lagrangian methods for 0-1 quadratic knapsack problems. *Discrete Applied Mathematics*, 42 :257–269, 1993. (Cité page 124.)

Ph. Michelon et L. Veilleux. Lagrangian methods for the 0-1 quadratic knapsack problem. *European Journal of Operational Research*, 92 :326–341, 1996. (Cité page 124.)

H. Min. Consolidation Terminal Location-Allocation and Consolidated Routing Problems. *Journal of Business Logistics*, 17(2) :235–263, 1996. (Cité page 91.)

H. Min, V. Jayaraman, et R. Srivastava. Combined location-routing problems : A synthesis and future research directions. *European Journal of Operational Research*, 108(1) :1–15, 1998. (Cité page 90.)

J.D. Murchland. The effect of increasing or decreasing the length of a single arc all shortest distances in a graph. Rapport technique, Report LBS-TNT-26, London graduate school of buisness studies, 1967. (Cité pages 71 et 110.)

Y.-S. Myung. Multicommodity flows in cycle graphs. *Discrete Applied Mathematics*, 154(11) :1615–1621, 2006. (Cité page 7.)

A. Nagih et G. Plateau. A Lagrangean decomposition for 0-1 hyperbolic programming problems. *International Journal of Mathematical Algorithms*, 1 :299–314, 2000a. (Cité page 128.)

A. Nagih et G. Plateau. A partition algorithm for 0-1 unconstrained hyperbolic programming problems. *Investigacion Operativa*, 9(1) :167–178, 2000b. (Cité page 128.)

G. Nagy et S. Salhi. The many-to-many location-routing problem. *TOP : An Official Journal of the Spanish Society of Statistics and Operations Research*, 6 (2) :261–275, 1998. (Cité pages 90 et 91.)

G. Nagy et S. Salhi. Location-routing : Issues, models and methods. *European Journal of Operational Research*, 177(2) :649–672, 2007. (Cité page 90.)

G. L. Nemhauser et M. W. P. Savelsbergh. Functional description of MINTO, a Mixed INTeger Optimizer. Rapport technique, Georgia Institute of Technology, Atlanta, GA, 1996. (Cité page 91.)

J.A. Nepomuceno, A. Troncoso Lora, et J.S. Aguilar-Ruiz. Biclustering of gene expression data by correlation-based scatter search. *BioData Mining*, 4(3), 2011. (Cité pages 26, 28, 29, 38 et 39.)

M. E. J. Newman. Detecting community structure in networks. *EPJB*, 38(2) : 321–330, 2004. (Cité page 17.)

V. P. Nguyen, C. Prins, et C. Prodhon. A multi-start iterated local search with tabu list and path relinking for the two-echelon location-routing problem. *Engineering Applications of Artificial Intelligence*, 25(1) :56–71, 2012a. (Cité page 91.)

V. P. Nguyen, C. Prins, et C. Prodhon. Solving the two-echelon location routing problem by a GRASP reinforced by a learning process and path relinking. *European Journal of Operational Research*, 216(1) :113–126, 2012b. (Cité page 91.)

U. Pape. Changes in networks and ajustement of the length of shortest paths. *Computing*, 12 :357–362, 1974. (Cité pages 71 et 110.)

J. Perl et M. S. Daskin. A Warehouse Location Routing Model. *Transportation Research Part B*, 19(5) :381–396, 1985. (Cité pages 91 et 102.)

D. Pisinger. The quadratic knapsack problem - a survey. *Discrete Applied Mathematics*, 155 :623–648, 2007. (Cité pages 124 et 133.)

D. Pisinger, A.B. Rasmussen, et R. Sandvik. Solution of large-sized quadratic knapsack problems through aggressive reduction. *INFORMS Journal on Computing*, 19(2) :280–290, 2007. (Cité page 124.)

C. Pizzuti, S. E. Rombo, et E. Marchiori. Complex detection in protein-protein interaction networks : a compact overview for researchers and practitioners. Dans *Proceedings of the 10th European conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, EvoBIO'12, pages 211–223, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-29065-7. URL `http://dx.doi.org/10.1007/978-3-642-29066-4_19`. (Cité page 17.)

M.-C. Plateau. Quadratic convex reformulations for quadratic 0-1 programming. *4OR*, 6 :187–190, 2008. (Cité page 147.)

A. Prelic, S. Bleuler, P. Zimmermann, A. Wille, P. Bühlmann, et W. Gruissem. A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, 22(9) :1122–1129, 2006. (Cité pages 26, 28, 29 et 30.)

C. Prins, C. Prodhon, et R. Wolfler Calvo. Solving the capacitated location-routing problem by a GRASP complemented by a learning process and a path relinking. *4OR*, 4(3) :221–238, 2006. (Cité pages 102 et 103.)

F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, et D. Parisi. Defining and identifying communities in networks. *PNAS*, 101(9) :2658–2663, 2004. (Cité page 17.)

V. Radionov. The parametric problem of shortest distances. *Urdmurt USSR. Comuta. Math. Math Phys*, 8 :336–342, 1968. (Cité pages 71 et 110.)

J. Reese. Solution methods for the p-median problem : An annotated bibliography. *Networks*, 48(3) :125–142, 2006. (Cité page 118.)

C.D. Rodrigues, D. Quadri, Ph. Michelon, S. Gueye, et M. Leblond. Applying the t-linearization to the quadratic knapsack problem. Dans *ROADEF 2009, Nancy (France)*, 2009. (Cité page 124.)

M. W. Schaffter. Scheduling with forbidden sets. *Discrete Applied Mathematics*, 72(1-2) :155–166, 1997. (Cité page 110.)

A. Schrijver, P. Seymour, et P. Winkler. The ring loading problem. *SIAM Journal on Discrete Mathematics*, 11(1) :1–14, 1998. (Cité page 7.)

R. Sedgewick. Implementing quicksort programs. *Communications of the Association for Computing Machinery*, 21 :847–857, 1978. (Cité page 128.)

J. K. Seppanen et H. Mannila. Dense itemsets. Dans *KDD*, pages 683–688, 2004. (Cité page 26.)

R. D. Singh. *Location-routing Problems*. PhD thesis, 1998. (Cité page 91.)

A.K. Sinop et L. Grady. Accurate banded graph cut segmentation of thin structures using laplacian pyramids. Dans *MICCAI*, volume 9, pages 896–903, 2006. (Cité page 12.)

O. J. Smith, N. Boland, et H. Waterer. Solving shortest path problems with a weight constraint and replenishment arcs. *Computers & OR*, 39(5) :964–984, 2012. (Cité page 84.)

P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, et B. Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Mol. Biol. Cell*, 9(12) :3273–3297, 1998. URL `http://www.molbiolcell.org/cgi/content/abstract/9/12/3273`. (Cité pages 27 et 28.)

P. Strandmark et F. Kahl. Parallel and distributed graph cuts by dual decomposition. Dans *CVPR*, pages 2085–2092, 2010. (Cité pages 11 et 12.)

N. Stransky, C. Vallot, F. Reyal, I. Bernard-Pierrot, S.G.D. de Medina, R. Segraves, Y. de Rycke, P. Elvin, A. Cassidy, C. Spraggon, et al. Regional copy number–independent deregulation of transcription in cancer. *Nature genetics*, 38(12) : 1386–1396, 2006. (Cité pages 30 et 31.)

C. D. Tarantilis, E. E. Zachariadis, et C. T. Kiranoudis. A hybrid guided local search for the vehicle-routing problem with intermediate replenishment facilities. *INFORMS Journal on Computing*, 20(1) :154–168, 2008. (Cité page 83.)

B. Thiongane. *Réoptimisation dans le dual lagrangien du biknapsack en variables 0-1*. PhD thesis, Université Paris 13 (France), 2003. (Cité page 126.)

B. Thiongane, A. Nagih, et G. Plateau. An Adapted Step Size Algorithm for a 0-1 Biknapsack Lagrangean Dual. *Annals of Operations Research*, 139(1) :353–373, 2005. (Cité pages 71, 75, 110 et 125.)

B. Thiongane, A. Nagih, et G. Plateau. Lagrangean heuristics combined with reoptimization for the 0-1 bidimensional knapsack problem. *Discrete Applied Mathematics*, 154 :2200–2211, 2006. (Cité pages 71, 75, 110, 125 et 126.)

N. Touati. *Amélioration des performances du schéma de la génération de colonnes : Application aux problèmes de tournées de véhicules*. PhD thesis, Université Paris 13 (France), 2008. (Cité page 126.)

N. Touati, L. Létocart, et A. Nagih. Diversification and reoptimization procedures in column generation for the resolution of the acyclic vehicle routing problem with time windows. Dans *Proceedings of INOC 2009 : International Network Optimization Conference, Pise (Italy), 6 pages*, 2009. (Cité page 126.)

A. L. Traud, E. D. Kelsic, P. J. Mucha, et M. A. Porter. Community structure in online ccollegiate social networks. 2008. URL http://arxiv.org/abs/0809.0690v1. (Cité page 17.)

T. Uno et H. Arimura. Ambiguous frequent itemset mining and polynomial delay enumeration. Dans *Proc. PAKDD 2008*, pages 357–368, 2008. (Cité page 26.)

R. Vachani, P. Kubat, A. Shulman, et J. Ward. Multicommodity flows in ring networks. *INFORMS Journal on Computing*, 8(3) :235–242, 1996. (Cité page 7.)

J.M. Valério de Carvalho. Exact solution of bin packing problems using column generation and branch and bound. *Annals of Operations Research*, 86 :629–659, 1999. (Cité page 114.)

J.M. Valério de Carvalho. Lp models for bin packing and cutting stock problems. *European Journal of Operational Research*, 141(2) :253 – 273, 2002. (Cité page 114.)

S. van Hoesel et A. Wagelmans. On the complexity of postoptimality analysis of 0/1 programs. *Discrete Applied Mathematics*, 91(1-3) :251–263, 1999. (Cité page 110.)

V. Ventos et H. Soldano. Alpha Galois lattices : An overview. Dans B. Ganter et R. Godin (Eds), éditeurs, *International Conference on Formal Concept Analysis (ICFCA)*, volume 3403 de *Lecture Notes on Computer Science*, pages 298–313. Springer Verlag, 2005. (Cité page 34.)

H. Zhao, L. Cloots, T. van den Bulcke, Y. Wu, R. de Smet, V. Storms, P. Meysman, K. Engelen, et K. Marchal. Query-based biclustering of gene expression data using probabilistic relational models. *BMC Bioinformatics*, 12(S-1) :S37, 2011. (Cité page 21.)

Titre Reformulation, relaxation et réoptimisation

Résumé Cette synthèse de travaux de recherche s'inscrit dans le cadre de la résolution de problèmes d'optimisation combinatoire en nombres entiers, avec des aspects théoriques, algorithmiques et applicatifs.
Nous nous intéressons aux problèmes de type (multi-)flot et (multi-)coupe, à leur formulation, leur reformulation, leur résolution et les réductions que l'on peut apporter au graphe pour résoudre plus efficacement ces problèmes, avec des applications en traitement d'images et en fouille de données biomédicales notamment.
Ces travaux concernent également la reformulation, la résolution, via des relaxations et des décompositions, et la réoptimisation de problèmes en nombres entiers afin de développer des approches exactes et/ou approchées pour des problèmes de type tournées de véhicules et sac à dos quadratique.

Mots-clés Reformulation, relaxation, réoptimisation, réduction, décomposition, (multi-)flot et (multi-)coupe, tournées de véhicules, sac à dos quadratique

Title Reformulation, relaxation and reoptimization

Abstract This work deals with integer combinatorial optimization problems, with theoretical, algorithmic and application aspects in order to solve such problems.
We are interested in (multi-)flow and (multi-)cut like problems, how to formulate and reformulate these problems and how can we reduce graphs in order to solve these problems, with applications in image processing and biomedical data mining.
This work also investigates how to reformulate and solve, exactly and/or heuristically, via relaxation, decomposition and reoptimization approaches, vehicle routing like problems and quadratic knapsack problems.

Keywords Reformulation, relaxation, reoptimization, reduction, decomposition, (multi-)flow et (multi-)cut, vehicle routing problem, quadratic knapsack problem