Titre: Title:	Problème de tarification sur un réseau
Auteur: Author:	Fabien Cirinei
Date:	2007
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Cirinei, F. (2007). Problème de tarification sur un réseau [Ph.D. thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/7794/

UNIVERSITÉ DE MONTRÉAL

PROBLÈME DE TARIFICATION SUR UN RÉSEAU

FABIEN CIRINEI
DÉPARTEMENT DE MATHÉMATIQUES ET DE GÉNIE INDUSTRIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIÆ DOCTOR
(MATHÉMATIQUES DE L'INGÉNIEUR)
FÉVRIER 2007

Your file Votre référence
ISBN: 978-0-494-24538-5
Our file Notre référence
ISBN: 978-0-494-24538-5

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée:

PROBLÈME DE TARIFICATION SUR UN RÉSEAU

présentée par: CIRINEI Fabien

en vue de l'obtention du diplôme de: Philosophiæ Doctor

a été dûment acceptée par le jury d'examen constitué de:

M. LANGEVIN André, Ph.D., président

M. SAVARD Gilles, Ph.D., membre et directeur de recherche

M. SEMET Frédéric, Docteur, membre et codirecteur de recherche

Mme. BROTCORNE Luce, Docteur, membre et codirecteur de recherche

M. GENDREAU Michel, Ph.D., membre

M. VANDERBECK François, Ph.D., membre externe

À mes parents,
et à ma soeur

REMERCIEMENTS

Je ne peux ignorer la contribution de ces dizaines de personnes sans qui la réalisation de ce travail n'aurait jamais été possible. Je profite de ces quelques lignes pour les remercier.

Tout d'abord, je tiens à remercier vivement mes directeurs Luce Brotcorne, Gilles Savard et Frédéric Semet de l'attention et du soutien qu'ils ont portés à mon travail de doctorant. Je remercie également Patrice Marcotte pour les nombreuses discussions qui ont permis d'approfondir de nombreux points.

Cette cotutelle de thèse m'a donné la chance de mener mes recherches au sein de deux laboratoires : le Centre de Recherche sur les Transports de Montréal et le Laboratoire d'Automatique de Mécanique et d'Informatique Industrielles et Humaines de l'Université de Valenciennes et du Hainaut-Cambrésis. Que les membres respectifs de ces deux équipes trouvent ici tout ma reconnaissance et tout particulièrement le groupe de recherche dirigé par Patrice et Gilles avec qui le travail est devenu un plaisir.

Merci à mes amis pour le soutien et les beaux moments qui ont agrémenté ces dernières années. En particulier, je veux témoigner toute ma gratitude à Pierre Noël. Sa gentillesse et sa *Bombita* m'ont permis de vivre de merveilleux séjours à Montréal avec mes amis du *bord de l'aile.*

Je ne pourrais terminer sans me tourner vers ma famille que je remercie pour leur dévouement et leur présence tout au long de mes études. Je sais que les hiéroglyphes qui décorent ces pages ne vous inspireront certainement pas mais sachez que vous y êtes pour beaucoup.

RÉSUMÉ

Dans cette thèse, nous étudions le problème de la détermination d'un ensemble de tarifs optimaux sur des arcs d'un réseau multi-produits. Plus précisément, nous étudions la situation où un agent (meneur) cherche à maximiser ses revenus en fixant des tarifs sur un sous-ensemble d'arcs d'un réseau, en tenant compte de la réaction d'un utilisateur (suiveur) qui souhaite acheminer à moindre coût un ensemble de produits sur le réseau. Ce processus de décision séquentiel et non coopératif peut être modélisé comme un programme mathématique à deux niveaux.

Tout d'abord, nous proposons une méthode exacte basée sur une énumération intelligente des solutions du problème du suiveur. Les améliorations que nous avons apportées à cette énumération ont permis de définir de nouvelles bornes supérieures sur le revenu du meneur. Les résultats numériques mettent en évidence le gain de performances de cette méthode sur les instances où les tarifs sont non contraints.

D'autre part, nous avons développé un algorithme de recherche locale qui a été repris au sein d'une méthode de recherche avec tabous pour la résolution du problème de tarification sur un réseau. Elle exploite la structure de réseau sous-jacente du problème du suiveur pour explorer efficacement l'espace des solutions. Cette méthode fournit des solutions situées à moins de 1% des solutions optimales en des temps de calcul très courts.

Soulignons que l'efficacité de nos méthodes de résolution pour le problème de tarification sur un réseau repose sur la résolution du problème d'optimisation inverse. Ce problème permet de déterminer les tarifs maximisant le revenu du meneur tout en reproduisant une réaction du suiveur connue. Une méthode de génération de colonnes est proposée pour résoudre efficacement ce problème et améliorer l'évaluation des solutions du problème du suiveur.

ABSTRACT

This thesis is concerned with the problem of identifying optimal tolls over a multi-commodity network. More precisely, we are interested in the situation whereby an economic agent, called the "Leader", seeks to maximize network revenue by imposing tolls on a subset of arcs, while taking into account the reaction of network users, called "Followers", who strive to transport products over the network at minimal cost.

We first propose an exact method based on a clever enumeration of the Follower's problem solutions. Improvements to the enumeration process also allow to identify better upper bounds on the Leader's revenue. Numerical results illustrate the performance of this approach, especially on instances where tolls are unconstrained.

Secondly, we present a local search algorithm and describe its integration into a tabu-search method. This algorithm exploits the underlying structure of the Follower's problem to more efficiently explore the solution space. The tabu-search method is shown to provide near-optimal (less than 1%) solutions within a very short computation time.

The great efficiency of our methods is achieved through the resolution of an inverse optimization problem. This formulation allows to quickly identify an optimal toll scheme given a known Follower's reaction. A column-generation approach is proposed to adress this problem and obtain better Follower's problem solutions.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ANNEXES

INTRODUCTION

Depuis quelques années, de nombreuses industries ont été amenées à appliquer des politiques de gestion du revenu afin de rester compétitif dans un contexte concurrentiel accru. En effet, de nouveaux facteurs et l'évolution des technologies telles que l'internet permettent un accès temps réel à de nombreuses informations et ainsi une connaissance approfondie du comportement des consommateurs.

Les méthodes de gestion du revenu sont apparues au début des années 80 aux États-Unis lors de la déréglementation des transports aériens. Pour se maintenir sur le marché, les compagnies aériennes ont compris qu'il valait mieux brader un siège inoccupé plutôt que de le laisser vide. C'est ainsi qu'est née une nouvelle politique de gestion commerciale décrite de façon farfelue par "Sell at the right price to the right customer at the right time". Elle est basée sur l'application de techniques visant à optimiser le profit généré par la vente d'un produit ou d'un service en exploitant une modélisation et une prévision du comportement de la demande. American Airlines a augmenté ses revenus d'environ 1,5 milliard entre 1989 et 1991 grâce à ce seul mécanisme. Aujourd'hui, cette pratique s'étend dans d'autres domaines tels que l'hôtellerie, la location de voiture, les télécommunications etc.

De façon classique la gestion du revenu comprend quatre éléments : la prévision de la demande, la surréservation, l'allocation de capacité et la tarification. Beaucoup de recherches ont été entreprises pour améliorer les techniques des trois premiers axes mais étrangement la problématique de la détermination des tarifs a longtemps été négligée. Néanmoins depuis quelques années, la tarification est considérée comme un axe important dans les méthodes de gestion du revenu car une mauvaise gestion des tarifs implique souvent des pertes de revenu importantes.

Labbé et al. (1998) ont proposé une nouvelle approche de la tarification tenant

compte de la réaction des consommateurs. Pour cela, ils considèrent deux agents de décision interagissant de façon séquentielle et hiérarchique. Un premier agent, appelé meneur, désire établir une politique tarifaire sur un ensemble de biens ou de services qu'il offre. De façon à maximiser son profit, il est important qu'il prenne en considération la réaction d'un autre agent de décision, appelé suiveur (les consommateurs), qui lui veut minimiser sa désutilité. Ainsi, un plan tarifaire pour le meneur est donc composé de tarifs d'une part suffisamment élevés pour générer des profits et d'autre part, pas trop élevés, pour ne pas dissuader le suiveur d'utiliser les biens ou les services offerts par le meneur. Pour tenir compte de l'interaction hiérarchique entre le meneur et le suiveur, Labbé et al. (1998) ont formulé ce problème sous forme d'un problème de programmation mathématique à deux niveaux. Ce problème a été prouvé comme étant *fortement NP-difficile* par Roch et al. (2003) et Grigoriev et al. (2005).

Cette approche peut être appliquée dans de nombreux domaines comme pour l'allocation et la tarification de sièges pour le transport aérien (Côté et al. (2003)), la tarification des autoroutes (Dewez (2004)). Mais elle permet de contrôler des situations grâce à la tarification comme pour améliorer le trafic lié aux transports de matières dangereuses afin de réduire le risque d'accident (Brotcorne et al. (2006b)).

Dans cette thèse, nous étudions le problème de tarification sur un réseau (PTR). Plus précisément, nous considérons la situation où le meneur possède un ensemble de tronçons sur ce réseau pour lequel il désire déterminer les tarifs et le suiveur doit acheminer des produits d'une origine vers une destination à moindre coût via le réseau.

Le but de cette thèse est de proposer des méthodes de résolution exacte ou approchée basées sur la structure de réseau sous-jacente pour le PTR. Elle s'organise comme suit.

Dans le premier chapitre, nous présentons les problèmes mathématiques à deux niveaux et nous donnons une revue de la littérature pour le problème de tarification.

Dans le second chapitre, nous nous intéressons plus en détails au PTR. En particulier, nous présentons la formulation à deux niveaux introduite par Labbé et al. (1998) et nous développerons quelques propriétés de ce problème.

Dans le troisième chapitre, nous étudions le problème d'optimisation inverse qui constitue une étape majeure dans les méthodes de résolution. Après avoir formulé ce problème, nous développons une méthode de génération de colonnes pour le résoudre. Finalement, nous concluons ce chapitre en présentant quelques extensions de ce problème et leurs méthodes de résolution.

Le quatrième chapitre est consacré à une méthode exacte basée sur le concept de K-chemins proposé par Didi et al. (1999). Nous présentons d'abord quelques améliorations pour la génération des K-chemins et nous définissons ensuite de nouvelles bornes supérieures sur le revenu d'un K-chemin. Nous concluons ce chapitre en analysant les performances de cette méthode sur un grand nombre d'instances.

Dans le cinquième chapitre, nous proposons une méthode de recherche locale explorant les optima locaux du PTR. Plusieurs stratégies pour les différentes composantes de cette méthode sont testées et analysées afin d'obtenir une méthode de recherche locale la plus efficace possible.

Le sixième chapitre est consacré à une méthode de recherche avec tabous basée sur la méthode de recherche locale du chapitre précédent. Après avoir présenté en détails cette méthode, une phase d'étalonnage est effectuée pour déterminer les valeurs des paramètres les plus judicieuses. Finalement, la méthode de recherche avec tabou est validée numériquement sur un ensemble d'instances de grande taille.

CHAPITRE 1

REVUE DE LITTÉRATURE

Le problème de tarification sur un réseau fait intervenir deux agents de décisions. Le premier, appelé meneur, décide d'une politique tarifaire appliquée sur une partie du réseau. Le second, appelé suiveur, utilise le réseau en prenant en compte la politique tarifaire proposée par le meneur. La résolution de ce problème consiste donc, pour le meneur, à déterminer la meilleure politique tarifaire afin de maximiser son revenu sachant qu'il connaît la réaction du suiveur par rapport aux tarifs. Les problèmes de programmation mathématique à deux niveaux constitue un outil particulièrement bien adapté pour modéliser le type d'interaction hiérarchique et séquentielle présent dans ce problème.

Depuis son introduction dans les années 70, la programmation mathématique à deux niveaux suscite un intérêt croissant. Sous sa forme la plus générale, le problème de programmation mathématique à deux niveaux (PDN) s'écrit

$$\min_{x} \quad f_1(x,y) \tag{1.1}$$

$$\text{s.c.} \quad g_1(x,y) \leq 0, \tag{1.2}$$

$$y = \arg\min_{y'} \quad f_2(x,y') \tag{1.3}$$

$$\text{s.c.} \quad g_2(x,y') \leq 0. \tag{1.4}$$

Les contraintes (1.3) et (1.4) définissent le problème du second niveau. Pour un vecteur x fixé, le vecteur y appartient implicitement à l'ensemble $R(x) = \{y : y \in \arg\min_{y'}\{f_2(x,y') : y' \in S(x)\}\}$ où $S(x) = \{y' : g_2(x,y') \leq 0\}$.

La présence de deux niveaux d'optimisation caractérise ce modèle. Le problème

de premier niveau est défini par les équations (1.1) et (1.2) et celui du deuxième niveau par les équations (1.3) et (1.4). L'appartenance de y à $R(x)$ implique que le vecteur y est déterminé après que le vecteur x ait été fixé. C'est pourquoi, le vecteur x est souvent appelé vecteur du premier niveau et le vecteur y du second niveau.

À la section 1.1, nous présentons une courte revue de littérature des problèmes mathématiques à deux niveaux. Nous particularisons ensuite cette revue bibliographique aux problèmes de tarification sur un réseau à la section 1.2.

1.1 Programmation mathématique à deux niveaux

La première référence à la programmation mathématique à deux niveaux date de 1973 dans un article de Bracken et McGill (1973). Cependant, Candler et Norton (1977) sont les premiers à utiliser les termes de programmation mathématique à deux niveaux. Il faut attendre une dizaine d'années pour que des chercheurs, travaillant sur la théorie des jeux, s'intéressent aux concepts fondamentaux de ce type de problèmes et proposent des algorithmes pour les résoudre.

Le jeu de Stackelberg (Von Stackelberg (1952)) constitue le problème de base de la programmation mathématique à deux niveaux. La résolution de ce jeu à deux joueurs (nommés 1 et 2) consiste à trouver la meilleure stratégie du joueur 1 sachant que celui-ci applique le premier sa stratégie et connaît la réaction du joueur 2. Ce dernier choisit sa stratégie après que le joueur 1 ait dévoilé la sienne. Nous supposons que pour chaque décision du joueur 1, il existe une réponse pour le joueur 2. Comme nous le verrons à la section suivante, le problème de tarification étudié dans cette thèse est un cas particulier du jeu de Stackelberg où le joueur 1 est le meneur et le joueur 2 est le suiveur.

En introduisant le vecteur de décision du meneur x et le vecteur de décision du suiveur y, une formulation plus compact du PDN est donnée par :

$$\begin{aligned} \min_{x} \quad & f_1(x,y) \\ \text{s.c.} \quad & g_1(x,y) \leq 0, \\ \min_{y} \quad & f_2(x,y) \\ \text{s.c.} \quad & g_2(x,y) \leq 0, \end{aligned}$$

où f_1 (resp. f_2) est la fonction de objectif du meneur (resp. suiveur), $g_1(x,y) \leq 0$ (resp. $g_2(x,y) \leq 0$) est un ensemble de contraintes devant être satisfaites par le meneur (resp. suiveur). L'ensemble $I = \{(x,y) : g_1(x,y) \leq 0, y \in R(x)\}$, appelé *région induite*, contient les solutions réalisables du PDN.

Une première difficulté dans la définition du PDN surgit lorsque la réponse du suiveur n'est pas unique pour certain vecteur x, c'est-à-dire que l'ensemble $R(x)$ n'est pas un singleton. Dans ce cas, deux approches peuvent être considérées. La première est une approche *optimiste*. Elle suppose que le suiveur coopère avec le meneur pour minimiser la fonction $f_1(x,y)$ pour un ensemble $R(x)$ de réactions possibles. Le PDN est alors redéfini par :

$$\begin{aligned} \min_{x,\, y^*} \quad & f_1(x,y^*) \\ \text{s.c.} \quad & g_1(x,y^*) \leq 0, \\ & y^* \in R(x). \end{aligned}$$

L'autre approche, la pessimiste, considère qu'il n'y a pas de coopération entre le meneur et le suiveur. Dans ce cas, le meneur doit envisager le pire de la part du

suiveur et le PDN se récrit :

$$\begin{aligned} \min_{x,\,y} \quad & \max_{y^*} f_1(x, y^*) \\ \text{s.c.} \quad & g_1(x, y^*) \leq 0, \\ & y^* \in R(x). \end{aligned}$$

Dans la suite, nous ne considérons que le cas optimiste. Pour plus de détails sur le cas pessimiste, nous renvoyons le lecteur aux études de Bard (1991), Loridan et Morgan (1989a,b), Ishizuka et Aiyoshi (1992).

Les problèmes à deux niveaux sont intrinsèquement difficiles à résoudre. Même dans la version la plus simple, Hansen et al. (1992) ont montré que les problèmes à deux niveaux linéaire-linéaire (les fonctions f_1, f_2, g_1 et g_2 sont linéaires) appartiennent à la classe des problèmes fortement NP-difficiles. De plus, Vicente et al. (1994) ont montré que prouver l'optimalité locale d'une solution d'un problème à deux niveaux linéaire-linéaire est un problème NP-difficile. La difficulté majeur des problèmes à deux niveaux est que la région induite I est souvent non convexe et peut être vide ou déconnectée en présence de contraintes du premier niveau.

De nombreux auteurs ont proposé des conditions d'optimalité pour ce problème soit en se basant sur une reformulation de celui-ci, soit en utilisant les concepts de l'optimisation non différentiable ou soit en exploitant la géométrie de la région induite. Pour plus de détails, nous renvoyons le lecteur à l'article de Dempe (2002). Notons toutefois qu'aucune des conditions proposées ne peut être appliquée en pratique comme critère d'arrêt dans les méthodes numériques.

La classe des problèmes à deux niveaux linéaire-linéaire a reçu une attention particulière et de nombreuses méthodes de résolution ont été proposées. Vu que pour ce type de problèmes, la solution optimale est située en un point extrême de la région

$S(x)$, beaucoup de méthodes sont basées sur une énumération des solutions extrémales (Bialas et Karwan (1984); Candler et Townsley (1982); Tuy et al. (1993)). Une autre approche consiste à remplacer le problème du second niveau par ses conditions d'optimalité primale et duale pour obtenir une formulation à un niveau avec des contraintes disjonctives. Des méthodes de séparation et d'évaluation ont été développées pour résoudre cette formulation (Bard et Falk (1982); Fortuny-Amat et McCarl (1981)). En combinant les méthodes de séparation et d'évaluation et les méthodes de pénalité, Hansen et al. (1992) ont proposé un algorithme capable de résoudre des instances de tailles moyennes. Júdice et Faustino (1992) ont quant à eux proposé un algorithme basé sur la théorie des pivots complémentaires. Pour plus de détails sur ces méthodes de résolution de cette classe de problème, nous renvoyons le lecteur vers la revue de littérature proposé par Ben-Ayed (1993).

Pour les problèmes à deux niveaux non linéaire, plusieurs de méthodes de résolution ont été proposées mais en imposant certaines hypothèses. Ces méthodes se divisent en trois catégories. Seule la première catégorie permet une résolution globale alors que les deux autres se limitent à la découverte de solutions stationnaires ou des optima locaux. La première catégorie est une extension des méthodes de séparation et d'évaluation mentionnée précédemment mais où le problème du suiveur est un problème quadratique en y avec contraintes linéaires (Al-Khayal et al. (1992); Edmunds et Bard (1991); Thoai et al. (2002)). La deuxième catégorie rassemble les méthodes de descentes pour des problèmes où la réaction du suiveur est unique. Ainsi, cette réaction peut être définie en fonction des variables du premier niveau et des directions de descentes sont calculées (Gauvin et Savard (1994); Kolstad et Lasdon (1990)). La dernière catégorie réunit les méthodes de pénalité (Aiyoshi et Shimizu (1984); Colson et al. (2005a)). Pour plus de détails, nous invitons le lecteur à consulter la revue bibliographique écrite par Colson et al. (2005b).

1.2 Modèles à deux niveaux bilinéaire pour des problèmes de tarification sur un réseau

La première formulation du PTR sous forme d'un problème à deux niveaux bilinéaire-bilinéaire a été proposée en 1998 par Labbé et al. (1998).

Le réseau est modélisé par un graphe $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, où $\mathcal{N}$ représente l'ensemble des noeuds et $\mathcal{A}$ l'ensemble des arcs. L'ensemble des arcs est partitionné en deux sous-ensembles $\mathcal{A}_1$ et $\mathcal{A}_2$ où $\mathcal{A}_1$ contient les arcs pour lesquels les tarifs doivent être déterminés, et, $\mathcal{A}_2$ l'ensemble des arcs n'étant pas sous la conduite du meneur. À chaque arc non tarifable ($a \in \mathcal{A}_2$) est associé un coût unitaire fixe d_a. À chaque arc tarifable ($a \in \mathcal{A}_1$) est associé un coût unitaire fixe c_a et un tarif unitaire T_a fixé par le meneur.

La demande sur le réseau est représentée par un ensemble $\mathcal{K}$ de produits. À chaque produit $k \in \mathcal{K}$ est associé une paire origine-destination $(o(k), d(k))$ et une demande n^k. Ainsi, les composantes du vecteur de demande b^k associées à chaque produit $k \in \mathcal{K}$ sont définies par :

$$b_i^k = \begin{cases} n^k & \text{si } i = o(k), \\ -n^k & \text{si } i = d(k), \\ 0 & \text{sinon.} \end{cases}$$

Pour tout produit $k \in \mathcal{K}$, la variable x_a^k (respectivement y_a^k) représente le flux du produit k présent sur l'arc tarifable $a \in \mathcal{A}_1$ (respectivement non tarifable $a \in \mathcal{A}_2$).

Le problème de tarification sur un réseau peut être formulé comme :

$$\text{PTRA1} : \max_{T,x,y} \quad \sum_{a\in\mathcal{A}_1} T_a \sum_{k\in\mathcal{K}} x_a^k \tag{1.5}$$

$$\min_{x,y} \quad \sum_{a\in\mathcal{A}_1}\sum_{k\in\mathcal{K}}(T_a + c_a)x_a^k + \sum_{a\in\mathcal{A}_2}\sum_{k\in\mathcal{K}} d_a y_a^k \tag{1.6}$$

$$\text{s.c.} \quad \sum_{a\in i^-}(x_a^k + y_a^k) - \sum_{a\in i^+}(x_a^k + y_a^k) = b_i^k, \qquad \forall k \in \mathcal{K}, \forall i \in \mathcal{N}, \tag{1.7}$$

$$x_a^k \geq 0, \qquad \forall a \in \mathcal{A}_1, \forall k \in \mathcal{K}, \tag{1.8}$$

$$y_a^k \geq 0, \qquad \forall a \in \mathcal{A}_2, \forall k \in \mathcal{K}, \tag{1.9}$$

où i^- (resp. i^+) représente l'ensemble des arcs ayant pour origine (resp. destination) le noeud i, c'est-à-dire, $i^- = \{(i,j) \in \mathcal{A}\}$ (resp. $i^+ = \{(j,i) \in \mathcal{A}\}$).

Le meneur maximise son revenu (1.5) qui est la somme sur l'ensemble des produits et l'ensemble des arcs du meneur du produit entre le flux et le tarif fixé par le meneur. Le suiveur minimise ses coûts d'utilisation du réseau (1.6). Les contraintes (1.7) sont les contraintes de conservation de flux sur le réseau pour chaque produit. Les contraintes (1.8) et (1.9) imposent la non-négativité des fluxs. Pour un niveau de tarifs fixé, le flux de chaque produit k est affecté au plus court chemin entre l'origine $o(k)$ et la destination $d(k)$.

Les principales propriétés du problème de tarification et la particularisation de celles-ci au PTR ont été présentées par Labbé et al. (1998). Nous détaillerons ces propriétés dans le chapitre suivant.

Récemment, le PTR a été prouvé comme étant *fortement NP-difficile* par Roch et al. (2003) et par Grigoriev et al. (2005). Ces résultats sont basés sur une réduction polynomiale du PTR sous forme d'un problème 3-SAT. Rappelons que le problème 3-SAT consiste à déterminer si une formule booléenne sous forme normale conjonctive composée de 3 littéraux par clause peut être satisfaite. La première réduction

(Roch et al. (2003)) transforme le problème 3-SAT en un PTR mono-produit alors que l'autre (Grigoriev et al. (2005)) se ramène à un PTR multi-produits.

Pour la résolution exacte du PTR, des méthodes exactes basées sur des formulations par arcs et par chemins ont été proposées. La première formulation linéaire mixte par arcs (MIP) a été proposée par Labbé et al. (1998) :

$$
\begin{aligned}
\text{MIP} : \max_{T,x,y} \quad & \sum_{k\in\mathcal{K}}\sum_{a\in\mathcal{A}_1} n^k T_a^k \\
\text{s.c.} \quad & \sum_{a\in i^-}(x_a^k + y_a^k) - \sum_{a\in i^+}(x_a^k + y_a^k) = e_i^k, && \forall k\in\mathcal{K}, \forall i\in\mathcal{N}, \\
& \sum_{a\in\mathcal{A}_1}(T_a^k + c_a x_a^k) + \sum_{a\in\mathcal{A}_2} d_a y_a^k = \sum_{i\in\mathcal{N}} \lambda_i^k e_i^k, && \forall k\in\mathcal{K}, \\
& \lambda_j^k - \lambda_i^k \le c_a + T_a, && \forall k\in\mathcal{K}, \forall a=(i,j)\in\mathcal{A}_1, \\
& \lambda_j^k - \lambda_i^k \le d_a, && \forall k\in\mathcal{K}, \forall a=(i,j)\in\mathcal{A}_2, \\
& -\underline{M}_a^k x_a^k \le T_a^k \le \overline{M}_a^k x_a^k, && \forall k\in\mathcal{K}, \forall a\in\mathcal{A}_1, \\
& -\underline{N}_a(1-x_a^k) \le T_a - T_a^k \le \overline{N}_a(1-x_a^k), && \forall k\in\mathcal{K}, \forall a\in\mathcal{A}_1, \\
& x_a^k \in \{0,1\}, && \forall k\in\mathcal{K}, \forall a\in\mathcal{A}_1, \\
& y_a^k \ge 0, && \forall k\in\mathcal{K}, \forall a\in\mathcal{A}_2, \\
& \lambda_i^k \text{ libre}, && \forall k\in\mathcal{K}, \forall i\in\mathcal{N},
\end{aligned}
$$

où la variable λ_i^k est la variable duale du problème de suiveur associées à la contrainte de conservation de flux (1.7) au noeud i pour le produit k et $\underline{M}_a^k$, $\overline{M}_a^k$, $\overline{N}_a$ et $\underline{N}_a$ sont des constantes suffisamment grandes. Notons que $b_i^k = n^k e_i^k, \forall i \in \mathcal{N}, \forall k \in \mathcal{K}$ Pour plus de détails sur cette formulation, nous renvoyons le lecteur à la section 2.5.

La résolution de la formulation MIP par un logiciel commercial comme CPLEX ne permet pas de résoudre des instances de grande taille à cause de la mauvaise qualité de la relaxation linéaire. Des améliorations ont été apportées à cette formulation

mais seulement lorsque les tarifs sont positifs.

Dewez (2004) a fixé la valeur des constantes $\underline{M}^k$, $\overline{M}^k$, $\overline{N}$ et $\underline{N}$ en calculant des bornes supérieures sur les tarifs et a proposé des coupes valides. Les expérimentations numériques montrent que l'ajustement des constantes divise par deux la valeur du saut de dualité à la racine dans l'arbre de séparation et d'évaluation alors que les coupes valides réduisent le nombre de noeuds explorés et permettent de réduire les temps de calcul.

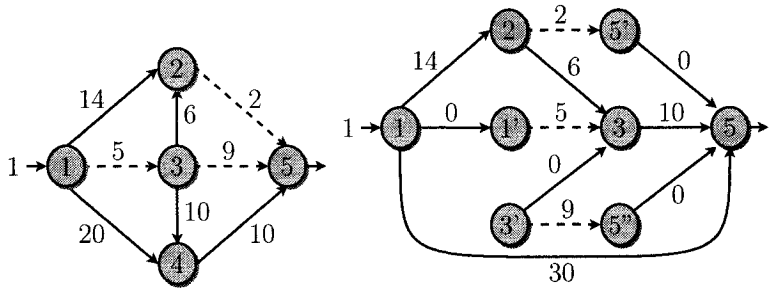
De leur coté, Bouhtou et al. (2003) ont proposé un prétraitement du réseau par produit. Il se compose d'une phase de transformation topologique et d'une phase d'élimination d'arcs. Bien que la première soit indépendante des tarifs, la deuxième n'est valable que lorsque des tarifs positifs sont requis.

La phase de transformation topologique consiste à remplacer les chemins non tarifables reliant une paire de noeuds (a, b) par un arc non tarifable de coût est égal à la valeur du plus court chemin non tarifable de a à b. Les paires de noeuds considérées sont :

- le noeud v d'un arc tarifable (u, v) au noeud u' d'un arc tarifable (u', v'),
- l'origine $o(k)$ d'un produit k au noeud u d'un arc tarifable (u, v),
- le noeud v d'un arc tarifable (u, v) à la destination $d(k)$ d'un produit k,
- l'origine $o(k)$ à la destination $d(k)$ d'un produit k.

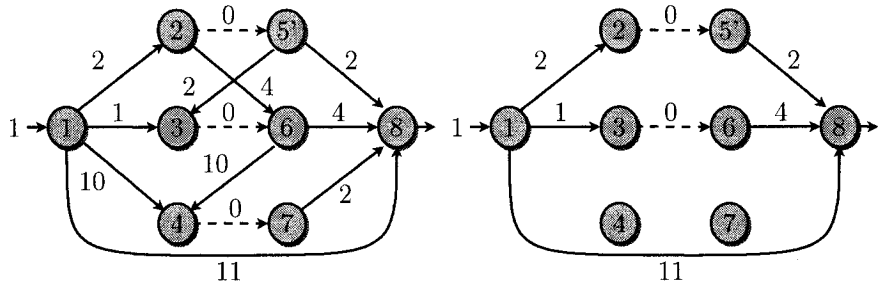
Le résultat de cette transformation est illustré à la figure 1.1 sur le réseau 1.1(a) avec un seul produit $(1, 5)$. Les arcs en pointillés représentent les arcs du meneur. Cette transformation met en valeur l'aspect combinatoire de ce problème.

Dans la deuxième phase, des règles de dominance sur le coût des chemins, renforcées par Dewez (2004), sont appliquées pour éliminer les arcs inutiles de chaque produit. Par exemple, le réseau de la figure 1.2(b) est le résultat d'une réduction appliquée



(a) avant transformation (b) après transformation

Figure 1.1 Exemple de transformation topologique d'un réseau



(a) avant réduction (b) après réduction

Figure 1.2 Exemple de réduction d'un réseau

au réseau 1.2(a). L'arc tarifable $(4, 7)$ est supprimé car la somme du coût du plus court chemin de 1 à 4 et du plus court chemin de 7 à 8 est supérieure au plus court chemin non tarifable de 1 à 8. Il est donc impossible de fixer un tarif qui rende l'arc attractif car les tarifs sont positifs. La suppression de l'arc $(4, 7)$ implique aussi celle des arcs $(1, 4)$, $(7, 8)$ et $(6, 4)$.

En pratique, ce prétraitement réduit le nombre d'arcs potentiellement utilisables de chaque produit et par conséquence le nombre de variables dans la formulation MIP que nous dénoterons MIP+. Il en résulte une réduction des temps de calcul grâce à la résolution de sous-problèmes de plus petite taille mais aucune amélioration de la qualité de la relaxation linéaire.

Bouhtou et al. (2003) ont proposé une autre formulation exploitant la structure de

plus court chemin présente dans le problème du suiveur. En posant P_k l'ensemble des chemins reliant l'origine $o(k)$ à la destination $d(k)$ du produit k, leur formulation linéaire mixte par chemin s'écrit :

$$\text{PMIP} : \max_{T,h,r} \quad \sum_{k\in\mathcal{K}} n^k \sum_{p\in P_k} \sum_{a\in\mathcal{A}_1\cup p} r_{pa} \tag{1.10}$$

$$\text{s.c.} \quad \sum_{p\in P_k} h_p = 1, \qquad \forall k \in \mathcal{K}, \tag{1.11}$$

$$l_p \geq \sum_{q\in P_k} \Big(\sum_{a\in\mathcal{A}_2\cup q} h_q d_a + \sum_{a\in\mathcal{A}_1\cup q} r_{qa} \Big), \quad \forall k \in \mathcal{K}, \forall p \in P_k, \tag{1.12}$$

$$l_p = \sum_{a\in\mathcal{A}_2\cup p} d_a + \sum_{a\in\mathcal{A}_1\cup q} (c_a + T_a), \quad \forall k \in \mathcal{K}, \forall p \in P_k, \tag{1.13}$$

$$r_{pa} - T_a \leq (1 - h_p)M, \qquad \forall k \in \mathcal{K}, \forall p \in P_k, \forall a \in \mathcal{A}_1 \cup p, \tag{1.14}$$

$$r_{pa} - T_a \geq -(1 - h_p)M, \qquad \forall k \in \mathcal{K}, \forall p \in P_k, \forall a \in \mathcal{A}_1 \cup p, \tag{1.15}$$

$$r_{pa} \leq h_p M, \qquad \forall k \in \mathcal{K}, \forall p \in P_k, \forall a \in \mathcal{A}_1, \tag{1.16}$$

$$h_p \in \{0, 1\}, \qquad \forall k \in \mathcal{K}, \forall p \in P_k, \tag{1.17}$$

$$T_a \geq 0, \qquad \forall a \in \mathcal{A}_1, \tag{1.18}$$

$$r_{pa} \geq 0, \qquad \forall k \in \mathcal{K}, \forall p \in P_k, \forall a \in \mathcal{A}_1, \tag{1.19}$$

où la variable T_a indique le tarif de l'arc $a \in \mathcal{A}_1$. La variable r_{pa} est égale à T_a si le suiveur utilise le chemin $p \in P_k$ du produit $k \in \mathcal{K}$ passant par l'arc $a \in \mathcal{A}_1$ et 0 sinon. La variable h_p est égale à 1 si le chemin $p \in P_k$ d'un produit $k \in \mathcal{K}$ est utilisé par le suiveur et 0 sinon. Finalement, la variable l_p représente le coût du chemin $p \in P_k$ d'un produit $k \in \mathcal{K}$.

La fonction objectif (1.10) maximise le revenu du meneur. Les contraintes (1.11) imposent l'utilisation d'un et d'un seul chemin pour chaque produit. Les contraintes (1.12) et (1.13) impliquent que le chemin utilisé par chaque produit par le suiveur soit le plus court. Les contraintes (1.14), (1.15) et (1.16) permettent le respect de la définition des variables r_{pa}.

Cette formulation comporte un nombre exponentielle de contraintes et de va-

riables. Cependant, l'utilisation du prétraitement sur le réseau réduit suffisamment le nombre de chemins et cette formulation peut être résolue par un logiciel commercial. Les résultats numériques obtenus en résolvant la formulation PMIP sont semblables à ceux obtenus par la formulation MIP+.

Didi et al. (1999) ont proposé une approche exacte différente des précédentes. Elle est basée sur une énumération intelligente des solutions du problème du suiveur. Cette méthode permet de résoudre très efficacement les instances ayant moins de 10 produits. Malheureusement, dès que le nombre de produits augmente, ses performances deviennent très mauvaises étant donné l'explosion combinatoire du nombre de solutions et la mauvaise qualité des bornes supérieures utilisées pour guider l'exploration. Pour plus de détails, nous renvoyons le lecteur au chapitre 4 où nous détaillons cette méthode et proposons des améliorations.

Comme mentionné ci-dessus, les méthodes exactes référencées ne sont pas assez efficaces pour résoudre des instances de grande taille du PTR. Il est donc nécessaire d'utiliser des méthodes heuristiques pour résoudre ces instances. Brotcorne et al. (2000, 2001) ont proposées plusieurs heuristiques pour résoudre le PTR lorsque les tarifs sont bornés dans un intervalle.

Tout d'abord, trois heuristiques primales-duales (Brotcorne et al. (2000)) ont été proposées pour résoudre le problème de tarification lorsque le problème du suiveur est un problème de transbordement mono-produit. Plus précisément, la demande du produit est répartie parmi un ensemble de sites d'origine et doit être acheminée vers un ensemble de sites de destination. Les heuristiques développées sont basées sur la reformulation du PTTM sous forme d'un programme bilinéaire sur un niveau et sur la pénalisation de contraintes non convexes dans la fonction objectif. Les deux premières heuristiques sont une généralisation de l'heuristique primale-duale proposée par Gendreau et al. (1996) et la troisième est basée sur la résolution

du problème pénalisé par la méthode de Gauss-Seidel. Les résultats numériques mettent en évidence l'efficacité des heuristiques en termes de qualité de solutions et de temps de calcul sur des instances comportant plus de 40 points de demande.

Brotcorne et al. (2001) ont proposé deux heuristiques pour le PTR. La première est une recherche séquentielle par arcs exploitant la procédure de résolution proposée par Labbé et al. (1998) pour le PTR ayant un seul arc tarifable. Le deuxième est une approche primale-duale similaire à celle présentée précédement. Elle est aussi basée sur une reformulation sous forme d'un programme bilinéaire sur un niveau du PTR et sur l'utilisation d'une pénalité quadratique maintenant des tarifs identiques pour chaque produit. Cette formulation non linéaire est résolue par l'algorithme de Frank-Wolfe. Les expérimentations numériques montrent que l'heuristique primale-duale fournit des solutions situées en moyenne à 1.5% de la solution optimale alors que la recherche séquentielle est à 7%. Les temps de calcul des deux méthodes sont négligeables par rapport à ceux des méthodes exactes.

Nous finissons cette revue de littérature par une présentation non exhaustive de quelques applications du PTR.

La première application a été proposée par Côté et al. (2003) pour le problème d'allocation et de la tarification pour le transport aérien. Ce problème diffère du PTR au niveau des décisions du meneur et de la réaction du suiveur. La compagnie aérienne (le meneur) doit allouer et tarifer ces sièges afin de maximiser ses revenus. Les usagers (le suiveur) quant à eux désirent acheter leur billet au meilleur prix. Pour modéliser une réaction plus réaliste, l'ensemble des usagers est segmenté en groupe de même préférence relativement au temps de vol et à la qualité de service du produit. Ainsi, chaque groupe d'usagers a une perception du prix du billet en fonction du prix mais aussi de la durée du voyage et de la qualité de service. À partir de la formulation à deux niveaux, les auteurs ont obtenu une formulation linéaire

mixte pour ce problème. Celle-ci est ensuite résolue par un logiciel commercial permettant de fournir des solutions avec des instances de petite taille.

La deuxième application concerne un problème conjoint de tarification et de conception sur un réseau. Dans ce problème, le meneur doit prendre deux types de décisions : d'une part les choix des arcs devant être ouverts sachant que l'ouverture engendre un coût, et d'autre part, le tarif des arcs ouverts. Brotcorne et al. (2003) ont proposé une formulation à deux niveaux mixte pour ce problème et une méthode de résolution. Cette méthode est basée sur la première application d'une relaxation Lagrangienne à la programmation mathématique à deux niveaux. Récemment, Brotcorne et al. (2005c) ont généralisé cette approche au cas où des capacités sont présentes sur les arcs tarifables.

La dernière application est relative à la structure du problème du second niveau où les effets de congestion sont explicitement pris en compte. Le problème du suiveur est cette fois un problème d'équilibre modélisé par une inégalité variationnelle. Fortin (2005) a étudié le problème de tarification dans les réseaux de transport avec congestion et segmentation de la demande. Chaque usager essaie de minimiser ses propres coûts de transports définis comme une combinaison linéaire du temps de transport et du coût du trajet. Les usagers sont repartis selon la valeur monétaire qu'ils accordent au temps en suivant une fonction de densité. Les variables de décision du problème du suiveur sont donc des densités de flux ce qui implique que le problème du suiveur est en dimension infinie. Ce problème est résolu en deux étapes. Tout d'abord, la solution optimale du problème discrétisé est calculée. Elle est ensuite utilisée comme solution de départ d'une méthode de descente. Ces problèmes de la programmation mathématique avec contraintes d'équilibre ont aussi été appliqués par Julsain (1999), Chouman et al. (2005) et Bouhtou et al. (2005) pour des problèmes de tarification en tenant compte de la qualité de service et d'une segmentation des clients dans le contexte des télécommunications.

CHAPITRE 2

PROBLÈME DE TARIFICATION SUR UN RÉSEAU

Dans ce chapitre, nous présentons plus en détails le problème de tarification sur un réseau (PTR). Dans une premier temps, nous posons les notations et nous présentons la formulation à deux niveaux du PTR (section 2.1). Puis, nous donnerons quelques propriétés de ce modèle (sections 2.2 et 2.3). À la section 2.4, nous introduisons le problème d'optimisation inverse. Finalement, une formulation linéaire mixte sera présentée à la section 2.5.

2.1 Formulation du problème de tarification sur un réseau

Nous rappelons la formulation PTRA1 présentée au chapitre 1 mais pour des simplicité d'écriture, nous utiliserons une notation matricielle du PTR. Soit A_1 (resp. A_2) la matrice d'incidence sommets-arcs tarifables (resp. sommets-arcs non tarifables), c (resp. d) le vecteur de coût des arcs tarifables (resp. des arcs non tarifables) et b^k le vecteur de demande du produit k. Soit T le vecteur de tarifs fixés par le meneur et le vecteur x^k (resp. y^k) de flux des arcs tarifables (resp. des arcs non tarifables) pour le produit k. Le problème PTR se réécrit alors de la manière suivante :

$$
\begin{aligned}
\text{PTRA1} : \max_{T,x,y} \quad & \sum_{k\in\mathcal{K}} Tx^k \\
\min_{x,y} \quad & \sum_{k\in\mathcal{K}} (T+c)x^k + dy^k \\
\text{s.c.} \quad & A_1x^k + A_2y^k = b^k, && \forall k \in \mathcal{K}, \\
& x^k, y^k \geq 0, && \forall k \in \mathcal{K}.
\end{aligned}
$$

En nous basant sur la propriété que le flux de chaque produit k est affecté au plus court chemin reliant l'origine $o(k)$ à la destination $d(k)$, nous pouvons remplacer les vecteurs de demande b^k par des vecteurs de demande unitaire e^k où :

$$e_i^k = \begin{cases} 1 & \text{si } i = o(k), \\ -1 & \text{si } i = d(k), \\ 0 & \text{sinon.} \end{cases}$$

Le modèle PTRA1 est alors équivalent à :

$$\begin{aligned} \text{PTRA2} : \max_{T,x,y} \quad & \sum_{k\in\mathcal{K}} n^k T x^k \\ \min_{x,y} \quad & \sum_{k\in\mathcal{K}} (T+c)x^k + dy^k \\ \text{s.c.} \quad & A_1 x^k + A_2 y^k = e^k, \qquad \forall k \in \mathcal{K}, \\ & x^k, y^k \geq 0, \qquad \forall k \in \mathcal{K}. \end{aligned} \tag{2.1}$$

Notons que les demandes n^k de chaque produit k n'apparaissent pas dans la formulation du problème suiveur car il est séparable par produit. Les notations matricielles ont été reportés dans le tableau 2.1.

La formulation du PTR nous porte à faire les remarques suivantes. Tout d'abord, afin d'éliminer les solutions triviales procurant un revenu infini au meneur, nous supposons qu'il existe pour chaque produit, un chemin n'empruntant pas les arcs du meneur. Cette hypothèse permet de garantir une borne supérieure sur le revenu réalisé par le meneur. En effet, si un tel chemin n'existait pas, le meneur posséderait un monopole sur une partie du réseau et pourrait y fixer des tarifs infinis sous une hypothèse de demande captive.

La seconde remarque porte sur le fait que pour un niveau de tarifs fixés, la solution

Tableau 2.1 Notations matricielles

A_1	$\in R^{\lvert\mathcal{N}\rvert\times\lvert\mathcal{A}_1\rvert}$	Matrice d'incidence sommet arc pour les arcs tarifables
A_2	$\in R^{\lvert\mathcal{N}\rvert\times\lvert\mathcal{A}_2\rvert}$	Matrice d'incidence sommet arc pour les arcs non tarifables
b^k	$\in R^{\lvert\mathcal{N}\rvert}$	Vecteur de demande pour le produit $k \in \mathcal{K}$
e^k	$\in R^{\lvert\mathcal{N}\rvert}$	Vecteur de demande unitaire pour le produit $k \in \mathcal{K}$
n^k	$\in R$	Demande du produit $k \in \mathcal{K}$
c	$\in R^{\lvert\mathcal{A}_1\rvert}$	Vecteur de coût unitaire fixe des arcs tarifables
d	$\in R^{\lvert\mathcal{A}_2\rvert}$	Vecteur de coût unitaire fixe des arcs non tarifables
T	$\in R^{\lvert\mathcal{A}_1\rvert}$	Vecteur des tarifs fixés par le meneur pour les arcs tarifables
x^k	$\in R^{\lvert\mathcal{A}_1\rvert}$	Vecteur de flux des arcs tarifables pour le produit $k \in K$
y^k	$\in R^{\lvert\mathcal{A}_2\rvert}$	Vecteur de flux des arcs non tarifables pour le produit $k \in \mathcal{K}$

optimale du problème du suiveur pourrait ne pas être unique. Nous supposons que, si une telle situation se présente, le chemin choisi sera celui qui maximise le revenu du meneur. Cette hypothèse peut être relâchée en diminuant la valeur du tarif associé à chaque arc d'une valeur ϵ suffisamment petite.

Finalement, la dernière remarque porte sur le fait que les tarifs ne sont pas contraints à être positifs. En effet, des tarifs négatifs peuvent être fixés sur certains arcs afin d'engendrer des compensations avec d'autres tarifs lorsque les arcs sont utilisés par plusieurs produits.

Pour illustrer le première et la troisième remarques, nous considérons le réseau décrit à la figure 2.1 comprenant deux produits. Le premier d'origine 1 et de destination 2 possède une demande de 1. Le deuxième d'origine 3 et destination 4 possède également une demande de 1. Le but est de fixer les tarifs des arcs $(5, 6)$ et $(6, 4)$ afin de maximiser le revenu du meneur.

Dans un premier temps, nous remarquons que si l'arc $(1, 2)$ n'existait pas, le premier produit n'aurait d'autre choix que de passer par les arcs $(1, 5)$, $(5, 6)$ et $(6, 2)$. Dans cette situation, le meneur devient libre de fixer le tarif de l'arc $(5, 6)$ à l'infini pour

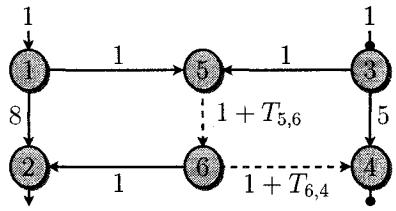


Figure 2.1 Exemple de réseau avec des tarifs optimaux négatifs

maximiser son revenu.

Illustrons maintenant le rôle des tarifs négatifs. En imposant une contrainte de positivité sur les tarifs, la solution optimale de revenu 5 est atteinte pour des tarifs $T_{5,6} = 5$ et $T_{6,4} = 0$. En effet, le plus court chemin pour le produit $(1, 2)$ est le chemin $1 - 5 - 6 - 2$ qui génère un revenu de 5 et le produit $(3, 4)$ emprunte le chemin non taxable $3 - 4$. Avec des tarifs positifs, nous remarquons que le meneur ne fait aucun revenu avec le produit $(3, 4)$. Plus précisément, la demande du produit $(3, 4)$ peut être acheminée soit par le chemin $3-4$ ou soit par le chemin $3-5-6-4$. Avec les tarifs positifs, le coût de $3 - 5 - 6 - 4$ est 3 unités plus cher que celui du chemin $3 - 4$. Cependant, si le meneur accepte de perdre 3 unités de revenu sur ce chemin, il rétablit la compétitivité de celui-ci et conserve 2 unités de revenu pour chaque unité de flux le traversant. C'est exactement ce qui se passe avec des tarifs négatifs où un revenu optimal de 7 est obtenu avec des tarifs $T_{5,6} = 5$ et $T_{6,4} = -3$. Le plus court chemin pour le produit $(1, 2)$ est toujours le chemin $1 - 5 - 6 - 2$ donnant un revenu de 5. Et le produit $(3, 4)$ a maintenant avantage grâce au tarif $T_{6,4} = -3$ à emprunter le chemin $3 - 5 - 6 - 4$ procurant un revenu de 2.

2.2 Fonction objectif du meneur

Comme mentionné par Labbé et al. (1998), la fonction objectif du meneur en fonction des tarifs est linéaire par morceau et non convexe. Cependant, elle est

semi-continue supérieurement, ce qui garantit l'existence d'une solution optimale du PTR.

Cette propriété est illustrée sur le réseau de la figure 2.2 ayant un seul arc tarifable. Ce réseau est soumis à la demande de deux produits. Le premier produit, d'origine 1 et de destination 2, possède une demande de 2 et le deuxième produit, d'origine 3 et de destination 4, possède une demande de 4.

Nous remarquons que si le tarif de l'arc (5,6) est inférieur ou égal à 2, les chemins utilisés sont $1-5-6-2$ et $3-5-6-4$. La totalité du flux emprunte donc l'arc (5,6). Dès que le tarif (5,6) est strictement supérieur à 2, cet arc n'est plus attractif pour le deuxième produit qui emprunte alors le chemin 3-4, mais il le reste pour le premier produit s'il est compris entre 2 et 5. Finalement, dès que le tarif devient strictement supérieur à 5, cet arc n'est plus attractif pour aucun produit. En particulier, le premier produit emprunte alors le chemin 1-2. Le revenu du meneur en fonction du tarif de l'arc (5,6) est représenté à la figure 2.3.

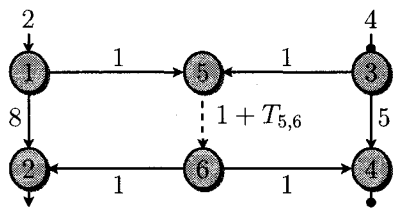


Figure 2.2 Exemple de réseau avec un seul arc tarifable

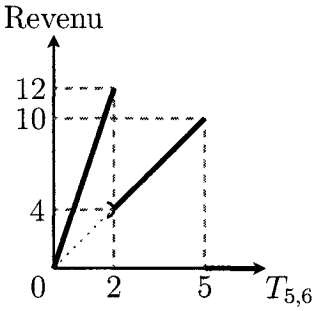


Figure 2.3 Valeur de la fonction objectif du meneur en fonction des tarifs de l'exemple de la figure 2.2

2.3 Borne supérieure sur le revenu du meneur

Il est très facile d'obtenir une borne supérieure sur le revenu du meneur. Considérons d'abord le cas d'un seul produit k. Soit $\gamma^k(\infty)$ le coût du plus court chemin allant de $o(k)$ vers $d(k)$ lorsque les tarifs du meneur sont fixés à l'infini (c'est-à-dire que nous nous interdisons d'utiliser les arcs du meneur). La valeur $\gamma^k(\infty)$ représente une borne supérieure sur le coût des chemins tarifables pour le produit k. Le revenu unitaire possible sur un chemin p reliant l'origine à la destination du produit k est donné par

$$\bar{\gamma}_p^k = \gamma^k(\infty) - \gamma_p^k(0) \tag{2.2}$$

où $\gamma_p^k(0)$ représente le coût du chemin p lorsque les tarifs des arcs du meneur sont fixés à 0.

Notons p_0^k le plus court chemin pour le produit $k \in K$ calculé sur G lorsque les tarifs des arcs de A_1 sont fixés à 0. Une borne supérieure, $\bar{\gamma}^k$, sur le revenu du meneur engendré par le produit k est définie par :

$$\begin{aligned}
\bar{\gamma}^k &= \max_p \bar{\gamma}_p^k, \\
&= \gamma^k(\infty) - \min_p \gamma_p^k(0), \\
&= \gamma^k(\infty) - \gamma_{p_0^k}^k(0), \\
&= \bar{\gamma}_{p_0^k}^k.
\end{aligned}$$

Finalement, en sommant ces bornes sur l'ensemble des produits, nous obtenons une borne supérieure, $\bar{\gamma}$, sur le revenu du meneur :

$$\bar{\gamma} = \sum_{k \in \mathcal{K}} n^k \bar{\gamma}^k. \tag{2.3}$$

Malheureusement cette borne n'est pas toujours atteinte et, dans certains cas, peut être de mauvaise qualité. Considérons le réseau de la figure 2.4 soumis à un produit d'origine 1 et de destination 5 et ayant une demande de 1. Les arcs tarifables sont les arcs $(2,3)$, $(4,5)$ et $(1,5)$

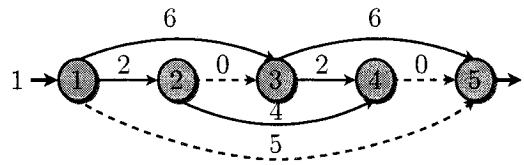


Figure 2.4 Exemple de réseau où la borne supérieure n'est pas atteinte

Le plus court chemin non tarifable est le chemin 1-3-5. Le coût de ce chemin est 12. D'autre part, le plus court chemin, lorsque les tarifs des arcs du meneur sont fixés à 0 est le chemin 1-2-3-4-5 dont le coût est 4. Ainsi, la borne supérieure $\bar{\gamma}$ sur le revenu du meneur pour ce réseau est égale à 12 - 4 = 8. Cette borne est strictement supérieure au revenu optimal du meneur égal à 7 qui est atteint en fixant les tarifs $T_{2,3} = 10$, $T_{4,5} = 14$ et $T_{1,5} = 7$ amenant le suiveur à emprunter le chemin $1 - 5$.

2.4 Problème d'optimisation inverse

Pour le PTR, il existe une relation étroite entre les tarifs fixés par le meneur et le flux emprunté par chaque produit. En effet, pour des tarifs fixés, il est facile de trouver les chemins utilisés par chaque produit en résolvant le problème du suiveur. Par ailleurs, il est également possible, lorsque les fluxs sont fixés, de trouver les tarifs qui engendrent ces fluxs et qui maximisent le revenu du meneur. Ce problème est appelé le *problème d'optimisation inverse* (POI).

Le POI peut être formulé en remplaçant d'abord le problème du suiveur par les conditions d'optimalités primales-duales. Le problème du suiveur étant séparable

par produit, nous désignons par $\lambda^k \in R^{|\mathcal{N}|}$ le vecteur des variables duales associées aux contraintes de conservation de flux (2.1) pour le produit k. PTRA2 est équivalent au problème bilinéaire à un seul niveau suivant :

$$\text{BILIN1} : \max_{T,x,y,\lambda} \quad \sum_{k\in\mathcal{K}} n^k T x^k \tag{2.4}$$

$$\text{s.c.} \qquad A_1 x^k + A_2 y^k = e^k, \qquad \forall k \in \mathcal{K}, \tag{2.5}$$

$$x^k, y^k \geq 0, \qquad \forall k \in \mathcal{K}, \tag{2.6}$$

$$\lambda^k A_1 \leq c + T, \quad \forall k \in \mathcal{K}, \tag{2.7}$$

$$\lambda^k A_2 \leq d, \qquad \forall k \in \mathcal{K}, \tag{2.8}$$

$$x^k(T + c - \lambda^k A_1) = 0, \qquad \forall k \in \mathcal{K}, \tag{2.9}$$

$$y^k(d - \lambda^k A_2) = 0, \qquad \forall k \in \mathcal{K}. \tag{2.10}$$

Les contraintes (2.5) et (2.6) (resp. (2.7) et (2.8)) sont les contraintes d'admissibilité primale (resp. duale) du problème du suiveur. Les contraintes (2.9) et (2.10) sont les contraintes de complémentarité associées à chaque produit.

Puisque les fluxs sont fixés, la fonction objectif est linéaire, les contraintes primales sont satisfaites et les contraintes de complémentarité peuvent être simplifiées. Le POI se réécrit donc :

$$\text{OPTINV} : \max_{T,\lambda} \quad \sum_{k\in\mathcal{K}} \sum_{a\in\mathcal{A}_1 | x_a^k = 1} n^k T_a \tag{2.11}$$

$$\text{s.c.} \quad \lambda_j^k - \lambda_i^k \leq c_{ij} + T_{ij}, \quad \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 0, \tag{2.12}$$

$$\lambda_j^k - \lambda_i^k \leq d_{ij}, \qquad \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 0, \tag{2.13}$$

$$\lambda_j^k - \lambda_i^k = c_{ij} + T_{ij}, \quad \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 1, \tag{2.14}$$

$$\lambda_j^k - \lambda_i^k = d_{ij}, \qquad \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 1. \tag{2.15}$$

Labbé et al. (1998) ont étudié ce problème dans le cas particulier d'un seul pro-

duit. Ils ont montré que le problème dual associé est équivalent à un problème de détermination de plus courts chemins. La même propriété est attendue pour le POI multi-produits puisque le problème (2.11 - 2.15) en λ est séparable par produit cependant les T sont communs. À notre connaissance, aucune étude n'a été effectuée pour le POI multi-produits.

2.5 Formulation linéaire mixte du problème de tarification sur un réseau

La formulation linéaire mixte présentée dans cette section a été proposée par Labbé et al. (1998). Elle s'inspire de BILIN1 où les contraintes (2.9) et (2.10) sont remplacées par les contraintes (2.18) imposant l'égalité des fonctions objectifs primal et dual associées au problème du suiveur. PTRA2 est équivalent à :

$$\text{BILIN2} : \max_{T,x,y,\lambda} \quad \sum_{k\in\mathcal{K}} n^k T x^k \tag{2.16}$$

$$\text{s.c.} \qquad A_1 x^k + A_2 y^k = e^k, \qquad \forall k \in \mathcal{K}, \tag{2.17}$$

$$(T+c)x^k + dy^k = \lambda^k e^k, \quad \forall k \in \mathcal{K}, \tag{2.18}$$

$$\lambda^k A_1 \leq c + T, \quad \forall k \in \mathcal{K}, \tag{2.19}$$

$$\lambda^k A_2 \leq d \qquad \forall k \in \mathcal{K}, \tag{2.20}$$

$$x^k, y^k \geq 0 \qquad \forall k \in \mathcal{K}. \tag{2.21}$$

La formulation BILIN2 est non linéaire d'une part, à cause de la fonction objectif (2.16) et d'autre part, à cause des contraintes (2.18). Pour linéariser ces dernières, nous définissons les variables T_a^k représentant le revenu sur l'arc $a \in \mathcal{A}_1$ pour le produit $k \in \mathcal{K}$:

$$T_a^k = T_a x_a^k, \ \forall a \in \mathcal{A}_1, \forall k \in \mathcal{K},$$

et nous ajoutons les trois contraintes suivantes :

$$-\underline{M}_a^k x_a^k \leq T_a^k \leq \overline{M}_a^k x_a^k, \qquad \forall a \in \mathcal{A}_1, \forall k \in \mathcal{K}, \tag{2.22}$$

$$-\underline{N}_a(1 - x_a^k) \leq T_a - T_a^k \leq \overline{N}_a(1 - x_a^k), \quad \forall a \in \mathcal{A}_1, \forall k \in \mathcal{K}, \tag{2.23}$$

$$x_a^k \in \{0, 1\}, \qquad \forall a \in \mathcal{A}_1, \forall k \in \mathcal{K}, \tag{2.24}$$

où $\underline{M}_a^k, \overline{M}_a^k, \overline{N}_a$ et $\underline{N}_a$ sont des constantes suffisamment grandes. Ces trois contraintes imposent que $T_a = T_a^k$ pour tout arc $a \in \mathcal{A}_1$ et pour tout $k \in \mathcal{K}$. Plus précisément, les contraintes (2.22) permettent de fixer le tarif T_a^k à zéro lorsque $x_a^k = 0$. Autrement dit, le revenu pour le produit k est nul pour cet arc lorsque le flux est nul. Les contraintes (2.23) permettent de fixer le tarif T_a^k à T_a lorsque $x_a^k = 1$. Enfin, les contraintes (2.24) indiquent que les variables x_a^k sont binaires. En définissant, $T^k \in \mathbb{R}^{|A_1|}$ le vecteur de revenu des arcs pour le produit $k \in \mathcal{K}$, nous obtenons finalement la formulation linéaire mixte suivante :

$$\begin{aligned}
\text{MIP} : \max_{T,x,y} \quad & \sum_{k \in \mathcal{K}} n^k T^k \\
\text{s.c.} \quad & A_1 x^k + A_2 y^k = e^k, && \forall k \in \mathcal{K}, \\
& T^k + c x^k + d y^k = \lambda^k e^k, && \forall k \in \mathcal{K}, \\
& \lambda^k A_1 \leq c + T, && \forall k \in \mathcal{K}, \\
& \lambda^k A_2 \leq d, && \forall k \in \mathcal{K}, \\
& -\underline{M}^k x^k \leq T^k \leq \overline{M}^k x^k, && \forall k \in \mathcal{K}, \\
& -\underline{N}(1 - x^k) \leq T - T^k \leq \overline{N}(1 - x^k), && \forall k \in \mathcal{K}, \\
& x^k \in \{0, 1\}, && \forall k \in \mathcal{K}, \\
& y^k \geq 0, && \forall k \in \mathcal{K}.
\end{aligned}$$

Cette formulation comporte $|\mathcal{A}_1| \times |\mathcal{K}|$ variables binaires.

CHAPITRE 3

PROBLÈME D'OPTIMISATION INVERSE

Comme défini dans la section 2.4, le problème d'optimisation inverse (POI) consiste à déterminer les tarifs maximisant le revenu du meneur et reproduisant une réaction du suiveur connue. Il constitue une étape majeure dans les méthodes de résolution du PTR développées dans cette thèse. Ce chapitre est consacré à son étude et à sa résolution. Nous rappelons, à la section 3.1, la méthode de résolution proposée par Labbé et al. (1998) pour le cas mono-produit et nous la généralisons au cas multi-produits à la section 3.2. Finalement, la section 3.3 est consacrée aux extensions de la formulation du POI.

3.1 Problème d'optimisation inverse mono-produit

Dans le cas d'un produit, le POI se formule de la façon suivante :

$$\begin{aligned}
\text{OPTINV1OD} : \max_{T,\lambda} \quad & \sum_{a \in \mathcal{A}_1 | x_a = 1} n T_a \\
\text{s.c.} \quad & \lambda_j - \lambda_i \leq c_{ij} + T_{ij}, \quad \forall (i,j) \in \mathcal{A}_1 | x_{ij} = 0, && (3.1) \\
& \lambda_j - \lambda_i \leq d_{ij}, \quad \forall (i,j) \in \mathcal{A}_2 | y_{ij} = 0, \\
& \lambda_j - \lambda_i = c_{ij} + T_{ij}, \quad \forall (i,j) \in \mathcal{A}_1 | x_{ij} = 1, && (3.2) \\
& \lambda_j - \lambda_i = d_{ij}, \quad \forall (i,j) \in \mathcal{A}_2 | y_{ij} = 1.
\end{aligned}$$

T_{ij} peut être remplacé par $\lambda_j - \lambda_i - c_{ij}$ dans les contraintes (3.1) car les valeurs des variables x_{ij} associées sont à zéro. D'où, en posant $T_{ij} = \lambda_j - \lambda_i - c_{ij}$ pour tout arc $(i,j) \in \mathcal{A}_1$, les contraintes (3.2) et (3.1) sont satisfaites et peuvent être

éliminées. En considérant $A^* = \{(i,j) \in \mathcal{A}_1 | x_{ij} = 1\} \cup \{(i,j) \in \mathcal{A}_2 | y_{ij} = 1\}$ et puisque (x,y) définit un chemin entre l'origine o et la destination d du produit, la fonction objectif se récrit alors :

$$\begin{aligned}
\sum_{a \in \mathcal{A}_1 | x_a = 1} nT_a &= \sum_{(i,j) \in \mathcal{A}_1 | x_{(i,j)} = 1} n(\lambda_j - \lambda_i - c_{ij}), \\
&= n(\sum_{(i,j) \in \mathcal{A}^*} (\lambda_j - \lambda_i) - \sum_{(i,j) \in \mathcal{A}_1 | x_{(i,j)} = 1} c_{ij} - \sum_{(i,j) \in \mathcal{A}_2 | y_{(i,j)} = 1} (\lambda_j - \lambda_i)), \\
&= n(\lambda_d - \lambda_o - \sum_{(i,j) \in \mathcal{A}_1 | x_{(i,j)} = 1} c_{ij} - \sum_{(i,j) \in \mathcal{A}_2 | y_{(i,j)} = 1} d_{ij}),
\end{aligned}$$

où n représente la quantité du produit. OPTINV1OD se réduit à

$$\max_{\lambda} \quad n(\lambda_d - \lambda_o) \tag{3.3}$$

$$\text{s.c.} \quad \lambda_j - \lambda_i \leq d_{ij}, \quad \forall (i,j) \in \mathcal{A}_2 | y_{ij} = 0 \quad (z_{ij}), \tag{3.4}$$

$$\lambda_j - \lambda_i \leq d_{ij}, \quad \forall (i,j) \in \mathcal{A}_2 | y_{ij} = 1 \quad (z_{ij}), \tag{3.5}$$

$$\lambda_i - \lambda_j \leq -d_{ij}, \quad \forall (i,j) \in \mathcal{A}_2 | y_{ij} = 1 \quad (z^-_{ji}). \tag{3.6}$$

En définissant z_{ij} les variables duales associées aux contraintes (3.4) et (3.5) et z^-_{ji} les variables duales associées aux contraintes (3.6), le dual de la formulation (3.3) à (3.6) s'écrit

$$\min_{z, z^-} \quad \sum_{(i,j) \in \mathcal{A}_2} d_{ij} z_{ij} + \sum_{(j,i) \in \mathcal{A}_2 | y_{ji} = 1} (-d_{ij}) z^-_{ji} \tag{3.7}$$

$$\begin{aligned}
\text{s.c.} \quad & A_2 z - A'_2 z^- = b, \\
& z_{ij} \geq 0, \qquad \forall (i,j) \in \mathcal{A}_2, \\
& z^-_{ij} \geq 0, \qquad \forall (j,i) \in \mathcal{A}_2 | y_{ji} = 1.
\end{aligned}$$

Comme le précise la proposition suivante, la résolution du modèle précédent consiste à rechercher un plus court chemin de o à d sur un graphe auxiliaire.

Proposition 1. *Soit $G = (N, A, c)$ un réseau orienté de coût négatif et p^* un chemin élémentaire de o à d. Définissons le graphe $G^{du} = (N, A')$ où $A' = A_2 \cup \{(i,j)|(j,i) \in A_2$ et $(i,j) \in p^*\}$. Le coût des arcs $(i,j) \in A'$ est défini par $d'_{ij} = d_{ij}$ si $(i,j) \in A_2$ et $d'_{ij} = -d_{ij}$ si $(j,i) \in A_2$. Alors pour tout arc $(i,j) \in A_1$, les tarifs optimaux sont donnés par : $T^*_{ij} = \lambda_j - \lambda_i - c'_{ij}$ où pour chaque sommet s, λ_s est le coût du plus court chemin de o à s dans le graphe G^{du}. S'il n'existe pas de plus court chemin de o à d dans G^{du} alors le POI n'admet pas de solution.*

Cette proposition est illustrée sur l'exemple de la figure 3.1(a) composé d'un produit d'origine 1 et de destination 5 de demande 1 et pour lequel la solution du problème du suiveur est définie par le chemin p = 1-2-3-4-5. Le réseau G^{du} résultant est représenté par la figure 3.1(b). Le plus court chemin de 1 à 5 dans G^{du} est 1-2-4-3-5 et les valeurs des variables duales λ sont $\lambda_1 = 0$, $\lambda_2 = 2$, $\lambda_3 = 9$, $\lambda_4 = 11$, $\lambda_5 = 21$. Nous obtenons ainsi les tarifs $T_{23} = 5$ et $T_{45} = 10$.

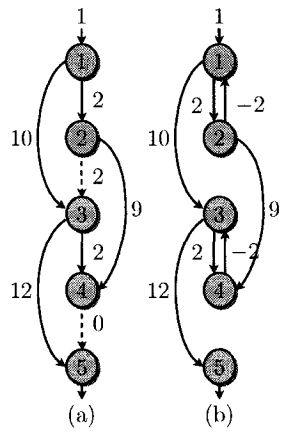


Figure 3.1 Exemple de transformation du réseau pour la résolution du POI mono-produit

3.2 Problème d'optimisation inverse multi-produits

Nous proposons une méthode de résolution du POI multi-produits basée sur une approche similaire (Brotcorne et al. (2004b)). Nous présentons tout d'abord deux

formulations duales du POI multi-produits ayant chacune une structure de multi-flux. Ces structures seront exploitées par une méthode de décomposition pour la résolution.

3.2.1 Formulations duales du problème d'optimisation inverse multi-produits

De façon similaire au cas mono-produit, la formulation du POI multi-produits est obtenue en considérant le modèle bilinéaire sur un niveau du PTR. Comme mentionné dans les sections 2.4 et 2.5, deux formulations bilinéaires équivalentes BILIN1 et BILIN2 sont possibles. Elles engendrent donc deux formulations équivalentes du POI.

3.2.1.1 Première formulation

Considérons la formulation BILIN1 et supposons la solution du problème du suiveur fixée. Nous obtenons alors la formulation du POI suivante (c.f. section 2.4)

$$\text{OPTINV} : \max_{T,\lambda} \quad \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}_1 | x_{ij}^k = 1} n^k T_{ij} \tag{3.8}$$

$$\begin{aligned}
\text{s.c.} \quad & \lambda_j^k - \lambda_i^k \leq c_{ij} + T_{ij} && \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 0, \\
& \lambda_j^k - \lambda_i^k \leq d_{ij} && \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 0, \\
& \lambda_j^k - \lambda_i^k = c_{ij} + T_{ij} && \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 1, \qquad (3.9) \\
& \lambda_j^k - \lambda_i^k = d_{ij} && \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 1.
\end{aligned}$$

Grâce aux contraintes (3.9), les variables T_{ij} sont remplacées par $\lambda_j^k - \lambda_i^k - c_{ij}$ dans la fonction objectif pour tout arc $(i,j) \in \mathcal{A}_1$ et pour tout produit $k \in \mathcal{K}$ tel que

$x_{ij}^k = 1$.

$$\begin{aligned}
\sum_{k\in\mathcal{K}} \sum_{a\in\mathcal{A}_1|x_a^k=1} n^k T_a &= \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}_1|x_{(i,j)}^k=1} n^k(\lambda_j^k - \lambda_i^k - c_{ij}), \\
&= \sum_{k\in\mathcal{K}} n^k\Big(\sum_{(i,j)\in\mathcal{A}_k^*} (\lambda_j - \lambda_i) - \sum_{(i,j)\in\mathcal{A}_1|x_{(i,j)}^k=1} c_{ij} - \sum_{(i,j)\in\mathcal{A}_2|y_{(i,j)}^k=1} (\lambda_j^k - \lambda_i^k)\Big), \\
&= \sum_{k\in\mathcal{K}} n^k(\lambda_{d(k)}^k - \lambda_{o(k)}^k - \sum_{(i,j)\in\mathcal{A}_1|x_{(i,j)}^k=1} c_{ij} - \sum_{(i,j)\in\mathcal{A}_2|y_{(i,j)}^k=1} d_{ij}),
\end{aligned}$$

où $A_k^* = \{(i,j) \in \mathcal{A}_1 | x_{ij}^k = 1\} \cup \{(i,j) \in \mathcal{A}_2 | y_{ij}^k = 1\}$ pour tout $k \in \mathcal{K}$. OPTINV se réduit à :

$$\max_{T,\lambda} \quad \sum_{k\in\mathcal{K}} n^k(\lambda_{d(k)}^k - \lambda_{o(k)}^k) \tag{3.10}$$

$$\text{s.c.} \quad \lambda_j^k - \lambda_i^k \leq c_{ij} + T_{ij} \quad \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 0, \quad (z_{ij}^k) \tag{3.11}$$

$$\lambda_j^k - \lambda_i^k \leq d_{ij} \quad \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 0, \quad (z_{ij}^k) \tag{3.12}$$

$$\lambda_j^k - \lambda_i^k = c_{ij} + T_{ij} \quad \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 1, \quad (z_{ij}^k) \tag{3.13}$$

$$\lambda_j^k - \lambda_i^k = d_{ij} \quad \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 1. \quad (z_{ij}^k) \tag{3.14}$$

En considérant les variables duales z^k, associées aux contraintes (3.10) à (3.14), le duale de la formulation 3.10 se formule comme :

$$\text{DOPTINV} : \min_{z} \quad \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}_1} c_{ij} z_{ij}^k + \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}_2} d_{ij} z_{ij}^k$$

$$\text{s.c.} \quad \sum_{(i,j)\in\mathcal{A}} z_{ij}^k - \sum_{(j,i)\in\mathcal{A}} z_{ji}^k = b_i^k, \quad \forall i \in \mathcal{N}, \forall k \in \mathcal{K}, \tag{3.15}$$

$$\sum_{k\in\mathcal{K}} -z_{ij}^k = 0, \quad \forall (i,j) \in \mathcal{A}_1, \tag{3.16}$$

$$z_{ij}^k \geq 0, \quad \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 0,$$

$$z_{ij}^k \geq 0, \quad \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 0,$$

$$z_{ij}^k \text{ libre}, \quad \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 1,$$

$$z_{ij}^k \text{ libre}, \quad \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 1,$$

Le problème DOPTINV est un problème de multi-flux généralisé. Les contraintes (3.15) sont les contraintes de conservation de flux classiques. Les contraintes (3.16) imposent une utilisation globale des arcs tarifables égale à zéro. Certains arcs peuvent être parcourus dans les deux sens (arcs ayant un flux libre), la résolution du problème DOPTINV ne peut donc se faire par les méthodes classiques proposées pour le problème de multi-flux.

Remarquons que dans le cas mono-produit, les arcs tarifables peuvent être supprimés du réseau puisque les contraintes (3.16) leur imposent un flux nul. De plus, en décomposant le flux libre des arcs (i, j) non tarifables comme la soustraction de deux flux positifs (l'un sur l'arc (i, j) et l'autre sur l'arc inverse (j, i)), nous nous ramenons à la proposition 1.

3.2.1.2 Deuxième formulation

Cette fois, nous considérons le modèle BILIN2 présenté à la section 2.5 dont la formulation est la suivante :

$$
\begin{aligned}
\text{BILIN2} : \max_{T,x,y,\lambda} \quad & \sum_{k\in\mathcal{K}} n^k T x^k \\
\text{s.c.} \quad & A_1 x^k + A_2 y^k = e^k, && \forall k \in \mathcal{K}, \\
& (T+c)x^k + d y^k = \lambda^k e^k, && \forall k \in \mathcal{K}, \\
& \lambda^k A_1 \leq c + T, && \forall k \in \mathcal{K}, \\
& \lambda^k A_2 \leq d, && \forall k \in \mathcal{K}, \\
& x^k, y^k \geq 0, && \forall k \in \mathcal{K}.
\end{aligned}
\tag{3.17}
$$

Les contraintes d'égalité entre les objectifs primal et dual du problème du suiveur (3.17) sont des contraintes bilinéaires. En dualisant ces contraintes par relaxation

lagragienne, une nouvelle formulation bilinéaire sur un niveau est obtenue :

$$
\begin{aligned}
\text{BILIN3} : \max_{T,\lambda,x,y} \quad & \sum_{k\in\mathcal{K}} (n^k T x^k - M n^k((T+c)x^k + dy^k - \lambda^k b^k)) \\
= \max_{T,\lambda,x,y} \quad & \sum_{k\in\mathcal{K}} ((1-M) n^k T x^k + M n^k \lambda^k b^k - M n^k c x^k - M n^k d y^k) \\
\text{s.c.} \quad & A_1 x^k + A_2 y^k = b^k, \qquad \forall k \in \mathcal{K}, \\
& x^k, y^k \geq 0, \qquad \forall k \in \mathcal{K}, \\
& \lambda^k A_1 \leq c + T, \quad \forall k \in \mathcal{K}, \\
& \lambda^k A_2 \leq d, \qquad \forall k \in \mathcal{K},
\end{aligned}
$$

où M est le multiplicateur lagrangien. Labbé et al. (1998) ont prouvé l'existence d'une pénalité exacte garantissant l'équivalence entre les formulations PTR et BILIN3.

Finalement à partir de BILIN3 et en supposant les variables de flux du problème du suiveur fixées, nous obtenons la formulation suivante du POI multi-produits :

OPTINVM :

$$
\begin{aligned}
& \max_{T,\lambda} \quad \sum_{k\in\mathcal{K}} \sum_{a\in\mathcal{A}_1 | x_a^k = 1} (1-M) n^k T_a + M \sum_{k\in\mathcal{K}} \sum_{i\in N} b_i^k \lambda_i^k - M \sum_{k\in\mathcal{K}} \Big(\sum_{a\in\mathcal{A}_1} n^k x_a^k c_a + \sum_{a\in\mathcal{A}_2} y_a^k d_a \Big) \\
= & \max_{T,\lambda} \quad \sum_{k\in\mathcal{K}} \sum_{a\in\mathcal{A}_1 | x_a^k = 1} (1-M) n^k T_a + \sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{N}} M b_i^k \lambda_i^k - cst \\
& \text{s.c.} \quad \lambda_j^k - \lambda_i^k \leq c_{ij} + T_{ij}, \quad \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K}, \\
& \qquad \lambda_j^k - \lambda_i^k \leq d_{ij}, \qquad \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K},
\end{aligned}
$$

dont le dual s'écrit

$$\text{DOPTINVM} : \min_{z} \quad \sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}_1} c_{ij}z_{ij}^k + \sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}_2} d_{ij}z_{ij}^k \tag{3.18}$$

$$\text{s.c.} \quad \sum_{(i,j)\in\mathcal{A}} z_{ij}^k - \sum_{(j,i)\in\mathcal{A}} z_{ji}^k = Mb_i^k \qquad \forall i \in \mathcal{N}, \forall k \in \mathcal{K}, \tag{3.19}$$

$$\sum_{k\in\mathcal{K}} -z_{ij}^k = \sum_{k\in\mathcal{K}} -(M-1)n^k x_a^k \qquad \forall (i,j) \in \mathcal{A}_1, \tag{3.20}$$

$$z_{ij}^k \geq 0, \quad \forall (i,j) \in \mathcal{A}_1 \qquad \forall k \in \mathcal{K}, \tag{3.21}$$

$$z_{ij}^k \geq 0, \quad \forall (i,j) \in \mathcal{A}_2 \qquad \forall k \in \mathcal{K}, \tag{3.22}$$

où les z_{ij}^k sont les variables duales associées aux contraintes de OPTINVM.

DOPTINVM est le modèle d'un problème de multi-flux où la quantité de flux sur les arcs tarifables est imposée par les contraintes (3.20). Notons que la résolution de DOPTINVM ne permet pas de détecter les solutions non réalisables à cause de la pénalisation des contraintes de complémentarité contrairement à la résolution de DOPTINV.

3.2.2 Résolution du problème d'optimisation inverse multi-produits

Dans cette section, nous présentons une méthode de résolution du POI multi-produits en nous basant sur la formulation DOPTINV. Un raisonnement semblable peut s'appliquer avec DOPTINVM. Rappelons que DOPTINV et DOPTINVM sont des problèmes de multi-flux. Plusieurs méthodes ont été proposées dans la littérature pour résoudre ce problème. Une bibliographie complète est proposée dans la thèse de Wang (2003) et des tests comparatifs entre les différentes méthodes sont effectués. La méthode de décomposition de Dantzig-Wolfe s'avère être la plus efficace pour résoudre ce type de problème, nous l'appliquerons donc à DOPTINV.

La méthode de décomposition de Dantzig-Wolfe permet de résoudre des programmes linéaires de grande taille dont les contraintes se séparent en deux groupes : les contraintes faciles et les contraintes difficiles (ou couplantes). Plus précisément, les contraintes faciles possèdent une structure particulière telle que si l'autre groupe est omis, on obtient un programme linéaire très facile à résoudre. Cette méthode est basée sur une reformulation du programme linéaire sous forme d'un autre programme linéaire équivalent, appelé *problème maître*. Ce problème maître contient seulement les contraintes difficiles, une ou plusieurs contraintes de convexité et un très grand nombre de variables. Les contraintes faciles sont utilisées pour définir les nouvelles variables.

Pour l'appliquer à DOPTINV, les contraintes (3.16) sont les contraintes difficiles. Pour chaque produit $k \in \mathcal{K}$, le domaine réalisable est défini par :

$$\begin{aligned}\Delta^k = \{(z_{ij}^k) \mid & \sum_{(i,j)\in\mathcal{A}} z_{ij}^k - \sum_{(j,i)\in\mathcal{A}} z_{ji}^k = b_i^k, \quad \forall i \in \mathcal{N} \\ & z_{ij}^k \geq 0, \quad \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 0, \\ & z_{ij}^k \geq 0, \quad \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 0, \\ & z_{ij}^k \text{ libre}, \quad \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 1, \\ & z_{ij}^k \text{ libre}, \quad \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 1\}\end{aligned}$$

Et par le théorème de Minkowski, chaque solution $(z^k) \in \Delta^k$ peut être exprimée comme une combinaison convexe des points extrêmes et une combinaison linéaire des rayons extrêmes. Notons que les points extrêmes de Δ^k sont des chemins élémentaires reliant l'origine à la destination du produit k et les rayons extrêmes des

cycles élémentaires présents dans le graphe. Ainsi, nous avons

$$(z^k) = \sum_{p \in P^k} Z_p \omega_p + \sum_{p \in \tilde{P}} \tilde{Z}_p \rho_p \text{ tel que } \begin{cases} \sum_{p \in P^k} Z_p = 1, & \\ Z_p \geq 0, & \forall p \in P^k \setminus P^{k^*} \\ Z_p \text{ libre}, & \forall p \in P^{k^*} \\ \tilde{Z}_p \geq 0, & \forall p \in \tilde{P} \end{cases}$$

où P^k est l'ensemble des indices des chemins élémentaires de Δ^k reliant l'origine à la destination du produit k, P^{k^*} est l'ensemble des indices des chemins élémentaires de Δ^k ayant un flux libre, ω_p est le vecteur de Δ^k associé au chemin $p^{ème}$ chemin, Z_p est le flux associé à ce chemin ; $\tilde{P}$ est l'ensemble des indices des cycles de Δ^k, ρ_p est le vecteur de Δ^k associé au $p^{ème}$ cycle, $\tilde{Z}_p$ est le flux associé à ce cycle.

En remplaçant les variables z_{ij}^k par leur expression dans la fonction objectif et les contraintes, DOPTINV est équivalent à

$$\begin{aligned} \text{DOPTINVDW} : \min_{Z} \quad & \sum_{k \in \mathcal{K}} \sum_{p \in P^k} C_p Z_p + \sum_{p \in \tilde{P}} C_p \tilde{Z}_p \\ \text{s.c.} \quad & \sum_{p \in P^k} Z_p = n^k, \forall k \in \mathcal{K}, && (3.23) \\ & -\sum_{k \in \mathcal{K}} \sum_{p \in P^k} I_{a,p} Z_p - \sum_{p \in \tilde{P}} I_{a,p} \tilde{Z}_p = 0, \forall a \in \mathcal{A}_1, && (3.24) \\ & Z_p \geq 0, \quad \forall k \in \mathcal{K}, \forall p \in P^k \setminus P^{k^*}, \\ & Z_p \text{ libre}, \quad \forall k \in \mathcal{K}, \forall p \in P^{k^*}, \\ & \tilde{Z}_p \geq 0, \quad \forall k \in \mathcal{K}, \forall p \in \tilde{P}, \end{aligned}$$

où C_p définit le coût d'un chemin ou d'un cycle et $I_{a,p}$ est une constante égale à 1 si l'arc a appartient au chemin ou cycle p et 0 sinon.

Cette formulation comporte peu de contraintes mais un nombre exponentiel de variables. Une énumération complète des chemins et des cycles élémentaires n'étant

pas possible, une méthode de génération de colonnes doit être utilisée pour résoudre la formulation DOPTINVDW. Plus précisément, à chaque itération, DOPTINVDW est résolu avec un nombre restreint de variables (problème maître restreint (PMR)). Ensuite, l'optimalité de la solution obtenue est testée par rapport au problème maître non restreint en résolvant un sous-problème. Si l'optimalité est atteinte, la résolution est terminée. Dans le cas contraire, la solution du sous problème est utilisée pour ajouter de nouvelles variables au PMR et effectuer une nouvelle résolution.

Nous dénotons par (λ^k) et (T_a) les variables duales associées aux contraintes (3.23) et (3.24). Le coût réduit associé à une variable Z_p, $p \in P^k$ est égale à $C_p + \sum_{a \in \mathcal{A}_1} I_{a,p} T_a - \lambda^k$ et celui associé à une variable $\tilde{Z}_p$, $p \in \tilde{P}$ est égale à $C_p + \sum_{a \in \mathcal{A}_1} I_{a,p} T_a$. La solution optimale du problème maître restreint est la solution optimale du problème maître non restreint si les conditions suivantes sont vérifiées :

$$\begin{cases} \min_{p \in P^k} (C_p + \sum_{a \in \mathcal{A}_1} I_{a,p} T_a) \omega_p - \lambda^k \geq 0, \quad \forall k \in \mathcal{K}, \\ \min_{p \in \tilde{P}} (C_p + \sum_{a \in \mathcal{A}_1} I_{a,p} T_a) \rho_p \geq 0. \end{cases}$$

Par conséquent, pour chaque produit $k \in K$, une variable avec un coût réduit positif peut être découverte en résolvant le sous problème suivant

$$\begin{aligned} \text{DOPTINV\_S}(\lambda^k, T) : \min_{Z} \quad & \sum_{k \in \mathcal{K}} \Big(\sum_{p \in P^k} (C_p + \sum_{a \in \mathcal{A}_1} I_{a,p} T_a) Z_p \Big) - \lambda^k \\ \text{s.c.} \quad & \sum_{p \in P^k} Z_p = n^k, \quad \forall k \in \mathcal{K}, \\ & Z_p \geq 0, \quad \forall k \in \mathcal{K}, \forall p \in P^k, \\ & Z_p \text{ libre}, \quad \forall k \in \mathcal{K}, \forall p \in P^{k*}, \\ & \tilde{Z}_p \geq 0, \quad \forall p \in \tilde{P}. \end{aligned}$$

Après la résolution de DOPTINV_S(λ^k, T), trois cas sont possibles :

Cas 1 : DOPTINV_S(λ^k, T) est non borné.

Il existe un cycle élémentaire avec un coût réduit négatif. Ce cycle permet de générer une nouvelle variable $\tilde{Z}_p$ pour le problème maître restreint.

Cas 2 : DOPTINV_S(λ^k, T) a une solution optimale de coût $z^k(\lambda^k, T_a) < 0$.

Il existe donc un chemin avec un coût réduit négatif. Ce chemin permet de générer une nouvelle variable Z_p pour le problème maître restreint.

Cas 3 : DOPTINV_S(λ^k, T) a une solution optimale de coût $z^k(\lambda^k, T_a) \geq 0$.

Aucune nouvelle variable ne peut être générée.

Si le troisième cas se présente pour l'ensemble des sous problèmes, alors la solution optimale de DOPTINVPDW est trouvée.

Finalement, l'algorithme de génération de colonnes pour la résolution du POI multi-produit est le suivant :

Génération de colonnes pour le problème d'optimisation inverse multi-produits

Pas 0 *Initialisation*

- Soit le compteur d'itérations $i = 0$.
- Initialiser les ensembles de variables P_i^k et $\tilde{P}_i$ afin que le problème maître restreint soit réalisable.

Pas 1 *Résolution du problème maître*

- Résoudre le problème maître restreint en considérant les variables P_i^k et $\tilde{P}_i$.
- **S'**il n'admet pas de solution,

  alors arrêt car le POI n'admet pas de solution,

  sinon obtenir une solution primale $(Z_p, \tilde{Z}_p)$ et une solution duale (λ^k, T_a).

Pas 2 *Résolution des sous-problèmes*

Pour chaque produit $k \in \mathcal{K}$,

- Résoudre DOPTINV_S(λ^k, T_a).
- **S'**il est non borné,

  alors ajouter le cycle de coût négatif à $\tilde{P}$,

  sinon si la valeur de la fonction objectif $v^k < 0$ **alors** ajouter le chemin de coût négatif à P^k.

PAS 3 *Test d'optimalité*

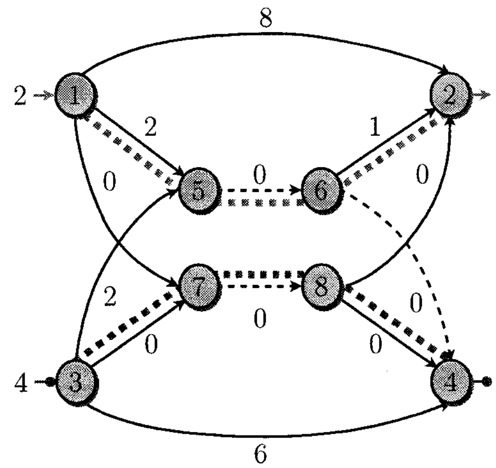
- **Si** $\min_{k \in \mathcal{K}} \{v^k\} = 0$ **alors** la solution optimale est trouvée,

  sinon aller au **Pas 1**.

Au Pas 0, chaque ensemble P_0^k est initialisé avec le plus court chemin non tarifable reliant l'origine à la destination du produit k et le chemin emprunté par le suiveur pour le produit k. L'ensemble des cycles $\tilde{P}$ est initialement vide. Ceci permet, d'une part, de garantir la réalisabilité du problème maître restreint et, d'autre part, elle simplifie la résolution des sous problèmes. En effet, le seul chemin d'un produit $k \in \mathcal{K}$ avec un flux libre est celui emprunté par le suiveur. Donc, s'il est ajouté à l'ensemble P_0^k, il ne sera plus nécessaire de le générer. Et comme tous les autres chemins du produit k ont un flux non négatif, le sous-problème est un problème de plus court chemin sur le réseau initial, évitant ainsi l'ajout d'arcs inverses.

Lors de résolutions consécutives, la gestion de la taille des ensembles des colonnes pour accroître les performances de la méthode de génération de colonnes se fait de façon à atteindre un compromis entre la génération de colonnes et le temps de résolution du problème maître restreint. Lorsque cette taille dépasse $30 \times |\mathcal{K}|$, nous supprimons toutes les colonnes ayant un coût réduit strictement positif.

L'application de la méthode de génération de colonnes est illustrée sur le réseau de la figure 3.2. Sur ce réseau, le meneur doit déterminer les tarifs des arcs $(5, 6)$, $(7, 8)$ et $(6, 4)$. Le suiveur doit acheminer deux produits : d'origine 1 - de destination 2 et



P	Chemins	C_p	$\in P^*$
1	1-2	8	non
2	1-5-6-2	3	oui
3	1-7-8-2	0	non
4	3-4	6	non
5	3-5-6-4	2	non
6	3-7-8-4	0	oui

Figure 3.2 Exemple utilisé pour la résolution du POI multi-produits

d'origine 3 - de destination 4 de demande respective 2 et 4. L'ensemble des chemins reliant l'origine à la destination de chaque produit est reporté sur le tableau de la figure 3.2. Nous supposons que le chemins $1-5-6-2$ est fixé pour le premier produit et $3-7-8-4$ pour le deuxième. Pour chaque chemin, nous indiquons son indice (P), les noeuds le constituant (Chemins), son coût à tarif nul (C_p) et s'il est solution du suiveur ($\in P^*$).

Nous commençons par initialiser les ensembles de chemins P^1 et P^2 de manière suivante : $P^1 = \{1,2\}, P^2 = \{4,6\}$. Le problème maître restreint et la solution optimale associée s'écrivent donc :

$$\begin{array}{rcccccccccl} v = \min & 8Z_1 & + & 3Z_2 & + & 6Z_4 & + & 0z_6 & & & \\ s.c. & Z_1 & + & Z_2 & & & & & = & 2 & (\pi^1) \\ & & & & & Z_4 & + & Z_6 & = & 4 & (\pi^2) \\ & & - & Z_2 & & & & & = & 0 & (T_{56}) \\ & & & & & & & & = & 0 & (T_{64}) \\ & & & & & & - & Z_6 & = & 0 & (T_{78}) \\ & Z_1 & & & & & & & \geq & 0 & \\ & & & & & Z_4 & & & \geq & 0 & \end{array} \quad \text{et} \quad \left\{ \begin{array}{r} v = 34 \\ Z_1 = 2 \\ Z_4 = 4 \\ \pi^1 = 8 \\ \pi^2 = 6 \\ T_{56} = 5 \\ T_{64} = 20 \\ T_{78} = 6 \end{array} \right. ,$$

À partir de la solution du problème maître restreint, nous résolvons les sous-problèmes DOPTINV_S ($\pi^1 = 8, T_{56} = 5, T_{78} = 6$) et DOPTINV_S ($\pi^2 = 6, T_{56} = 5, T_{78} = 6$) associés à chaque produit. C'est-à-dire que nous cherchons le plus court chemin reliant l'origine à la destination du produit 1 (resp. produit 2) lorsque les tarifs sont $T_{56} = 5$ et $T_{78} = 6$. Le coût du plus court chemin est comparé à celui proposé par la solution du problème maître $\pi^1 = 8$ (resp. $\pi^2 = 6$). Les solutions de DOPTINV_S ($\pi^1 = 8, T_{56} = 5, T_{78} = 6$) et DOPTINV_S ($\pi^2 = 6, T_{56} = 5, T_{78} = 6$) sont reportées dans le tableau suivant :

Produits	p	Chemins	Coûts Réduits
1	3	1-7-8-2	-3
2	6	3-7-8-4	0

La solution du sous-problème du produit 1, nous indique un chemin possédant un coût réduit négatif : c'est-à-dire que le coût du chemin 3 pour des tarifs fixés à $T_{56} = 5$ et $T_{78} = 6$ est inférieur au coût espéré $\pi^1 = 8$. Ce chemin est donc ajouté à l'ensemble P^1 qui devient $P^1 = \{1, 2, 3\}$. Le processus est répété en considérant ce nouvel ensemble de chemins. Le problème maître restreint et la solution associée sont alors donnés par :

$$
\begin{array}{rrcrcrcrcrcll}
v = \min & 8Z_1 & + & 3Z_2 & + & 6Z_4 & + & 0Z_6 & + & 0Z_3 & & & \\
 & Z_1 & + & Z_2 & & & & & + & Z_3 & = & 2 & (\pi^1) \\
 & & & & & Z_4 & + & Z_6 & & & = & 4 & (\pi^2) \\
 & & - & Z_2 & & & & & & & = & 0 & (T_{56}) \\
 & & & & & & & & & & = & 0 & (T_{64}) \\
 & & & & & & - & Z_6 & - & Z_3 & = & 0 & (T_{78}) \\
 & Z_1 & & & & & & & & & \geq & 0 & \\
 & & & & & Z_4 & & & & & \geq & 0 &
\end{array}
\quad \text{et} \quad
\left\{
\begin{array}{l}
v = 30 \\
Z_3 = 2 \\
Z_4 = 4 \\
Z_6 = -2 \\
\pi^1 = 6 \\
\pi^2 = 6 \\
T_{56} = 3 \\
T_{64} = 20 \\
T_{78} = 6
\end{array}
\right. ,
$$

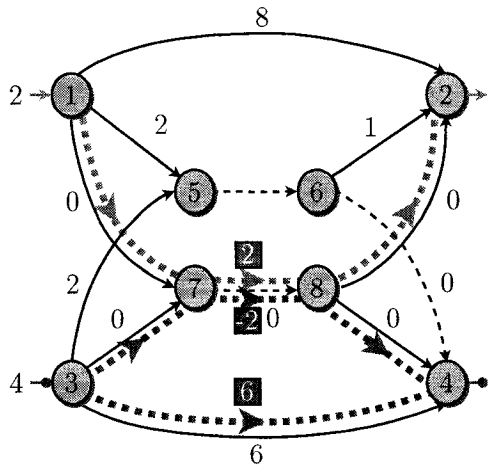


Figure 3.3 Solution optimale du problème maître de l'exemple de la figure 3.2

Les solutions des sous-problèmes associés aux produits 1 et 2 sont reportées dans le tableau suivant :

Produits	p	Chemins	Coûts Réduits
1	2	1-7-8-2	0
2	6	3-7-8-4	0

Tous les chemins ayant un coût réduit nul, la solution optimale du POI est obtenue (c.f. figure 3.3).

Avant d'analyser cette solution, rappelons que le POI permet de déterminer les tarifs des arcs tarifables utilisés par la solution du problème du suiveur afin de maximiser le revenu du meneur. Dans notre exemple, le tarif T_{56} a pour borne supérieure 5 étant donné le coût de 8 du chemin non tarifable $1-2$ du premier produit. De même, le tarif T_{78} est borné par 6 à cause du chemin non tarifable du deuxième produit $3-4$. Par ailleurs, la valeur de la borne sur le tarif T_{78} a un impact sur celle de l'arc $(5,6)$. En effet, puisque le chemin utilisé par le suiveur est $1-5-6-2$, son coût doit être inférieur à celui du chemin $1-7-8-2$ soit 6. Donc, le tarif T_{56} est borné par 3. La maximisation du revenu du meneur et les

valeurs des bornes supérieures sur les tarifs nous donne les tarifs $T_{56} = 3, T_{78} = 6$ et $T_{64} = 20$.

Ce raisonnement peut être obtenu à partir de la solution optimale du problème maître (représentée à la figure 3.3). Pour le produit $(3, 4)$, les 4 unités de demande sont acheminées par le chemin non tarifable 3–4. Donc, le coût du chemin 3–7–8–4 emprunté par le suiveur est borné par celui du chemin $3 - 4$ ce qui implique que le tarif T_{78} est borné par 6. Pour le produit $(1, 2)$, les 2 unités de demande sont acheminées par le chemin $1 - 7 - 8 - 2$. Donc, le coût du chemin $1 - 5 - 6 - 2$ est borné par celui du chemin $1 - 7 - 8 - 2$. Cependant, le coût de ce chemin dépendant du tarif T_{78} borné à cause du produit $3 - 4$. Dans ce cas, le chemin emprunté par le suiveur $3 - 7 - 8 - 4$ est utilisé à contre sens avec le produit $(3, 4)$ pour tenir compte de la valeur de la borne supérieure sur le tarif T_{78}. En réunissant tous ces chemins, nous remarquons que le coût du chemin $1 - 5 - 6 - 2$ est borné par celui du chemin $1 - 7 - 3 - 4 - 8 - 2$ soit 3.

3.3 Extensions du problème d'optimisation inverse

Dans cette section, nous présentons deux extensions du POI. La première est essentielle pour accroître les performances des heuristiques développées dans cette thèse. Elle permet la détermination de tarifs des arcs non utilisés par la solution du problème du suiveur. La deuxième permet d'intégrer des contraintes de borne inférieure sur les tarifs (comme par exemple des tarifs positifs). De plus, nous expliquons comment les résoudre en adaptant la méthode de génération de colonnes proposée précédemment.

3.3.1 Problème d'optimisation avec détermination du tarif des arcs tarifables non utilisés

Le POI, considéré à la section précédente, permet de déterminer les tarifs des arcs tarifables utilisés par la solution du problème du suiveur afin de maximiser le revenu du meneur et d'ajuster ceux des arcs tarifables non utilisés, de façon à ne pas les rendre attractifs pour le suiveur.

Considérons à nouveau l'exemple de la figure 3.2 comprenant trois arcs tarifables $(5,6)$, $(6,4)$ et $(7,8)$. Seul l'arc tarifable $(6,4)$ n'est pas utilisé par le suiveur, ce qui implique que la valeur optimale de T_{64} n'est pas unique. Nous recherchons donc le plus petit tarif pouvant être appliqué tel que toute diminution de celui-ci implique une nouvelle réaction pour le suiveur. Ici, le plus petit tarif T_{64} pouvant être appliqué est 3. En effet, un tarif T_{64} inférieur à 3 implique l'utilisation du chemin $3-5-6-4$ pour le produit $(3,4)$.

Les tarifs ajustés peuvent être obtenus en modifiant la fonction objectif de la formulation OPTINV de façon à obtenir, pour les arcs non utilisés, des tarifs les plus petits possibles. Nous obtenons ainsi la formulation OPTINV$^+$ du POI avec ajustement des tarifs :

$$
\begin{aligned}
\text{OPTINV}^+ : \max_{T,\lambda} \quad & \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}_1 | x_{ij}^k = 1} n^k T_{ij} - \sum_{(i,j)\in\mathcal{A}_1 | \forall k\in\mathcal{K}, x_{ij}^k = 0} \epsilon T_{ij} \\
\text{s.c.} \quad & \lambda_j^k - \lambda_i^k \leq c_{ij} + T_{ij} \quad \forall (i,j)\in\mathcal{A}_1, \forall k\in\mathcal{K} | x_{ij}^k = 0, \\
& \lambda_j^k - \lambda_i^k \leq d_{ij} \quad \forall (i,j)\in\mathcal{A}_2, \forall k\in\mathcal{K} | y_{ij}^k = 0, \\
& \lambda_j^k - \lambda_i^k = c_{ij} + T_{ij} \quad \forall (i,j)\in\mathcal{A}_1, \forall k\in\mathcal{K} | x_{ij}^k = 1, \\
& \lambda_j^k - \lambda_i^k = d_{ij} \quad \forall (i,j)\in\mathcal{A}_2, \forall k\in\mathcal{K} | y_{ij}^k = 1,
\end{aligned}
$$

Par un raisonnement similaire à celui utilisé à la section précédente, la formulation duale DOPTINV$^+$ est donnée par :

$$
\begin{aligned}
\text{DOPTINV}^+ : \min_{z} \quad & \sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}_1} c_{ij}z_{ij}^k + \sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}_2} d_{ij}z_{ij}^k \\
\text{s.c.} \quad & \sum_{(i,j)\in\mathcal{A}} z_{ij}^k - \sum_{(j,i)\in\mathcal{A}} z_{ji}^k = b_i^k, \quad \forall i \in \mathcal{N}, \forall k \in \mathcal{K}, \\
& \sum_{k\in\mathcal{K}} -z_{ij}^k = \begin{cases} \epsilon & \text{si } x_{ij}^k = 0 \\ 0 & \text{sinon} \end{cases}, \forall (i,j) \in \mathcal{A}_1, \qquad (3.25) \\
& z_{ij}^k \geq 0, \quad \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 0, \\
& z_{ij}^k \geq 0, \quad \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 0, \\
& z_{ij}^k \text{ libre}, \quad \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 1, \\
& z_{ij}^k \text{ libre}, \quad \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 1,
\end{aligned}
$$

où ϵ est une constante suffisamment petite. Dans cette dernière formulation, les contraintes (3.16) de la formulation DOPTINV ont été remplacées par les contraintes (3.25).

La méthode de génération de colonnes peut être appliquée en remplaçant les contraintes (3.24) dans la formulation du problème maître par

$$
-\sum_{k\in\mathcal{K}}\sum_{p\in P^k} I_{a,p}Z_p - \sum_{p\in\tilde{P}} I_{a,p}\tilde{Z}_p = \begin{cases} \epsilon & \text{si } x_a^k = 0 \\ 0 & \text{sinon} \end{cases}, \forall a \in \mathcal{A}_1.
$$

3.3.2 Problème d'optimisation inverse avec bornes inférieures sur les tarifs

Nous nous intéressons au POI lorsque chaque tarif T_{ij} du meneur doit être supérieur à une borne inférieure T_{ij}^{min}. La formulation $\text{OPTINV}_{T_{min}}$ est donnée par :

$$
\begin{aligned}
\text{OPTINV}_{T_{min}} : \max_{T,\lambda} \quad & \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}_1 | x_{ij}^k = 1} n^k T_{ij} \\
\text{s.c.} \quad & \lambda_j^k - \lambda_i^k \leq c_{ij} + T_{ij} && \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 0, \\
& \lambda_j^k - \lambda_i^k \leq d_{ij} && \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 0, \\
& \lambda_j^k - \lambda_i^k = c_{ij} + T_{ij} && \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 1, \\
& \lambda_j^k - \lambda_i^k = d_{ij} && \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 1, \\
& T_{ij} \geq T_{ij}^{min} && \forall (i,j) \in \mathcal{A}_1.
\end{aligned}
$$

En posant $T_{ij} = T'_{ij} + T_{ij}^{min}, \forall (i,j) \in \mathcal{A}_1$, $\text{OPTINV}_{T_{min}}$ se récrit :

$$
\begin{aligned}
\text{OPTINV'}_{T_{min}} : \max_{T',\lambda} \quad & \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}_1 | x_{ij}^k = 1} n^k T'_{ij} + \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}_1 | x_{ij}^k = 1} n^k T_{ij}^{min} \\
\text{s.c.} \quad & \lambda_j^k - \lambda_i^k \leq c_{ij} + T'_{ij} + T_{ij}^{min} && \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 0, \\
& \lambda_j^k - \lambda_i^k \leq d_{ij} && \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 0, \\
& \lambda_j^k - \lambda_i^k = c_{ij} + T'_{ij} + T_{ij}^{min} && \forall (i,j) \in \mathcal{A}_1, \forall k \in \mathcal{K} | x_{ij}^k = 1, \\
& \lambda_j^k - \lambda_i^k = d_{ij} && \forall (i,j) \in \mathcal{A}_2, \forall k \in \mathcal{K} | y_{ij}^k = 1, \\
& T'_{ij} \geq 0 && \forall (i,j) \in \mathcal{A}_1.
\end{aligned}
$$

Comme la quantité $\sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}_1|x_{ij}^k=1} n^k T_{ij}^{min}$ est constante et par un raisonnement similaire à la section précédente, le dual DOPTINV'$_{T_{min}}$ s'écrit :

$$
\begin{aligned}
\text{DOPTINV'}_{T_{min}} : \min_{z} \quad & \sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}_1}(c_{ij}+T_{ij}^{min})z_{ij}^k + \sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{A}_2} d_{ij}z_{ij}^k \\
\text{s.c.} \quad & \sum_{(i,j)\in\mathcal{A}} z_{ij}^k - \sum_{(j,i)\in\mathcal{A}} z_{ji}^k = b_i^k, \quad \forall i\in\mathcal{N}, \forall k\in\mathcal{K}, \\
& \sum_{k\in\mathcal{K}} -z_{ij}^k \geq 0, \quad \forall (i,j)\in\mathcal{A}_1, \qquad (3.26) \\
& z_{ij}^k \geq 0, \quad \forall (i,j)\in\mathcal{A}_1, \forall k\in\mathcal{K}|x_{ij}^k = 0, \\
& z_{ij}^k \geq 0, \quad \forall (i,j)\in\mathcal{A}_2, \forall k\in\mathcal{K}|y_{ij}^k = 0, \\
& z_{ij}^k \text{ libre}, \quad \forall (i,j)\in\mathcal{A}_1, \forall k\in\mathcal{K}|x_{ij}^k = 1, \\
& z_{ij}^k \text{ libre}, \quad \forall (i,j)\in\mathcal{A}_2, \forall k\in\mathcal{K}|y_{ij}^k = 1.
\end{aligned}
$$

Dans cette dernière formulation, les contraintes (3.16) de la formulation DOPTINV ont été remplacées par les contraintes (3.26). Comme précédemment, la méthode de génération de colonnes peut être appliquée simplement en remplaçant les contraintes (3.24) dans la formulation du problème maître par

$$-\sum_{k\in\mathcal{K}}\sum_{p\in P^k} I_{a,p}Z_p - \sum_{p\in\tilde{P}} I_{a,p}\tilde{Z}_p \geq 0, \forall a\in\mathcal{A}_1$$

et en considérant le coût fixe suivant pour les arcs tarifables

$$c'_{ij} = c_{ij} + T_{ij}^{min}, \forall (i,j)\in\mathcal{A}_1.$$

3.4 Conclusion

Ce chapitre fut consacré à l'étude et la résolution du POI. Pour une solution du suiveur fixée, ce problème permet de déterminer un plan tarifaire qui engendrent les flux du suiveur et maximisent le revenu du meneur.

Deux formulations du POI ont été proposées. Le dual de ces formulations est résolu par une méthode de génération de colonnes où les sous-problèmes sont des problèmes de plus court chemin. Quelques extensions du POI ont également été présentées : soit pour tenir compte de la présence de bornes inférieures sur les tarifs, soit pour trouver des tarifs ajustés sur les arcs tarifables non utilisés par la solution du suiveur.

Des expérimentations numériques présentées à la section 5.5.1.1 mettent en évidence l'efficacité de la méthode de génération colonnes pour résoudre le POI.

CHAPITRE 4

MÉTHODE EXACTE BASÉE SUR LE CONCEPT DE K-CHEMINS POUR LE PROBLÈME DE TARIFICATION SUR UN RÉSEAU

Dans ce chapitre, nous présentons une méthode exacte basée sur la méthode proposée par Didi et al. (1999). Plus précisément, nous proposons différentes améliorations visant à rendre l'énumération des solutions du problème du suiveur (appelés K-chemins) plus efficace (Brotcorne et al. (2005a,b)).

Ce chapitre se décompose de la manière suivante. À la section 4.1, nous présentons la méthode proposée par Didi et al. (1999). Puis, une amélioration de l'algorithme permettant de réduire le nombre de K-chemins non réalisables générés est présentée à la section 4.2. Une nouvelle structure de données, pour l'exploration des K-chemins, est introduite à la section 4.3. Elle sera exploitée pour le calcul de nouvelles bornes supérieures sur le revenu engendré à la section 4.4. Une heuristique permettant de déterminer un revenu de meilleure qualité est décrite à la section 4.5. Finalement, l'algorithme de la méthode exacte est résumé à la section 4.6 et les résultats numériques sont présentés à la section 4.7.

4.1 Principe de la méthode

La méthode de résolution exacte proposée par Didi et al. (1999) pour le PTR exploite la structure du problème du suiveur et le problème d'optimisation inverse (cf. section 2.4). Définissons un *K-chemin* comme un vecteur de $|\mathcal{K}|$ chemins utilisés

par chaque produit et $\mathcal{C}$ l'ensemble des K-chemins :

$$\mathcal{C} = \{(\mathcal{P}_1, ..., \mathcal{P}_{|\mathcal{K}|}) | \mathcal{P}_k \in P^k, k = 1...|\mathcal{K}|\},$$

où P^k est l'ensemble des chemins reliant l'origine à la destination du produit k. Nous représentons le coût d'un K-chemin $E = (\mathcal{P}_1, ..., \mathcal{P}_{|K|}) \in \mathcal{C}$ par $c(E)$. Ce coût est égal à la somme des coûts fixes de chaque chemin

$$c(E) = \sum_{k \in \mathcal{K}} n^k (\sum_{a \in \mathcal{A}_1 \cap \mathcal{P}_k} c_a + \sum_{a \in \mathcal{A}_2 \cap \mathcal{P}_k} d_a).$$

Sans perte de généralité, nous supposons que les éléments de $\mathcal{C}$ sont ordonnés de la manière suivante :

$$c(E^1) \leq c(E^2) \leq ... \leq c(E^{|\mathcal{C}|}).$$

Ainsi, nous appellerons E^i le $i^{ème}$ K-chemin de l'ensemble $\mathcal{C}$. Nous notons $V(E)$ le revenu optimal associé au K-chemin E.

Proposition 2. *Une solution optimale du problème de tarification sur un réseau est un K-chemin $E^* \in \mathcal{C}$ tel que :*

$$V(E^*) = \max\{V(E) | E \in \mathcal{C}\}.$$

Cette proposition permet de caractériser une solution optimale du PTR à partir d'une solution du problème du suiveur. De plus, il est facile d'obtenir une borne supérieure $\mathcal{B}(E)$ sur le revenu d'un K-chemin $E = (\mathcal{P}_1, ..., \mathcal{P}_{|\mathcal{K}|})$ en utilisant la borne supérieure sur le revenu du meneur généré par un chemin introduite par Labbé et al. (1998) (cf. section 2.3), ainsi :

$$V(E) \leq \mathcal{B}(E) = \sum_{k \in \mathcal{K}} n^k(\gamma^k(\infty) - (\sum_{a \in \mathcal{A}_1 \cap \mathcal{P}_k} c_a + \sum_{a \in \mathcal{A}_2 \cap \mathcal{P}_k} d_a)),$$

où $\gamma^k(\infty)$ représente le coût du plus court chemin de $o(k)$ à $d(k)$ si les tarifs sont fixés à ∞, c'est-à-dire le coût du plus court chemin non tarifable du produit k.

L'algorithme de Didi et al. (1999) peut être décrit de la manière suivante. Soit le K-chemin E^* représentant la meilleure solution courante et V^* le revenu associé à cette solution. À l'itération i, nous calculons d'abord la borne supérieure $\mathcal{B}_i = \mathcal{B}(E_i)$ sur le revenu du i^e K-chemin E^i. Si la valeur de cette borne supérieure est inférieure ou égale au meilleur revenu V^*, une solution optimale est trouvée. En effet, les valeurs $\mathcal{B}_i$ décroissent en fonction des itérations ($\mathcal{B}_i \geq \mathcal{B}_{i+1}$). Par contre, si $\mathcal{B}_i > V^*$, le revenu $V(E^i)$ est calculé en résolvant le problème d'optimisation inverse. Si la valeur du revenu $V(E^i)$ est égale à celle de la borne supérieure $\mathcal{B}_i$, une solution optimale est trouvée. Sinon nous avons $B_i > V(E^i)$. Dans ce cas, si $V(E^i) > V^*$, la meilleure solution courante et le meilleur K-chemin sont mis à jour. Le processus est itéré jusqu'à l'obtention d'une solution optimale.

Algorithme de Didi, Marcotte et Savard

Pas 0 *Initialisation*

- $E^* \leftarrow \emptyset$; $V^* \leftarrow 0$; $i \leftarrow 1$.

Pas 1 *Calcul du $i^{ème}$ K-chemin*

- Calculer le i^{ème} K-chemin de $\mathcal{C}$, $E^i = (\mathcal{P}_1^{i(1)}, ..., \mathcal{P}_{|K|}^{i(|K|)})$ (cf. section 4.2).
- $\mathcal{B}_i \leftarrow \sum_{k \in \mathcal{K}} n^k(\gamma^k(\infty) - (\sum_{a \in \mathcal{A}_1 \cap \mathcal{P}_k^{i(k)}} c_a + \sum_{a \in \mathcal{A}_2 \cap \mathcal{P}_k^{i(k)}} d_a))$.
- **Si** $B_i \leq V^*$ **alors** la solution optimale est E^*.

Pas 2 *Calcul du revenu du i^e K-chemin*

- Calculer de $V(E^i)$ en résolvant le problème d'optimisation inverse.
- **Si** $B_i = V(E^i)$ **alors** la solution optimale est E^i.

Pas 3 *Mise à jour de E^**

- **Si** $V^* < V(E^i)$ **alors** $V^* \leftarrow V(E^i)$; $E^* \leftarrow E^i$.
- $i \leftarrow i + 1$.

– Aller au pas 1.

Comme nous l'avons mentionné à la section 1.2, les performances de cette méthode deviennent mauvaises lorsque le nombre de produit est supérieur à 10 à cause de la mauvaise qualité des bornes supérieures et de l'explosion combinatoire du nombre de solution du problème du suiveur. Dans les sections suivantes, nous proposons plusieurs stratégies pour améliorer cette méthode.

4.2 Prétraitement sur la génération des k-plus courts chemins

Nous proposons un prétraitement pour la génération des chemins d'un produit visant à supprimer les chemins non réalisables pour le PTR mono-produit car les K-chemins comprenant ces chemins seront également non réalisables.

Un chemin est non réalisable pour le PTR s'il n'existe aucune politique tarifaire pour le générer. Illustrons l'existence de chemins non réalisables pour le PTR sur le réseau de la figure 4.1. Les arcs tarifables sont représentés en pointillés et la demande est donnée par trois produits (o_1, d_1), (o_2, d_1) et (o_1, d_2) de demande unitaire. Les chemins du produit (o_1, d_1) sont énumérés dans l'ordre de générations

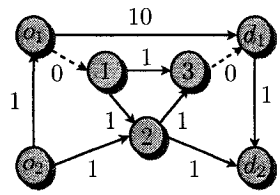


Figure 4.1 Cas de chemin non réalisable

des k-plus courts chemins avec les tarifs à zéro :

$$
\begin{aligned}
1: &\quad o_1 - 1 - 3 - d_1 \\
2: &\quad o_1 - 1 - 2 - 3 - d_1 \\
3: &\quad o_1 - 1 - d_1
\end{aligned}
$$

Il n'est pas possible de trouver un plan tarifaire pour le chemin 2 car le sous-chemin $1 - 2 - 3$ ne sera jamais plus court que celui $1 - 3$.

La détection de chemin non réalisable pour le PTR peut se faire en résolvant le POI mono-produit. En effet, un chemin est non réalisable s'il n'existe aucun ensemble de tarifs pouvant le rendre attractif pour le suiveur. En d'autres termes, le POI mono-produit n'admet pas de solution. Cette détection peut facilement être intégrée à l'algorithme de génération des k-plus courts chemins proposé par Yen (1971) afin de réduire le nombre de K-chemins non réalisables visités par la méthode exacte.

L'algorithme de Yen (1971) permet de calculer séquentiellement les J-plus courts chemins entre les deux noeuds d'un graphe. Initialement, le plus court chemin reliant ces noeuds est calculé. Puis, le chemin de coût minimum dans un ensemble de chemins contenant les chemins déviant à moindre coût des plus courts chemins calculés est le plus court chemin suivant. Pour cela, nous définissons le noeud de déviation d'un chemin P comme le noeud à partir duquel le chemin change de direction par rapport au dernier chemin calculé. Ainsi, dans l'exemple de la figure 4.2 où nous supposons que le dernier chemin calculé est $\mathcal{P}_1$, le noeud de déviation du chemin $\mathcal{P}_2$ est le noeud 1 car c'est à partir de ce noeud que les chemins $\mathcal{P}_1$ et $\mathcal{P}_2$ diffèrent.

Pour ne pas alourdir les notations, nous considérons le cas d'un produit d'origine o et de destination d. Soit $\mathcal{P}_j$ le $j^{ème}$ plus court chemin de o à d et $\mathcal{X}$ l'ensemble des chemins candidats pour être le $j^{ème}$ plus court chemin. Nous appelons $\mathcal{T}_j$ l'arbre de racine o regroupant les J plus courts chemins et les chemins candidats de $\mathcal{X}$.

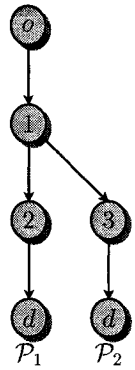


Figure 4.2 Exemple de noeud de déviation

Pour tout noeud $v \in \mathcal{N}$, $\mathcal{A}(v)$ représente l'ensemble des arcs ayant pour origine v et $\mathcal{A}_{\mathcal{T}_j}(v, \mathcal{P}_j)$ l'ensemble des arcs d'origine v ayant induit une déviation du chemin $\mathcal{P}_j$ dans $\mathcal{T}_j$. Le sous-chemin reliant le noeud u au noeud v d'un chemin $\mathcal{P}$ est noté $\mathcal{P}_{uv}$ et l'ensemble des noeuds le constituant $\mathcal{N}(P_{uv})$.

La procédure de calcul du j^{ème} plus court chemin entre o et d est la suivante. Le plus court chemin reliant o à d est le premier élément de l'ensemble de chemins candidats $\mathcal{X}$. Lors de la j^{ème} itération, le chemin de coût minimum dans $\mathcal{X}$ est appelé le j^{ème} plus court chemin $\mathcal{P}_j$. Ce chemin est remplacé dans $\mathcal{X}$ par un ensemble de nouveaux chemins déviant de $\mathcal{P}_j$ calculés à partir des noeuds situés après le noeud de déviation v_j car ceux obtenus à partir des noeuds du sous-chemin reliant o à v_j ont déjà été générés. Plus précisément, pour tout $v \in \mathcal{P}^j_{v_j d}$ dont il est possible de dévier, c'est-à-dire $A(v) \setminus A_{\mathcal{T}_j}(v) \neq \emptyset$, nous calculons, s'il existe, le plus court chemin $\mathcal{P}^*_{vd}$ sur le réseau privé des noeuds $\mathcal{N}(P^j_{ov})$(afin d'éviter un cycle) et des arcs $A_{\mathcal{T}_i}(v)$ (pour avoir une nouvelle déviation). Notons qu'une très bonne implantation de cette phase de calcul a été proposée par Martins et Pascoal (2003) suivant un principe de réoptimisation d'arbre de plus courts chemins. Finalement, l'algorithme de Yen peut donc être résumé de la façon suivante :

Calcul des J-plus courts chemins

Pas 0 *Initialisation*

- Calculer $\mathcal{P}_1$ le plus court chemin de o à d.
- Initialiser le compteur de chemins j à 1, l'ensemble des chemins candidats $\mathcal{X} = \{\mathcal{P}_j\}$ et l'arbre de Yen $\mathcal{T}_j = \{\mathcal{P}_j\}$.

Pas 1 *Critère d'arrêt*

- **si** $j == J$ ou $\mathcal{X} == \emptyset$ **alors** arrêt.

Pas 2 *Calcul du $j^{ème}$ plus court chemin*

- Supprimer le chemin $\mathcal{P}_j$ de $\mathcal{X}$ ($\mathcal{X} = \mathcal{X} \setminus \{\mathcal{P}_j\}$).
- Soit v_j le noeud de déviation de $\mathcal{P}_j$.
- Pour tout $v \in P^j_{v_j d}$,
  si $(A(v) \setminus A_{\mathcal{T}_j}(v) \neq \emptyset)$ **alors**
  - Calculer le plus court chemin P^*_{vd} dans $G' =< \mathcal{N} \setminus \mathcal{N}(P^j_{ov}), \mathcal{A} \setminus A_{\mathcal{T}_j}(v) >$.
  - **si** P^*_{vd} existe **alors** $\mathcal{X} = \mathcal{X} \cup \{\mathcal{P}^j_{ov} + \mathcal{P}^*_{vd}\}$; $\mathcal{T}_j = \mathcal{T}_j + \mathcal{P}^*_{vd}$.
- $j = j + 1$.
- $\mathcal{T}_j = \mathcal{T}_{j-1}$.
- Sélectionner $\mathcal{P}_j$ le plus court chemin de l'ensemble $\mathcal{X}$.
- aller au pas 1.

Illustrons les différentes étapes de l'algorithme de Yen sur l'exemple de la figure 4.3. Les trois premières itérations sont données à la figure 4.4 où l'arbre $\mathcal{T}_j$ pour chaque itération j et l'ensemble des chemins candidats $\mathcal{X}$ sont reportés (le nombre à coté de chaque noeud représente le coût du chemin de o à ce noeud).

Initialement, le plus court chemin reliant o à d est $\mathcal{P}_1 = o - 3 - 4 - d$ et le noeud de déviation initial est $v_1 = o$. Ainsi, l'ensemble $\mathcal{X} = \{\mathcal{P}_1\}$, $\mathcal{T}_1$ est le chemin $\mathcal{P}_1$. Nous commençons par dévié $\mathcal{P}_1$ à partir du noeud $v_1 = o$. Les arcs ayant le noeud o comme origine sont $(o, 2)$,$(o, 3)$ et (o, d). Comme $(o, 3)$ appartient à $\mathcal{P}_1$, nous cherchons le plus court chemin de o à d n'empruntant pas $(o, 3)$ c'est-à-dire le chemin $\mathcal{P}'_2 = o - 2 - 3 - 4 - d$. De la même manière pour le noeud 3, nous obtenons le chemin $\mathcal{P}'_3 = o - 3 - d$. Notons que $\mathcal{P}_1$ ne peut être dévier à partir des noeuds 4

et d. Donc, P'_2 et P'_3 remplacent $\mathcal{P}_1$ dans $\mathcal{X}$ et sont ajoutés à $\mathcal{T}_1$ pour former l'arbre $\mathcal{T}_2$. Finalement, le deuxième plus court chemin $\mathcal{P}_2$ est $\mathcal{P}'_2 = o-2-3-4-d$ de coût 4. En réitérant, nous obtenons comme troisième plus court chemin $\mathcal{P}_3 = o-3-d$ de coût 10.

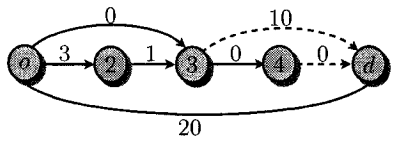


Figure 4.3 Réseau de base pour illustrer l'algorithme de Yen

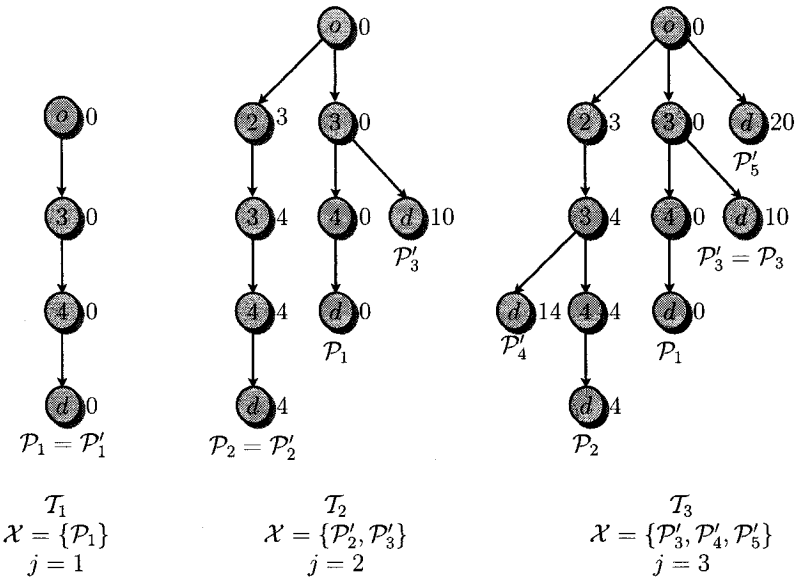


Figure 4.4 Les trois premières itérations de l'algorithme de Yen

Supposons maintenant que les arcs en pointillés sur la figure 4.3 sont tarifables. Dans ce cas, nous ne pouvons déterminer de tarifs tels que $\mathcal{P}_2$ soit attractif pour le suiveur car le sous-chemin non tarifable $o-2-3$ est plus long que le chemin non tarifable $o-3$. Donc, le chemin $\mathcal{P}_2$ est non réalisable à cause du sous-chemin $o-2-3$, et par conséquence tous les chemins déviant de $\mathcal{P}_2$ à partir des noeuds 3 ou 4 sont aussi non réalisables comme par exemple $\mathcal{P}'_4$.

L'algorithme de Yen est adapté de la manière suivante pour éviter les chemins

non réalisables. À la j^{ème} itération, la réalisabilité du chemin $\mathcal{P}_j$ est déterminé en résolvant le POI mono-produit. Si, le POI admet une solution, $\mathcal{P}_j$ est le $j^{ème}$ plus court chemin. Dans le cas contraire, $\mathcal{P}_j$ est remplacé par les nouveaux chemins potentiellement réalisables dans $\mathcal{X}$ et l'itération j est répétée en choisissant un nouveau chemin $\mathcal{P}_j$ dans $\mathcal{X}$.

Finalement en remplaçant le pas 3 dans l'algorithme de Yen par les pas 2' et 3 suivant, nous obtenons un algorithme de k-plus courts chemins adapté au PTR.

Pas 2' *Mise à jour* $\mathcal{X}$

- $\mathcal{X} = \mathcal{X} \setminus \{\mathcal{P}_j\}$.
- Pour tout $v \in P^j_{v_j d}$,
  si $(A(v) \setminus A_{\mathcal{T}_j}(v) \neq \emptyset)$ **alors**
  - Calculer le plus court chemin P^*_{vd} sur $G' =< \mathcal{N} \setminus \mathcal{N}(P^j_{ov}), \mathcal{A} \setminus A_{\mathcal{T}_j}(v) >$.
  - **si** P^*_{vd} existe **alors** $\mathcal{X} = \mathcal{X} \cup \{\mathcal{P}^j_{ov} + \mathcal{P}^*_{vd}\}$; $\mathcal{T}_j = \mathcal{T}_j + \mathcal{P}^*_{vd}$.
- $j = j + 1$.

Pas 3 *Calcul du* $j^{ème}$ *plus court chemin*

- Sélectionné $\mathcal{P}_j$ le plus court chemin de l'ensemble $\mathcal{X}$
- **Si** le problème d'optimisation inverse mono-produit avec $\mathcal{P}_j$ admet une solution réalisable **alors**
  - $\mathcal{T}_j = \mathcal{T}_{j-1}$.
  - aller au pas 1.
- **sinon**
  - $\mathcal{X} = \mathcal{X} \setminus \{\mathcal{P}_j\}$.
  - aller au pas 2'.

L'application des deux premières itérations de l'algorithme modifié sur le réseau de la figure 4.3 est reporté à la figure 4.5. Le plus court chemin de o à d est $\mathcal{P}_1 = o - 3 - 4 - d$ et la détermination de $\mathcal{P}_2 = o - 2 - 3 - 4 - d$ est similaire à

ce qui a été décrit précédemment. P_2 étant non réalisable, il doit être supprimé de $\mathcal{X}$ et remplacé par les chemins potentiellement réalisables en déviant. Le noeud de déviation de $\mathcal{P}_2$ est $v_2 = o$. Les arcs ayant le noeud o comme origine sont $(o,2)$,$(o,3)$ et (o,d). Or, $(o,2)$ et $(o,3)$ appartiennent aux chemins $\mathcal{P}_1$ et $\mathcal{P}_2$, nous cherchons alors uniquement le plus court chemin de o à d empruntant $(o,4)$ ($\mathcal{P}'_4 = o-d$). Le sous-chemin se réduisant au noeud o, il est trivialement réalisable. Pour $v = 2$, il n'est pas possible de dévier le chemin P_2. Pour $v = 3$, le POI appliqué au sous chemin $o-2-3$ n'admet pas de solution. Aucun chemin réalisable ne peut donc être généré avec des noeuds situés après le noeud 3 dans le chemin $\mathcal{P}_2$. Ainsi, les noeuds 3, 4 et d sont hachurés sur la figure 4.5. Le chemin $\mathcal{P}_2$ est remplacé par le chemin $\mathcal{P}'_4$ dans $\mathcal{X}$ et $\mathcal{P}'_4$ est inséré dans $\mathcal{T}_1$. Finalement, l'ensemble $\mathcal{X}$ est égale à $\{\mathcal{P}'_3, \mathcal{P}'_4\}$ et le deuxième plus court chemin réalisable est $\mathcal{P}_2 = \mathcal{P}'_3 = o-3-d$.

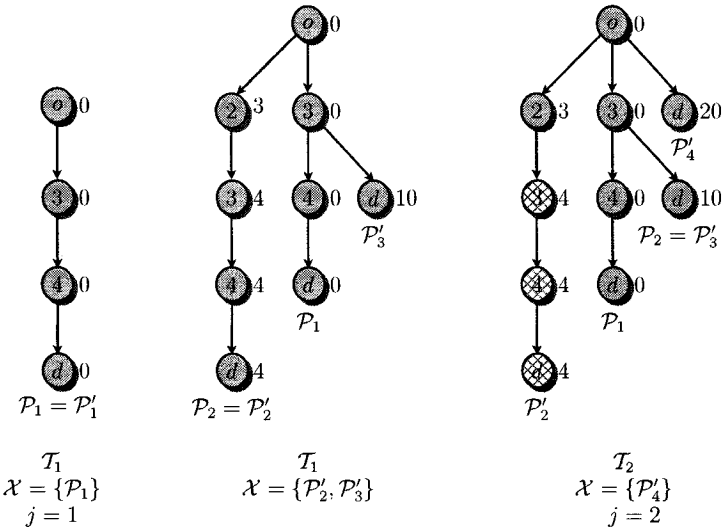


Figure 4.5 Les deux premières itérations de l'algorithme de Yen adapté au PTR

4.3 Amélioration de l'énumération des K-chemins

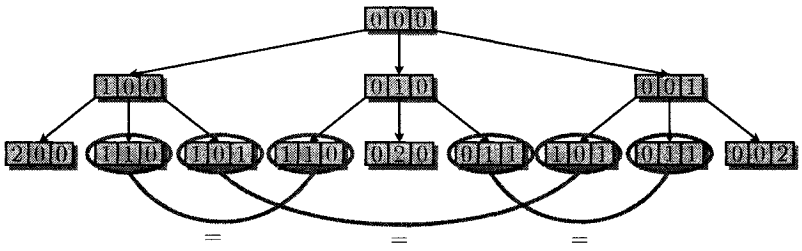
Un K-chemin indique pour chaque produit le chemin emprunté par le suiveur. Le chemin de chaque est identifié à partir de son rang par rapport à l'algorithme de Yen présenté à la section précédente. Ainsi, un chemin de rang zéro correspond au plus court chemin, un chemin de rang 1 au deuxième plus court chemin, ...

La méthode exacte proposée par Didi *et al.* repose sur l'énumération de ces K-chemins. Elle commence par le K-chemin initial composé des plus court chemin de chaque produit. Puis de nouveau K-chemins sont obtenus en incrémentant le rang du chemin utilisé d'un seul produit.

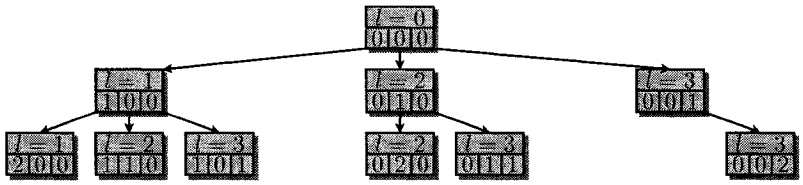
La figure4.6(a) illustre cette énumération lorsque trois produits sont considérés. À partir du K-chemin initial $(0, 0, 0)$, trois nouveaux K-chemins sont obtenus : $(1, 0, 0)$, $(0, 1, 0)$ et $(0, 0, 1)$. Cependant certains K-chemins peuvent être générés plusieurs fois comme le K-chemin $(1, 1, 0)$ qui peut être obtenu à partir de $(1, 0, 0)$ et $(0, 1, 0)$ en changeant respectivement le rang du chemin du deuxième et premier produit.

Pour éliminer les redondances lors du processus de génération de K-chemins, nous associons d'abord une étiquette l à chaque K-chemin. Cette étiquette indique l'indice du dernier produit dont le rang d'un chemin a changé. Par exemple, le K-chemin $(1, 0, 0)$ aura comme étiquette $l = 1$ car il a été obtenu en modifiant le rang du chemin du produit 1 du K-chemin $(0, 0, 0)$.

Ensuite, nous imposons la règle qu'un K-chemin ne peut être obtenu qu'en modifiant le rang du chemin d'un produit d'indice supérieur ou égal à l'étiquette. Ainsi, le K-chemin $(1, 1, 0)$ ne peut plus être généré en modifiant le chemin du premier produit à partir de $(0, 1, 0)$ car l'étiquette l de ce dernier est égale à 2. La figure



(a) Énumération classique des K-Chemins



(b) Énumération des K-Chemins

Figure 4.6 Énumération des K-chemins

4.6(b) représente la nouvelle procédure d'énumération.

4.4 Bornes supérieures sur le revenu engendré par un K-chemin

La définition de l'étiquette lors du processus de génération des K-chemins induit une nouvelle structure de données dont l'information peut être exploitée pour le calcul de meilleures bornes supérieures sur le revenu engendré par un K-chemin. Rappelons que le principe de décroissance des valeurs des bornes supérieures doit être respecté pendant le processus de génération des K-chemins.

L'étiquette décompose un K-chemin (cf. figure 4.7) en trois parties : le *préfixe*, l'*étiquette* et le *suffixe*. Le *préfixe* est l'ensemble des produits dont l'indice est strictement inférieur à l'étiquette l. Cet ensemble est fixe puisque par définition le chemin d'un produit $j < l$ ne peut être modifié. Tous les descendants d'un K-chemin hérite donc de son préfixe. L'*étiquette* l est l'indice du dernier produit ayant changé de chemin mais, par un abus de langage, nous l'utiliserons pour désigner le produit

d'indice l. Le *suffixe* est l'ensemble des produits d'indice strictement supérieur à l'étiquette l. Cette ensemble n'est pas fixe mais chaque produit du suffixe utilise un plus court chemin à tarif nul. Par conséquence tous les K-chemins d'étiquette l ont le même suffixe.

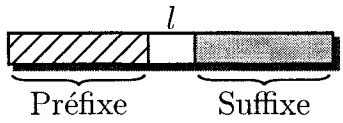


Figure 4.7 Décomposition d'un K-chemin

Une borne supérieure sur le revenu engendré par les produits du préfixe est facilement obtenue en résolvant le POI en fixant les chemins associés au préfixe. La qualité de cette borne est la meilleure possible pour le préfixe. De plus, si le problème d'optimisation inverse n'admet aucune solution, tous les K-chemins avec ce préfixe sont non réalisables et une branche de K-chemins non réalisables peut être élaguée dans le processus de recherche.

Une borne supérieure sur le revenu engendré par le produit de l'étiquette peut être définie comme la différence entre le coût du chemin non tarifable du produit d'indice l et le coût à tarif nul du chemin associés à l'étiquette (cf. section 2.3). Bien que de qualité médiocre, cette borne est compatible avec le processus de génération des K-chemins puisqu'elle garantit la décroissance des valeurs des bornes supérieures.

Une borne supérieure sur le revenu engendré par les produits du suffixe de bonne qualité est calculée en exploitant le fait que tous les K-chemins avec une étiquette l ont le même suffixe noté $suffixe(l)$. Il existe donc $|\mathcal{K}| - 1$ suffixes différents. Le calcul des bornes supérieures : $B(suffix(1)), \ldots, B(suffix(|\mathcal{K}| - 1))$ peut se faire en appliquant la méthode exacte de façon récursif. Plus précisément, une borne supérieure sur le revenu du PTR avec les produits du suffixe est calculée par la méthode exacte de manière heuristique.

Finalement, une borne supérieure sur le revenu d'un K-chemin est définie par la proposition suivante.

Proposition 3. *Soit un K-chemin E_l réalisable d'étiquette l. Une borne supérieure sur le revenu de E_l est définie par :*

$$\tilde{B}(E_l) = B_{Prefixe}(E_l) + \min\{B_{Suffixe}(l-1), B_P(E_l) + B_{Suffixe}(l)\}$$

où $B_{Prefixe}(E_l)$ est une borne supérieure sur le revenu engendré par les produits du préfixe, $B_P(E_l)$ est une borne supérieure sur le revenu du chemin utilisé par le produit l et $B_{Suffixe}(l)$ est une borne supérieure sur le revenu engendré par les produits du suffixe.

Démonstration. Soit un K-chemin E_l d'étiquette l, et des bornes supérieures $B_{Prefixe}(E_l)$, $B_P(E_l)$ et $B_{Suffixe}(l)$ sur le revenu engendré par les produits du préfixe, de l'étiquette et du suffixe de E_l. Notons $v(E_l)$ la valeur du revenu optimal associée à un K-chemin E_l.

Appliquons un raisonnement par l'absurde en supposant que $\tilde{B}(E_l) < v(E_l)$. Deux cas sont possibles :

Cas 1 : *Supposons que* $\tilde{B}(E_l) = B_{Prefixe}(E_l) + B_{Suffixe}(l-1)$. Or, $B_{Prefixe}(E_l) + B_{Suffixe}(l-1) < v(E_l) = v(E_l)_{Prefixe} + v(E_l)_{Suffixe}(l-1)$ où $v(E_l)_{Prefixe}$ (resp. $v(E_l)_{Suffixe}(l-1)$) est la contribution au revenu engendré par les produits du préfixe (resp. du suffixe $l-1$) de E_l.

Cas 1.1 : *Supposons que* $B_{Prefixe}(E_l) \leq v(E_l)_{Prefixe}$. La solution engendrant $v(E_l)_{Prefixe}$ est une solution réalisable pour le préfixe (car E_l est réalisable) donc $v(E_l)_{Prefixe} \leq B_{Prefixe}(E_l)$ ce qui contredit l'hypothèse que $B_{Prefixe}(E_l)$ est une borne supérieure sur le revenu engendré par le préfixe de E_l.

Cas 1.2 : *Supposons que* $B_{Suffixe}(l-1) \leq v(E_l)_{Suffixe}(l-1)$. La solution engendrant $v(E_l)_{Suffixe}(l-1)$ est une solution réalisable pour le suffixe $l-1$ (car E_l est réalisable) donc $v(E_l)_{Suffixe}(l-1) \leq B_{Suffixe}(l-1)$ ce qui contredit l'hypothèse que $B_{Suffixe}(l-1)$ est une borne supérieure sur le revenu engendré par le suffixe $l-1$ de E_l.

Cas 2 : *Supposons que* $\tilde{B}(E_l) = B_{Prefixe}(E_l) + B_P(E_l) + B_{Suffixe}(l)$. Par un raisonnement similaire, l'hypothèse $\tilde{B}(E_l) < v(E_l)$ peut être contredite.

Les cas 1 et 2 nous permettent de conclure que $\tilde{B}(E_l)$ définit une borne supérieure sur le revenu généré par E_l. □

La proposition suivante permet de définir un lien entre les bornes supérieures sur les revenus d'un K-chemin et leurs descendants.

Proposition 4. *Soit un K-chemin* E_l *avec une étiquette* l *et un* $E'_{l'}$ *un descendant de* E_l *alors* $\tilde{B}(E_l)$ *est une borne supérieure sur le revenu du K-chemin* $E'_{l'}$.

La preuve de la proposition 4 est similaire à celle de la proposition 3. Le calcul d'une borne supérieure sur le revenu du meneur définit dans les propositions 3 et 4 est illustré à la figure 4.8.

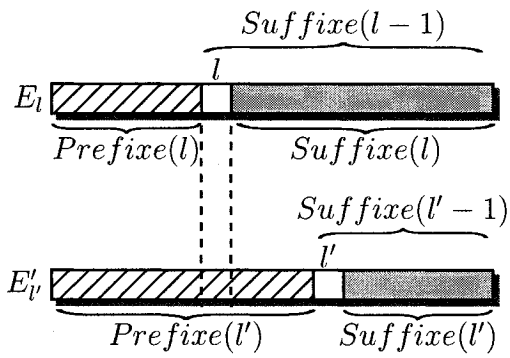


Figure 4.8 Borne supérieure sur le revenu d'un K-chemin à partir des différentes parties d'un K-chemin

Finalement, la proposition 5 nous assure la décroissance des bornes supérieures sur le revenu lors du processus de génération des K-chemins.

Proposition 5. *Soit un K-chemin E_l avec une étiquette l et $E'_{l'}$ un descendant de E_l. Alors $B(E'_{l'}) = \min\{B(E_l), \tilde{B}(E'_{l'})\}$ est une borne supérieure sur le revenu du K-chemin $E'_{l'}$ et $B(E_l) \geq B(E'_{ls'})$.*

Le résultat de cette proposition est une conséquence directe des propositions 3 et 4.

4.5 Heuristique

Nous décrivons une heuristique de type glouton permettant d'améliorer la qualité de la borne inférieure sur le revenu du meneur. Pour cela, nous introduisons le voisinage de degré $u \in I\!N$ associé au K-chemin E_l^j que nous noterons $\mathcal{C}(u, E_l^j)$. Il est défini par :

$$\mathcal{C}(u, E_l^j) = \{(p_1, p_2, ..., p_{|\mathcal{K}|}) \in \mathcal{C} | \exists k' \in [l, |\mathcal{K}|] \text{ tel que } p_k = \left\{ \begin{array}{ll} \mathcal{P}_k^{rg(k)+u}, & \text{si } k' = k, \\ \mathcal{P}_k^{rg(k)}, & \text{sinon,} \end{array} \right\}$$

où $\mathcal{P}_k^{rg(k)}$ est le chemin de rang $rg(k)$ utilisé par le produit k. Notons que si $u = 1$, ce voisinage est identique à celui de la méthode d'énumération décrite à la section 4.3.

L'heuristique repose sur l'exploration du voisinage présenté en faisant varier le paramètre u d'une itération à l'autre afin de visiter des solutions situées à une "certaine distance" de la solution courante. Cette stratégie permet d'explorer des solutions de plus en plus éloignées de la solution courante afin de permettre la découverte de solutions de meilleures qualités. Plus précisément, l'heuristique fonc-

tionne de la manière suivante. Pour une valeur de u fixée, les solutions de $C(u, E_l^i)$ sont évaluées en résolvant un POI. Si aucune solution améliorante n'est trouvée, u est incrémenté de un et une nouvelle exploration est commencée. Par contre, si une solution améliorante est trouvée, la solution courante est mise à jour et u est réinitialisée à 1. Le processus est répété jusqu'à ce que la valeur u atteigne un certain seuil. L'algorithme de cette heuristique est détaillé dans ce qui suit.

Heuristique

Pas 1 *Initialisation*

- Soit E un K-chemin et $V(E)$ le revenu associé.
- $u = 1$.

Pas 2 *Évaluation du voisinage*

- Soit E^* le meilleur K-chemin du voisinage $\mathcal{C}(u, E)$.

Pas 3 *Ajustement de u et mise à jour de la meilleure solution*

- **Si** $V(E^*) > V(E)$ **alors** $E = E^*$, $u = 1$, aller au Pas 2,
- **Sinon** $u = u + 1$, aller au Pas 3.

Pas 4 *Condition d'arret*

- **Si** $u > U$ **alors** arrêt,
- **Sinon** aller au Pas 2.

4.6 Algorithme de la méthode exacte

Nous présentons maintenant l'algorithme de la méthode exacte basée sur la méthode proposée par Didi *et al.* pour le PTR. L'algorithme se décompose en deux phases : la phase des suffixes et la phase de résolution. La phase des suffixes consiste à déterminer les bornes supérieures engendrées par chaque suffixe. Un calcul efficace nécessite de commencer par l'évaluation de $B(suffixe(|\mathcal{K}| - 1))$,

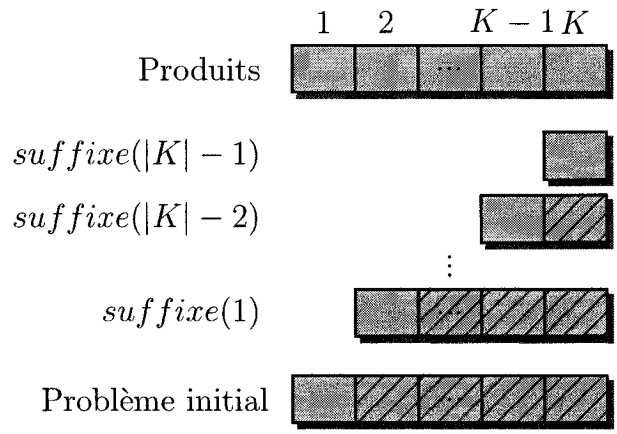


Figure 4.9 calcul des bornes supérieures sur le revenu des suffixes

puis, $B(suffixe(|\mathcal{K}-2|))$ en exploitant $B(suffixe(|\mathcal{K}-1|))$ et ainsi de suite. Ce processus est illustré à la figure 4.9. Le problème initial est résolu lorsque toutes les bornes supérieures sur le revenu des suffixes ont été calculées.

Nous présentons ci-dessous l'algorithme de résolution permettant le calcul des bornes supérieures sur le revenu des suffixes. Deux paramètres d'entrée sont nécessaires : le nombre d'itérations maximal *iter* et l'indice de produit k permettant de ne considérer que le sous-ensemble de produits $k, \ldots, |\mathcal{K}|$. À la fin de la résolution, le résultat comprend le meilleur K-chemin E^* et la valeur de la borne supérieure B^* pour le PTR réduit au sous ensemble de produits $k, \ldots, |\mathcal{K}|$.

DMS(*iter*,*k*)

Pas 0 *Initialisation*

- Soit le compteur d'itération $i = 1$.
- $E^* = NUL, B^* = +\infty$.
- Soit la liste des K-chemins candidats $LIST = \{E_0\}$ où $E_0 = (\mathcal{P}^0_k, ..., \mathcal{P}^0_{|\mathcal{K}|})$.

Pas 1 *Sélection du K-chemin*

- Sélectionner E_l tel que $B(E_l) = \max\{B(E'_{l'})|E'_{l'} \in LIST\}$.
- Mise à jour de la borne supérieure $B^* = B(E_l)$.

Pas 2 *Évaluation de* E_l

- Calculer le revenu $V(E_l)$ associé au K-chemin E_l
- **si** $V(E^*) < V(E_l)$ **alors**
  - Appeler l'heuristique présentée à la section 4.5 à partir de E_l.
  - Poser E^* égale à la solution retournée par l'heuristique

Pas 3 *Critères d'arrêt*

- **Si** $B^* \leq V(E^*)$ **alors** arrêt car la solution optimale est trouvée.
- **Si** $i > iter$ **alors** arrêt car le nombre d'itérations maximal est atteint.

Pas 4 *Mise à jour de* $LIST$

- Calculer les descendants de E_l et leurs bornes supérieures associées suivant la

  proposition 5.
- Ajouter les descendants de E_l dont le préfixe est réalisable à $LIST$.
- $i \leftarrow i + 1$.
- Aller au pas 1.

Cet algorithme de résolution est utilisé dans l'algorithme de la méthode exacte suivant :

Algorithme de la méthode exacte basée sur le concept de K-chemins

Pas 0 *Initialisation*

- Initialisation des ensembles de chemins de chaque produit avec l'algorithme de k-plus courts chemins de la section 4.2
- $k = |\mathcal{K}|$.
- $E^{**} = NUL$.

Pas 1 *Phase des suffixes*

- (B^*, E^*)=DMS($Maxiter$,k).
- Soit $\tilde{E}^*$ la meilleure solution de l'heuristique présentée à la section 4.5 à partir du K-chemin dont les produits de $k, \ldots, |\mathcal{K}|$ utilisent les mêmes

chemins que E^* et les produits de $1, \ldots, k-1$ les plus courts chemins calculés pour des tarifs fixés à 0.

- **Si** $V(E^{**}) < V(\tilde{E}^*)$ **alors** $E^{**} = \tilde{E}^*$.
- $Suffixe(k) = B^*$, $k = k - 1$.
- **si** $k > 1$ **alors** aller au pas 1.

  sinon aller au Pas 2

Pas 2 *Résolution du problème initial*

- (B^*, E^*) =DMS($+\infty$,1).
- **Si** $V(E^{**}) < V(E^*)$ **alors** $E^{**} = E^*$.

Au pas 1, nous exploitons le principe que toute borne inférieure sur le revenu d'un suffixe est aussi une borne inférieure sur le revenu associé au problème initial lorsque les produits non utilisés empruntent un chemin non tarifable. Nous appliquons l'heuristique décrite à la section 4.5 à partir de cette solution pour mettre à jour si nécessaire la meilleure solution de PTR en se basant sur tous les produits.

Il n'est pas possible de conserver les arbres de Yen de chaque produit au cours du processus de génération des K-plus courts chemins. Donc, lors de la génération d'un nouveau chemin, nous devons utiliser l'algorithme de Yen comme si aucun chemin n'avait été généré. Pour diminuer le temps de calcul de l'algorithme de Yen, nous avons choisi de générer des ensembles de chemins de la manière suivante. Pour chaque produit, l'ensemble de chemins est initialisé par l'algorithme de Yen avec les chemins ayant un coût inférieur au plus court chemin non tarifable. Par la suite, lorsqu'un nouveau chemin doit être généré pour un produit, nous générons par l'algorithme de Yen autant de chemins que nécessaire pour doubler la taille de l'ensemble des chemins de ce produit.

4.7 Résultats Numériques

Avant de présenter les résultats numériques obtenus à partir de la méthode exacte, nous précisons quelques détails sur son implantation en C++. Les formulations linéaires et les formulations linéaires mixtes ont été résolues par le logiciel commercial CPLEX (2006) dans sa version 9.0. Les résolutions des problèmes de plus courts chemins ont été effectuées par l'algorithme proposé par Tarjan (1981). Précisons que l'ensemble des tests ont été réalisés sur une machine possédant un processeur AMD Opteron(tm) Processor 248 de 3.00 GHz.

4.7.1 Impact des bornes supérieures sur le revenu généré par des K-chemins sur les performances de la méthode exacte

Dans cette section, nous étudions en quoi les nouvelles bornes supérieures sur le revenu du meneur évaluées à partir du préfixe, de l'étiquette et du suffixe permettent d'accélérer la détection de la solution optimale. Ces tests ont été effectués sur les instances proposées par Brotcorne et al. (2001) avec 15% d'arcs tarifables avec un nombre de produits variant entre 10, 20, 30 et 40 et une contrainte de positivité imposée aux tarifs. Pour plus de détails, nous renvoyons le lecteur à l'annexe I.

Les résultats sont reportés dans les tableaux 4.1 et 4.2 où sont indiqués le nombre de produits (#OD), le pourcentage d'arcs tarifables ($\%T$), les contraintes de borne inférieure sur les tarifs (T), la valeur de la meilleure solution trouvée (Sol), la qualité de celle-ci (%) définie par l'écart entre la valeur de la borne supérieure Z_{sup} et la valeur de la meilleure solution trouvée : $(Z_{sup} - \text{Sol})/\text{Sol}$, le nombre d'itérations (#it) et le temps de calcul en seconde (CPU) que nous avons borné à 2 heures (7200 secondes). Pour chaque méthode testée, la moyenne sur 5 instances est rapportée.

Tableau 4.1 Comparaison des méthodes exactes DMS et DMS_PREF

Instances			DMS_INIT				DMS_PREF			
#OD	%T	T	Sol	%	#it	CPU	Sol	%	#it	CPU
10	15	≥ 0	3183.20	1.85	2568653.50	3578.97	3183.20	0.00	17858.20	78.22
20	15	≥ 0	4859.00	22.46	2086527.25	7200.01	4859.00	11.69	899430.62	7200.01
30	15	≥ 0	6096.00	28.12	1336290.38	7200.01	6096.00	19.07	543383.19	7200.01
40	15	≥ 0	6708.60	36.76	611609.38	7200.02	6708.60	30.91	376874.81	7200.01

4.7.1.1 Borne supérieure sur le revenu généré par le préfixe

Nous étudions l'impact du calcul des bornes supérieures sur le revenu engendré par le préfixe. Pour cela, nous comparons les solutions, obtenues par la méthode exacte améliorée, où les bornes supérieures sont calculées comme dans la version initiale proposée par Didi et al. (1999) (DMS_INIT) et la méthode exacte où les bornes supérieures sur le revenu sont calculées à partir de la décomposition présentée à la section 4.4 mais où la borne supérieure sur le revenu des suffixes est donnée par la somme des bornes supérieures proposées par Labbé et al. (1998) sur le revenu de chaque chemin (DMS_PREF). Les résultats numériques sont reportés dans le tableau 4.1.

Nous remarquons tout d'abord que les bornes inférieures sur le revenu, obtenues à partir de l'heuristique décrite à la section 4.5, sont identiques pour les deux méthodes. Ensuite, comme indiqué par le nombre d'itérations, la méthode DMS_INIT explore deux fois plus de solutions que DMS_PREF pour un temps donné. Ceci est dû aux deux évaluations nécessaires de chaque K-chemin dans la méthode DMS_PREF : une première pour le calcul la borne supérieure sur le revenu du préfixe et une deuxième pour le revenu du K-chemin.

Enfin, nous remarquons que les bornes supérieures sur le revenu d'un K-chemin sont de meilleures qualités avec DMS_PREF. En effet, en ce qui concerne les instances de 10 produits, la solution optimale a été trouvée avec DMS_PREF contrai-

rement à DMS_INIT. Pour les instances de plus de 20 produits, même si aucune des deux méthodes ne permet d'obtenir la solution optimale ou de prouver l'optimalité de la meilleure solution, la qualité des solutions retournées par DMS_PREF est 10% meilleure que celle de DMS_INIT. Notons que ces résultats non seulement proviennent de l'amélioration de la qualité des bornes supérieures sur le revenu calculées par DMS_PREF mais aussi grâce à la suppression des branches de l'arbre correspondant à des K-chemins avec des préfixes non réalisables.

4.7.1.2 Borne supérieure sur le revenu généré par les suffixes

Nous nous intéressons à présent à l'impact du calcul de la borne supérieure sur le revenu généré par le suffixe. Considérons la méthode DMS_PREF_SUFF($Maxiter$) composée de DMS_PREF où les bornes supérieures sur le revenu engendré par les suffixes sont calculées suivant le principe présenté à la section 4.4 et où $Maxiter$ est le nombre maximal d'itérations de DMS pour le calcul des bornes supérieures sur les suffixes (cf. section 4.6). Ce nombre varie entre 0, 500, 3000 et 5000. Notons que lorsque que $Maxiter = 0$, la méthode DMS_PREF_SUFF(0) est identique à DMS_PREF de la section précédente. Les résultats numériques sont donnés dans le tableau 4.2 où nous avons ajouté une colonne (SUFF) représentant les temps de calcul nécessaire pour le calcul des bornes supérieures sur les suffixes.

Nous remarquons tout d'abord que pour les instances comprenant 10 produits où la solution optimale a été trouvée, le nombre d'itérations requis pour prouver l'optimalité semble réduire logarithmiquement en fonction de $Maxiter$. Plus précisément, ce nombre est divisé par 5 lorsque $iter$ augmente de 0 à 500 itérations, il varie très peu pour une augmentation de $Maxiter$ de 500 à 3000 itérations et ne change pas pour une variation de $Maxiter$ de 3000 à 5000. D'autre part, les temps calcul totaux sont 4 fois plus petits si $Maxiter$ est supérieur à 500 même

Tableau 4.2 Impact du nombre d'itérations pour le calcul des bornes sur le revenu généré par les suffixes

			DMS_PREF_SUFF(0)				DMS_PREF_SUFF(500)			
#OD	%T	T	%	#it	CPU	SUFF	%	#it	CPU	SUFF
10	15	≥ 0	0.00	17858.20	78.22	-	0.00	3650.00	25.16	8.18
20	15	≥ 0	11.69	899430.62	7200.01	-	4.25	751234.38	7200.01	62.68
30	15	≥ 0	19.07	543383.19	7200.01	-	9.72	442866.00	7200.01	209.45
40	15	≥ 0	30.91	376874.81	7200.01	-	19.44	310426.59	7200.01	422.75

			DMS_PREF_SUFF(3000)				DMS_PREF_SUFF(5000)			
#OD	%T	T	%	#it	CPU	SUFF	%	#it	CPU	SUFF
10	15	≥ 0	0.00	3319.00	27.42	12.04	0.00	3319.00	27.53	12.12
20	15	≥ 0	3.57	737598.62	7200.01	200.26	3.34	798337.19	7200.01	274.17
30	15	≥ 0	8.66	423321.00	7200.02	644.87	8.44	399673.81	7200.01	959.93
40	15	≥ 0	17.69	267324.81	7200.01	1307.54	17.32	234493.80	7200.01	1979.97

si les temps de calcul des bornes supérieures sur le revenu généré par les suffixes semblent augmenter de manière logarithmique en fonction de *Maxiter*.

Pour les instances où la solution optimale n'a pu être obtenue en moins de 2 heures, la qualité des solutions en terme d'écart entre la borne supérieure et inférieure sur le revenu du meneur semble augmenter logarithmiquement en fonction de *Maxiter*. Par exemple pour les instances comprenant 40 produits, une augmentation de *Maxiter* de 0 à 500 induit une amélioration d'environ 11% de la qualité des solutions, l'augmentation de 500 à 3000 itérations génère une l'amélioration d'environ 2% et enfin l'augmentation de 3000 à 5000 résulte en une amélioration de 0.4%. Par ailleurs, le temps de calcul nécessaire pour le calcul des bornes supérieures sur le revenu engendré par les suffixes semble croître linéairement en fonction de *Maxiter*.

La qualité des bornes supérieures sur le revenu des suffixes et l'évolution des temps de calcul de ces bornes s'expliquent par l'amélioration logarithmique de la qualité des bornes supérieures au travers des deux cas suivants

Cas 1 : si le nombre d'itérations nécessaires pour trouver la borne supérieure optimale d'un suffixe est inférieur à *Maxiter*, alors l'augmentation de *Maxiter* ne modifie pas les temps de calcul.

Cas 2 : si le nombre d'itérations nécessaires pour trouver la borne supérieure optimale d'un suffixe est supérieur à *Maxiter*, alors l'augmentation de *Maxiter* implique une augmentation proportionnelle des temps de calcul.

Ainsi, pour les instances comportant 10 produits, l'augmentation de *Maxiter* permet lors du calcul des bornes d'obtenir de plus en plus de bornes supérieures optimales sur le revenu des suffixes impliquant une stabilisation des temps de calculs de chaque borne et donc de la phase des suffixes. Pour les autres instances, l'augmentation de *Maxiter* n'est pas suffisante pour permettre le calcul d'un nombre suffisant de bornes supérieures optimales sur le revenu des suffixes. Donc, les temps de calcul pour chacune des bornes non optimales restantes augmente linéairement en fonction de *Maxiter*.

4.7.2 Comparaison entre la méthode exacte à base de K-chemins et la résolution de la formulations MIP

Dans cette section, nous comparons les performances de la méthode exacte décrite dans cette thèse avec celles de la résolution de la formulation MIP présentée à la section 2.5 par le solveur commercial CPLEX (2006).

Pour la méthode exacte avec les K-chemins, nous utilisons la version DMS_PREF_SUFF(3000) présentée à la section précédente que nous renommons DMS_PREF_SUFF. Nous considérons deux versions pour la résolution de la formulation MIP. La première (MIP) repose sur la méthode de séparation et d'évaluation par défaut de CPLEX. Dans la deuxième version (MIP$^+$), nous avons ajouté une procédure améliorant la qualité des bornes inférieures à CPLEX. Cette procédure

calcule la solution du POI à partir de la solution du problème du suiveur associée aux tarifs fournis par la solution de la relaxation. Notons que les constantes M apparaissant dans la formulation MIP sont ajustées suivant la méthode proposée par Dewez (2004) lorsque les tarifs sont positifs et fixés à 500 lorsque les tarifs sont libres. Les temps de calcul de l'ensemble des méthodes exactes sont bornés à 12 heures (43200 secondes).

Les tests ont été effectués sur les familles d'instances présentée dans l'annexe I en considérant des tarifs non bornés et des tarifs positifs. Nous avons reporté dans les tableaux de 4.3 à 4.8 le nombre de produits (#OD), le pourcentage d'arcs tarifables ($\%T$) et les contraintes de borne inférieure sur les tarifs (T), le nombre d'instance résolue à l'optimalité (NOpt), l'écart (%) entre la borne supérieure Z_{sup} et la borne inférieure Z_{inf} donné par $(Z_{sup} - Z_{inf})/Z_{inf}$ (la valeur $+\infty$ indique qu'aucune borne inférieure n'a pu être découverte pour une des instances), le nombre d'itérations (#it) et le temps de calcul en seconde (CPU). Pour MIP, la colonne (NoInf) indique le pourcentage d'instances n'ayant pu trouver de solution réalisable pour le PTR. Pour MIP$^+$ et DMS_PREF_SUFF, deux colonnes sont ajoutées. La première (resp. deuxième) colonne Z_{inf}^{XXX} (Z_{sup}^{XXX}) indique le ratio entre la borne inférieure (resp. supérieure) de MIP$^+$ ou DMS_PREF_SUFF et celle obtenue par la résolution de l'algorithme xxx. Pour chaque méthode testée, la moyenne sur 5 instances est reportée.

Nous remarquons tout d'abord que les instances où la positivité des tarifs est requise sont plus faciles à résoudre que celles avec des tarifs libres. En effet, la diminution du nombre de compensation entre les tarifs implique une réduction du nombre de solution réalisables. Il en résulte une augmentation du nombre d'instances résolues à l'optimalité et une amélioration de la qualité des solutions fournit par les méthodes exactes lorsque la positivité des tarifs est requise.

Nous constatons que la résolution de la formulation MIP par le logiciel CPLEX est améliorée grâce à la procédure ajoutée (MIP$^+$). Elle permet de garantir la découverte de solutions réalisables comme pour les instances de Voronoï de 100 produits avec 20% d'arcs tarifables et améliore le revenu du meneur des solutions réalisables fournies par CPLEX. De plus, elle semble aider le logiciel CPLEX dans le choix des branchements car une amélioration de la qualité des bornes supérieures sur le revenu du meneur est observée malgré une réduction du nombre de noeuds visités. Notons que ces différences sont beaucoup plus marquées lorsque les tarifs sont libres. Dans la suite, nous ne considérons que la résolution de la formulation MIP par le logiciel CPLEX aidé de notre procédure améliorant la qualité des bornes inférieures (MIP$^+$).

Les performances de la résolution de la formulation MIP par le logiciel CPLEX et notre méthode exacte DMS_PREF_SUFF se distinguent suivant les contraintes imposées aux tarifs. Lorsque des tarifs positifs sont exigés, la résolution de la formulation MIP est plus efficace car elle fournit pour l'ensemble des instances des solutions réalisables procurant un meilleur revenu pour le meneur et des bornes supérieures de meilleures qualités. Par contre, lorsque les tarifs sont non contraints, notre méthode exacte permet de trouver des solutions réalisables et des bornes supérieures sur le revenu du meneur de meilleures qualités. Notons que ceci n'est plus vrai pour les instances de grandes tailles avec une structure de Voronoï de plus de 100 produits et dont le pourcentage d'arcs tarifables dépasse 15%.

L'évolution des bornes supérieures et inférieures sur le revenu du meneur dans les méthodes exactes en fonction des temps de calcul déterminé sont présentées à la figure 4.10 et 4.11. Notons que la courbe inf (resp. sup) fait référence à l'évolution de la borne inférieure (resp. supérieure) sur le revenu du meneur.

Remarquons d'abord que les courbes associées aux résultats de notre méthode

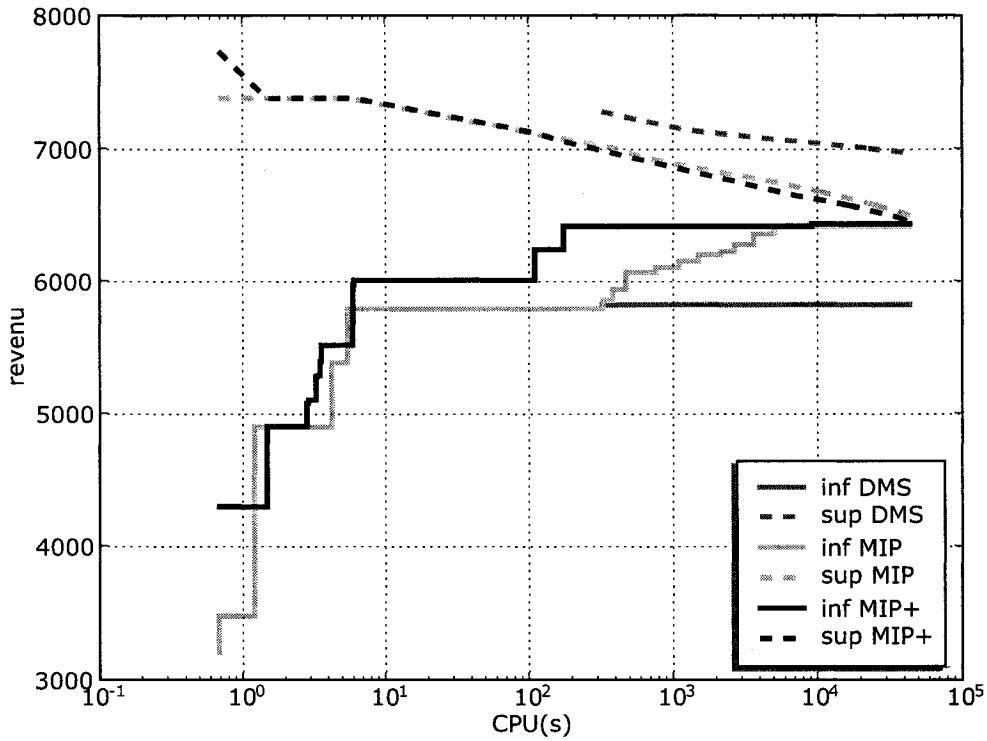


Figure 4.10 Évolution des bornes supérieures et inférieures sur le revenu du meneur en fonction des temps de calcul déterminé des méthodes exactes sur une instance de Brotcorne *et al.* de 30 produits avec 15% d'arcs tarifables lorsque des tarifs positifs sont requis

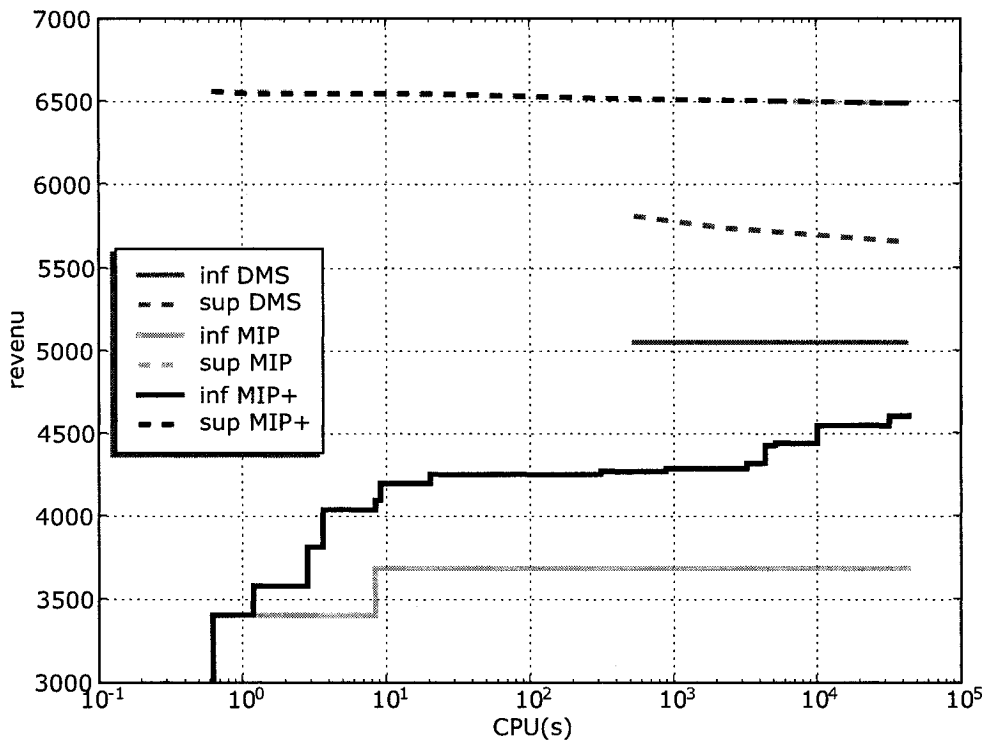


Figure 4.11 Évolution des bornes supérieures et inférieures sur le revenu du meneur en fonction des temps de calcul déterminé des méthodes exactes sur une instance de Brotcorne *et al.* de 30 produits avec 15% d'arcs tarifables lorsque les tarifs sont non contraints

commencent tardivement à cause du temps de calcul des bornes supérieures sur le revenu généré par les suffixes. D'autre part, nous constatons que la valeur de la borne inférieure de DMS_PREF_SUFF n'évolue pas en fonction des temps de calcul car les améliorations ont lieu pendant la phase des suffixes. Enfin, les courbes associées à la résolution de la formulation MIP par le logiciel CPLEX montrent clairement l'impact de la procédure sur la qualité des solutions réalisables.

4.8 Conclusions

Dans ce chapitre, nous avons proposé une méthode exacte basée sur le concept de K-chemins proposée par Didi et al. (1999). La méthode permet de générer des solutions du problème du suiveur, dénotées K-chemins, à partir des chemins obtenus par la résolution d'un problème de k-plus courts chemins. Cette génération permet une exploration suivant un ordre décroissant des valeurs d'une borne supérieure sur le revenu engendré par des K-chemins. Ce revenu est calculé en résolvant un problème d'optimisation inverse.

Nous avons proposé plusieurs améliorations à la méthode. Tout d'abord, la génération des k-plus courts chemins a été améliorée afin de réduire le nombre de K-chemins non réalisables. Ensuite, nous avons introduit une nouvelle structure de données afin d'éliminer les redondances lors de la génération des K-chemins. Elle est ensuite exploitée pour le calcul de nouvelles bornes supérieures sur le revenu d'un K-chemin. Finalement, une heuristique gloutonne a été développée afin d'améliorer la qualité de la borne inférieure dans notre méthode exacte.

Les performances de la méthode exacte proposée dans cette thèse ont été comparées à celles de la résolution de la formulation MIP par un solveur commercial auquel nous avons ajouté une procédure améliorant la qualité des bornes inférieures. Les

résultats numériques mettent en évidence l'efficacité de notre méthode sur les instances où aucune contrainte sur les tarifs n'est imposée. Par contre, elle s'avère moins efficace sur les instances où des tarifs positifs sont requis que la résolution de formulation MIP par un logiciel commercial.

Tableau 4.3 Comparaisons des résultats obtenus par les méthodes exactes DMS_PREF_SUFF, MIP et MIP$^+$ sur les instances de Brotcorne *et al.* lorsque les tarifs sont positifs

			MIP					MIP+						DMS_PREF_SUFF					
#OD	%T	Borne	NOpt	%	#it	CPU	NoInf	NOpt	%	#it	CPU	$\%Z_{inf}^{MIP}$	$\%Z_{sup}^{MIP}$	NOpt	%	#it	CPU	$\%Z_{inf}^{MIP+}$	$\%Z_{sup}^{MIP+}$
10	5	≥ 0	5	0.00	18.88	0.74	0.00	5	0.00	17.67	0.75	100.00	100.00	5	0.00	105.74	0.28	100.00	100.00
10	10	≥ 0	5	0.00	65.11	3.57	0.00	5	0.00	94.69	3.70	100.00	100.00	5	0.00	4180.85	8.76	100.00	100.00
10	15	≥ 0	5	0.00	2233.87	30.72	0.00	5	0.00	1870.58	30.98	100.00	100.00	5	0.00	27147.30	83.00	100.00	100.00
10	20	≥ 0	5	0.00	11266.85	182.88	0.00	5	0.00	9677.69	170.67	100.00	100.00	5	0.00	197884.69	740.78	100.00	100.00
20	5	≥ 0	5	0.00	85.79	4.73	0.00	5	0.00	53.56	4.66	100.00	100.00	5	0.00	4462.96	9.61	100.00	100.00
20	10	≥ 0	5	0.00	9054.87	273.45	0.00	5	0.00	12318.03	311.92	100.00	100.00	5	0.00	2132983.38	11998.98	100.00	100.00
20	15	≥ 0	3	1.24	822305.56	21111.27	0.00	4	0.91	602916.67	16689.36	100.01	99.67	0	4.74	7125746.86	43200.00	99.03	102.88
20	20	≥ 0	1	7.14	623027.78	38396.27	0.00	1	5.79	613583.33	38975.47	101.19	99.84	0	10.04	5112115.73	43200.00	97.94	101.44
30	5	≥ 0	5	0.00	326.95	17.82	0.00	5	0.00	295.00	16.63	100.00	100.00	5	0.00	59464.78	128.13	100.00	100.00
30	10	≥ 0	5	0.00	20530.05	734.50	0.00	5	0.00	14052.13	620.10	100.00	100.00	0	7.18	8649500.78	43200.00	96.22	102.55
30	15	≥ 0	1	4.13	552444.44	36700.47	0.00	2	3.37	610416.67	39566.84	100.30	99.60	0	12.12	4509300.66	43200.00	96.02	104.39
30	20	≥ 0	0	16.09	342083.33	43482.34	0.00	0	13.38	307361.11	43558.65	102.04	99.57	0	13.06	3017038.85	43200.00	99.48	98.87
40	5	≥ 0	5	0.00	1201.99	86.71	0.00	5	0.00	345.59	74.68	100.00	100.00	5	0.00	327573.67	1019.39	100.00	100.00
40	10	≥ 0	2	5.51	369555.56	37583.43	0.00	1	4.65	346722.22	37079.00	100.48	99.95	0	16.71	5882197.37	43200.00	95.04	105.06
40	15	≥ 0	0	26.20	234638.89	43435.48	0.00	0	17.53	182277.78	43492.37	107.01	100.35	0	20.02	2855459.19	43200.00	98.66	100.58
40	20	≥ 0	0	44.86	158277.78	43392.82	0.00	0	23.43	116583.33	43440.05	117.49	99.75	0	23.86	1983919.69	43200.00	98.75	98.67

Tableau 4.4 Comparaisons des résultats obtenus par les méthodes exactes DMS_PREF_SUFF, MIP et MIP$^+$ sur les instances de Brotcorne *et al.* lorsque les tarifs sont non bornés

			MIP					MIP+						DMS_PREF_SUFF					
#OD	%T	Borne	NOpt	%	#it	CPU	NoInf	NOpt	%	#it	CPU	$\%Z_{inf}^{MIP}$	$\%Z_{sup}^{MIP}$	NOpt	%	#it	CPU	$\%Z_{inf}^{MIP+}$	$\%Z_{sup}^{MIP+}$
10	5	Libre	5	0.00	16480.09	51.05	0.00	5	0.00	30543.92	131.30	100.00	100.00	5	0.00	346.61	1.74	100.00	100.00
10	10	Libre	1	9.41	14582222.22	44713.40	0.00	2	4.80	8493888.89	42538.24	102.15	99.94	5	0.00	57753.76	211.46	100.33	95.46
10	15	Libre	0	10.07	13228250.00	45243.89	0.00	0	10.00	7145444.44	45620.50	100.78	100.76	5	0.00	480524.02	2117.46	100.41	91.34
10	20	Libre	0	14.92	11439972.22	44953.81	0.00	0	9.95	6056444.44	45564.17	103.92	99.76	5	0.00	1076958.53	6780.29	100.76	91.69
20	5	Libre	2	18.78	3832777.78	27088.63	0.00	4	2.54	2510388.89	23959.65	109.03	96.57	5	0.00	12940.92	41.77	100.42	97.57
20	10	Libre	0	53.49	6147805.56	45116.32	0.00	0	30.87	3850194.44	45378.86	113.97	98.63	1	1.03	6586320.56	39414.43	104.41	80.95
20	15	Libre	0	46.45	5397916.67	44920.06	0.00	0	31.28	3031027.78	45231.15	108.90	99.66	0	11.03	4866872.74	43200.00	102.25	86.62
20	20	Libre	0	42.97	4455666.67	44837.98	0.00	0	33.22	2499388.89	45136.12	107.24	99.94	0	13.48	3334660.38	43200.00	105.05	89.32
30	5	Libre	0	37.92	4277305.56	44763.73	0.00	0	16.57	3266583.33	45166.15	114.14	98.13	5	0.00	345027.20	935.33	101.46	88.98
30	10	Libre	0	67.96	3015583.33	44555.96	0.00	0	38.58	2232250.00	44956.41	119.34	99.31	0	9.85	5200749.67	43200.00	104.33	82.71
30	15	Libre	0	88.45	2704250.00	44625.72	0.00	0	40.13	1631944.44	44755.51	133.43	99.68	0	25.10	2384113.74	43200.00	97.26	87.27
30	20	Libre	0	78.69	1960222.22	44343.61	0.00	0	40.08	1453944.44	44832.22	126.15	99.89	0	20.25	1115621.37	43200.58	106.82	91.70
40	5	Libre	0	97.57	2165638.89	44474.71	0.00	0	31.64	1885555.56	44923.53	139.70	97.07	4	0.76	3373667.09	15714.19	101.81	78.23
40	10	Libre	0	99.04	1842361.11	44335.99	0.00	0	49.79	1377027.78	44987.47	131.60	99.44	0	30.58	3072971.89	43200.00	96.93	84.22
40	15	Libre	0	101.22	1577444.44	44308.47	0.00	0	58.99	956611.11	44563.13	125.86	99.90	0	25.31	1814703.23	43200.00	111.90	87.58
40	20	Libre	0	102.62	1081722.22	44059.49	0.00	0	52.86	756361.11	44384.04	130.39	99.94	0	32.47	1058196.89	43200.91	105.66	90.98

Tableau 4.5 Comparaisons des résultats obtenus par les méthodes exactes DMS_PREF_SUFF, MIP et MIP$^+$ sur les instances avec une structure de Didi

			MIP					MIP+						DMS_PREF_SUFF					
#OD	%T	Borne	NOpt	%	#it	CPU	NoInf	NOpt	%	#it	CPU	$\%Z_{inf}^{MIP}$	$\%Z_{sup}^{MIP}$	NOpt	%	#it	CPU	$\%Z_{inf}^{MIP+}$	$\%Z_{sup}^{MIP+}$
10	5	≥ 0	5	0.00	0.00	0.04	0.00	5	0.00	0.00	0.04	100.00	100.00	5	0.00	2.33	0.04	100.00	100.00
10	10	≥ 0	5	0.00	0.00	0.06	0.00	5	0.00	0.00	0.06	100.00	100.00	5	0.00	4.06	0.11	100.00	100.00
10	15	≥ 0	5	0.00	0.43	0.13	0.00	5	0.00	0.43	0.15	100.00	100.00	5	0.00	3.00	0.18	100.00	100.00
10	20	≥ 0	5	0.00	1.40	0.22	0.00	5	0.00	1.40	0.24	100.00	100.00	5	0.00	67.78	0.79	100.00	100.00
50	5	≥ 0	5	0.00	0.00	0.96	0.00	5	0.00	0.00	1.00	100.00	100.00	5	0.00	43.49	2.55	100.00	100.00
50	10	≥ 0	5	0.00	13.09	4.20	0.00	5	0.00	13.59	4.71	100.00	100.00	5	0.00	20449.91	316.04	100.00	100.00
50	15	≥ 0	5	0.00	108.94	16.36	0.00	5	0.00	95.97	16.96	100.00	100.00	4	1.05	1253328.64	11840.87	99.36	100.44
50	20	≥ 0	5	0.00	1475.83	128.42	0.00	5	0.00	1697.78	184.52	100.00	100.00	1	7.71	3351162.02	34307.14	97.29	105.56
100	5	≥ 0	5	0.00	8.42	4.85	0.00	5	0.00	8.42	4.97	100.00	100.00	5	0.00	16679.52	295.89	100.00	100.00
100	10	≥ 0	5	0.00	351.43	52.32	0.00	5	0.00	395.18	61.80	100.00	100.00	1	8.95	2696259.11	34995.98	95.40	103.75
100	15	≥ 0	5	0.00	3543.89	515.39	0.00	5	0.00	3812.73	597.15	100.00	100.00	0	18.36	2349158.38	43200.00	94.56	112.12
100	20	≥ 0	2	3.82	94711.39	27683.98	0.00	2	4.02	84178.89	28239.38	99.95	100.09	0	30.37	1510957.52	43200.00	94.05	116.59
10	5	Libre	5	0.00	0.00	0.10	0.00	5	0.00	0.00	0.10	100.00	100.00	5	0.00	2.00	7.83	100.00	100.00
10	10	Libre	5	0.00	11.85	0.39	0.00	5	0.00	29.19	0.48	100.00	100.00	5	0.00	5.61	1.90	100.00	100.00
10	15	Libre	5	0.00	377.28	2.97	0.00	5	0.00	161.72	2.14	100.00	100.00	5	0.00	2.97	2.16	100.00	100.00
10	20	Libre	5	0.00	886.17	7.97	0.00	5	0.00	936.67	10.20	100.00	100.00	5	0.00	175.13	5.78	100.00	100.00
50	5	Libre	5	0.00	1153.61	24.73	0.00	5	0.00	1428.33	39.72	100.00	100.00	5	0.00	91.03	120.32	100.00	100.00
50	10	Libre	4	15.27	736697.22	16104.50	0.00	4	1.93	461058.89	13431.37	108.42	99.27	4	0.04	280267.24	7861.66	100.07	97.91
50	15	Libre	0	49.46	1358525.56	44406.58	0.00	0	18.61	1125428.61	44857.57	122.18	98.14	1	3.59	888458.20	34597.63	104.06	90.87
50	20	Libre	0	62.03	1101760.28	44163.28	0.00	0	25.49	729874.44	44448.24	126.38	98.27	0	16.22	896025.68	43200.02	100.72	93.89
100	5	Libre	4	0.41	600099.83	21498.87	0.00	4	0.89	401190.28	20684.17	100.05	100.69	5	0.00	73445.02	4069.33	100.00	99.05
100	10	Libre	0	138.92	760908.33	44313.69	0.00	0	36.24	672721.94	44738.03	171.02	98.88	0	20.59	955007.84	43200.00	96.08	85.05
100	15	Libre	0	157.44	541919.44	44086.76	0.00	0	48.38	466045.28	44531.78	170.12	99.06	0	26.78	395328.20	43200.04	102.50	87.81
100	20	Libre	0	186.20	295232.78	43735.47	0.00	0	60.45	287328.61	44163.13	176.96	99.91	0	33.38	201067.76	43200.06	111.64	92.53

Tableau 4.6 Comparaisons des résultats obtenus par les méthodes exactes DMS_PREF_SUFF, MIP et MIP$^+$ sur les instances avec une structure de grille

			MIP					MIP+						DMS_PREF_SUFF					
#OD	%T	Borne	NOpt	%	#it	CPU	NoInf	NOpt	%	#it	CPU	$\%Z_{inf}^{MIP}$	$\%Z_{sup}^{MIP}$	NOpt	%	#it	CPU	$\%Z_{inf}^{MIP+}$	$\%Z_{sup}^{MIP+}$
10	5	≥ 0	5	0.00	2.69	0.21	0.00	5	0.00	2.48	0.21	100.00	100.00	5	0.00	9.03	0.29	100.00	100.00
10	10	≥ 0	5	0.00	11.04	0.50	0.00	5	0.00	11.04	0.49	100.00	100.00	5	0.00	105.44	1.14	100.00	100.00
10	15	≥ 0	5	0.00	24.38	1.03	0.00	5	0.00	24.00	0.98	100.00	100.00	5	0.00	591.50	5.71	100.00	100.00
10	20	≥ 0	5	0.00	64.15	1.71	0.00	5	0.00	70.47	1.83	100.00	100.00	5	0.00	52109.92	218.38	100.00	100.00
50	5	≥ 0	5	0.00	615.86	47.07	0.00	5	0.00	528.31	44.50	100.00	100.00	5	0.00	138316.56	842.64	100.00	100.00
50	10	≥ 0	4	0.38	179295.56	9943.39	0.00	5	0.00	93495.28	5918.32	100.13	99.73	0	6.18	3617139.50	43200.00	97.34	103.49
50	15	≥ 0	2	4.43	228807.22	26938.46	0.00	2	3.21	199595.56	26960.66	100.58	99.49	0	11.14	2118237.14	43200.02	95.57	102.91
50	20	≥ 0	0	8.87	340494.17	43585.62	0.00	0	7.65	278463.06	43663.27	101.30	100.20	0	23.06	1180668.53	43200.02	90.23	101.73
100	5	≥ 0	5	0.00	1565.28	196.98	0.00	5	0.00	1725.09	208.72	100.00	100.00	4	4.94	690493.27	10946.73	96.57	100.67
100	10	≥ 0	1	15.14	136574.17	42757.38	0.00	1	11.68	125805.56	41128.73	101.96	99.53	0	24.62	1736140.76	43200.00	94.83	105.18
100	15	≥ 0	0	28.68	97895.56	43435.24	0.00	0	18.46	77770.28	43471.72	107.57	99.20	0	29.21	922851.94	43200.02	94.55	102.83
100	20	≥ 0	0	$+\infty$	70478.89	43394.65	0.20	0	23.90	51708.06	43414.58	-	99.78	0	28.79	483701.07	43200.04	97.22	100.84
10	5	Libre	5	0.00	146.94	0.93	0.00	5	0.00	128.34	0.98	100.00	100.00	5	0.00	17.11	1.71	100.00	100.00
10	10	Libre	5	0.00	2797.78	8.58	0.00	5	0.00	2264.17	10.15	100.00	100.00	5	0.00	234.91	2.70	100.00	100.00
10	15	Libre	5	0.00	3301.70	11.85	0.00	5	0.00	1540.28	9.16	100.00	100.00	5	0.00	830.87	7.47	100.00	100.00
10	20	Libre	5	0.00	569798.60	2334.07	0.00	5	0.00	468305.78	2640.15	100.00	100.00	5	0.00	3090.15	16.47	100.00	100.00
50	5	Libre	2	12.97	1626789.72	28004.39	0.00	2	3.26	1159319.44	28248.01	105.44	96.46	4	2.58	2702511.53	16070.86	99.95	99.60
50	10	Libre	0	69.33	1976327.22	44698.57	0.00	0	25.49	1189274.44	44708.56	130.65	99.05	0	16.47	2932134.44	43200.00	96.46	89.56
50	15	Libre	0	60.70	1193109.44	44279.77	0.00	0	26.12	878470.00	44574.71	125.81	99.33	0	19.67	1036278.89	43211.59	100.89	95.80
50	20	Libre	0	66.45	1152866.39	44261.18	0.00	0	21.17	807150.56	44521.36	134.20	99.58	0	19.40	998294.22	43205.89	99.65	97.44
100	5	Libre	0	179.23	812540.00	44355.65	0.00	0	18.72	728744.72	44630.57	207.78	96.92	2	7.67	2618425.88	32292.67	96.37	87.13
100	10	Libre	0	137.94	591247.50	44067.11	0.00	0	44.04	377540.56	44189.13	158.65	99.72	0	29.00	888248.30	43200.00	101.80	90.34
100	15	Libre	0	171.45	557068.89	44093.94	0.00	0	50.81	359069.44	44243.53	181.15	99.75	0	32.40	553459.99	43200.02	104.12	91.44
100	20	Libre	0	150.00	330522.78	43823.29	0.00	0	53.97	250581.11	44064.72	160.25	99.88	0	26.54	300559.46	43200.06	113.23	93.69

Tableau 4.7 Comparaisons des résultats obtenus par les méthodes exactes DMS_PREF_SUFF, MIP et MIP$^+$ sur les instances avec une structure de Delaunay

			MIP					MIP+						DMS_PREF_SUFF					
#OD	%T	Borne	NOpt	%	#it	CPU	NoInf	NOpt	%	#it	CPU	$\%Z_{inf}^{MIP}$	$\%Z_{sup}^{MIP}$	NOpt	%	#it	CPU	$\%Z_{inf}^{MIP+}$	$\%Z_{sup}^{MIP+}$
10	5	≥ 0	5	0.00	0.73	0.29	0.00	5	0.00	0.73	0.30	100.00	100.00	5	0.00	10.58	0.37	100.00	100.00
10	10	≥ 0	5	0.00	2.47	0.37	0.00	5	0.00	2.47	0.39	100.00	100.00	5	0.00	14.77	1.01	100.00	100.00
10	15	≥ 0	5	0.00	1.24	0.36	0.00	5	0.00	0.17	0.37	100.00	100.00	5	0.00	7.83	3.55	100.00	100.00
10	20	≥ 0	5	0.00	0.43	0.44	0.00	5	0.00	0.00	0.46	100.00	100.00	5	0.00	13.88	3.59	100.00	100.00
50	5	≥ 0	5	0.00	35.82	16.95	0.00	5	0.00	35.24	18.77	100.00	100.00	5	0.00	22226.61	395.65	100.00	100.00
50	10	≥ 0	5	0.00	723.33	159.80	0.00	5	0.00	722.50	173.98	100.00	100.00	2	1.28	1875072.73	30716.11	99.91	101.09
50	15	≥ 0	5	0.00	3417.78	560.74	0.00	5	0.00	2525.56	513.62	100.00	100.00	1	4.03	986254.61	39482.84	99.74	103.77
50	20	≥ 0	4	0.12	103182.78	16745.87	0.00	3	0.30	101769.72	21027.24	99.98	100.13	0	4.84	661761.47	43200.00	99.53	103.96
100	5	≥ 0	5	0.00	629.86	225.18	0.00	5	0.00	796.81	269.86	100.00	100.00	4	0.36	1683506.67	27589.94	99.89	100.23
100	10	≥ 0	4	0.91	41495.56	17256.93	0.00	4	0.74	44136.11	20052.50	100.15	100.00	0	12.08	940014.39	43200.00	97.29	108.28
100	15	≥ 0	0	6.39	60453.89	43400.28	0.00	1	5.68	53682.78	42976.56	100.32	99.65	0	16.99	286387.10	43200.10	97.83	108.30
100	20	≥ 0	0	10.39	37996.67	43353.67	0.00	0	9.04	29572.22	43385.53	101.51	100.41	0	19.05	146238.46	43200.06	98.71	107.68
10	5	Libre	5	0.00	8.62	0.55	0.00	5	0.00	7.88	0.64	100.00	100.00	5	0.00	6.64	7.86	100.00	100.00
10	10	Libre	5	0.00	220.00	1.71	0.00	5	0.00	240.00	2.59	100.00	100.00	5	0.00	3.21	1.52	100.00	100.00
10	15	Libre	5	0.00	42.78	1.26	0.00	5	0.00	64.17	1.68	100.00	100.00	5	0.00	2.21	2.34	100.00	100.00
10	20	Libre	5	0.00	180.44	2.88	0.00	5	0.00	110.11	2.93	100.00	100.00	5	0.00	61.96	5.61	100.00	100.00
50	5	Libre	4	4.70	643716.39	22195.97	0.00	3	3.06	509259.72	25406.60	101.92	100.48	5	0.00	446008.52	4934.84	100.22	96.96
50	10	Libre	0	16.82	1177159.44	44563.23	0.00	0	11.91	752776.67	44817.86	104.17	100.00	0	5.51	1838456.01	43200.00	100.80	95.06
50	15	Libre	0	20.14	983843.33	44506.62	0.00	0	12.34	637329.72	44821.92	106.69	100.01	0	5.95	1215454.62	43200.97	101.96	96.35
50	20	Libre	0	21.80	649203.06	44136.64	0.00	0	15.31	431165.56	44460.08	105.02	99.99	0	6.09	947519.55	43200.00	105.23	96.93
100	5	Libre	0	59.93	322753.33	43771.99	0.00	0	17.87	272629.72	44198.77	132.48	98.36	0	10.87	1657471.03	43200.00	98.41	92.13
100	10	Libre	0	78.95	292340.28	43917.83	0.00	0	26.12	255546.39	44384.56	137.24	99.76	0	15.83	501145.10	43200.00	103.50	95.10
100	15	Libre	0	81.84	232395.56	43815.68	0.00	0	36.88	219883.61	44353.77	132.25	99.99	0	19.17	288167.36	43200.02	110.39	96.11
100	20	Libre	0	64.40	209837.78	43792.09	0.00	0	37.46	173123.06	44226.64	119.14	99.91	0	20.76	167939.59	43200.08	111.33	97.81

Tableau 4.8 Comparaisons des résultats obtenus par les méthodes exactes DMS_PREF_SUFF, MIP et MIP$^+$ sur les instances avec une structure de Voronoï

			MIP					MIP+						DMS_PREF_SUFF					
#OD	%T	Borne	NOpt	%	#it	CPU	NoInf	NOpt	%	#it	CPU	$\%Z_{inf}^{MIP}$	$\%Z_{sup}^{MIP}$	NOpt	%	#it	CPU	$\%Z_{inf}^{MIP+}$	$\%Z_{sup}^{MIP+}$
10	5	≥ 0	5	0.00	0.00	0.66	0.00	5	0.00	0.00	0.67	100.00	100.00	5	0.00	107.20	0.73	100.00	100.00
10	10	≥ 0	5	0.00	116.19	2.16	0.00	5	0.00	80.29	2.11	100.00	100.00	5	0.00	255.96	2.43	100.00	100.00
10	15	≥ 0	5	0.00	144.25	3.31	0.00	5	0.00	112.73	3.08	100.00	100.00	5	0.00	546.67	4.14	100.00	100.00
10	20	≥ 0	5	0.00	1004.66	12.63	0.00	5	0.00	715.31	11.10	100.00	100.00	5	0.00	1084.05	11.68	100.00	100.00
50	5	≥ 0	5	0.00	584.56	60.51	0.00	5	0.00	472.50	59.16	100.00	100.00	4	0.25	992957.29	12663.17	99.83	100.05
50	10	≥ 0	1	5.25	295587.78	41628.77	0.00	1	5.33	233234.17	38679.99	99.86	100.06	1	16.82	2780659.07	41391.48	96.97	107.49
50	15	≥ 0	0	19.44	153871.94	43443.20	0.00	0	10.47	137925.28	43523.17	105.30	96.71	0	30.42	1920908.21	43200.00	89.30	104.90
50	20	≥ 0	0	$+\infty$	99590.00	43391.60	0.20	0	14.81	81613.06	43444.53	-	98.67	0	42.78	1098592.21	43200.02	81.80	102.22
100	5	≥ 0	4	0.40	118182.50	23197.97	0.00	3	0.43	106395.83	23615.18	100.00	100.05	2	4.69	1882737.86	33799.64	97.13	101.03
100	10	≥ 0	0	$+\infty$	77880.00	43415.58	0.20	0	19.98	50359.72	43475.47	-	100.07	0	36.90	1166613.37	43200.00	91.86	104.45
100	15	≥ 0	0	$+\infty$	38659.72	43352.35	0.60	0	31.11	25456.67	43380.90	-	99.73	0	58.70	541545.93	43200.00	85.04	102.67
100	20	≥ 0	0	$+\infty$	26698.06	43330.29	1.00	0	30.34	18755.56	43363.03	-	99.88	0	65.76	296746.56	43200.02	81.39	101.11
10	5	Libre	5	0.00	835.55	3.65	0.00	5	0.00	798.59	4.89	100.00	100.00	5	0.00	1310.67	2.77	100.00	100.00
10	10	Libre	5	0.00	185113.81	828.09	0.00	5	0.00	48844.92	307.06	100.00	100.00	5	0.00	7169.05	20.56	100.00	100.00
10	15	Libre	5	0.00	734005.58	5049.99	0.00	5	0.00	289325.83	2641.92	100.00	100.00	5	0.00	3277.03	12.98	100.00	100.00
10	20	Libre	3	4.41	2234431.94	19054.82	0.00	3	2.13	1545974.44	20010.22	102.51	105.01	5	0.00	7523.13	33.00	100.49	98.10
50	5	Libre	0	57.10	717676.11	43966.94	0.00	0	13.12	642358.89	44208.95	131.14	96.88	3	1.07	3653036.29	22646.06	101.67	90.10
50	10	Libre	0	100.15	528086.94	43795.62	0.00	0	36.40	364675.00	43910.83	144.87	99.58	0	23.01	2519521.72	43200.00	102.35	91.86
50	15	Libre	0	134.40	319799.17	43583.17	0.00	0	25.92	266635.00	43845.43	174.04	99.92	0	41.76	1382847.71	43200.00	89.23	98.50
50	20	Libre	0	123.55	222523.33	43511.97	0.00	0	21.23	188732.50	43731.43	176.79	99.89	0	52.86	941072.68	43200.00	85.62	105.59
100	5	Libre	0	320.47	170250.28	43557.51	0.00	0	44.23	157401.67	43767.95	270.14	98.38	0	16.53	2900387.54	43200.00	99.82	80.62
100	10	Libre	0	239.32	147543.89	43510.83	0.00	0	66.31	118115.00	43681.57	188.20	99.75	0	40.58	771199.92	43200.02	107.53	91.12
100	15	Libre	0	194.89	90401.67	43441.28	0.00	0	59.83	80301.67	43591.38	183.81	99.92	0	60.87	419706.82	43200.04	99.39	100.49
100	20	Libre	0	$+\infty$	63156.11	43419.99	0.20	0	38.01	55606.11	43534.27	-	99.91	0	74.33	176218.53	43200.08	85.63	107.60

CHAPITRE 5

MÉTHODE DE RECHERCHE LOCALE BASÉE SUR LES SOLUTIONS DU PROBLÈME DU SUIVEUR

Dans ce chapitre, nous présentons une méthode de recherche locale (Brotcorne et al. (2004a,b)) de type glouton qui sera insérée par la suite dans une métaheuristique de type tabou. Après avoir présenté son principe, nous détaillons la structure de voisinage et la stratégie de mise à jour de la solution courante. Puis, nous analysons les résultats numériques obtenus par cette méthode et nous concluons ce chapitre.

5.1 Principe de la méthode de recherche locale pour le PTR

La méthode de recherche locale proposée, nous permet de résoudre le PTR en explorant un voisinage de solutions qui sont des optima locaux. Ceux-ci se caractérisent facilement à partir des solutions du problème du suiveur et du problème d'optimisation inverse POI comme l'illustre l'exemple suivant.

Dans cet exemple, le meneur a sous sa conduite un seul arc tarifable pour lequel il doit déterminer un tarif T et le suiveur doit acheminer un unique produit. La fonction objectif du meneur en fonction du seul tarif T est représentée à la figure 5.1 (cf. section 2.2 pour plus de détails sur cette fonction). Chaque segment gris foncé représente le revenu d'une solution du suiveur de tarif T et le point gris clair, considérant cette solution, représente le revenu optimal obtenu en résolvant le POI. Cette figure illustre le fait que chaque point gris clair est un optimum local du PTR et qu'il est associé à une solution du problème du suiveur.

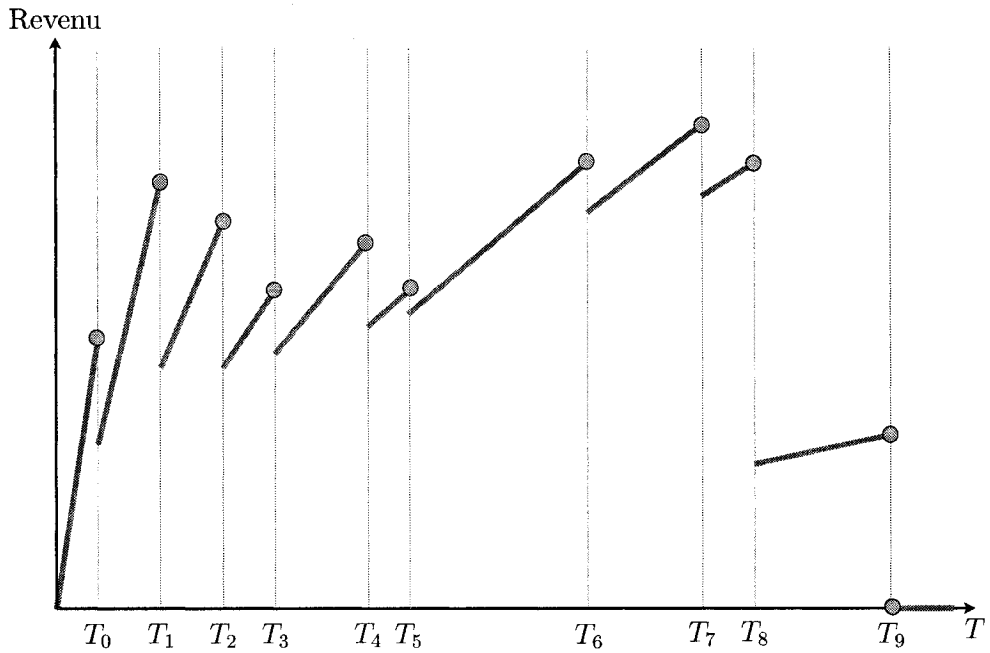


Figure 5.1 Exemple de fonction objectif du meneur en fonction du tarif T

La méthode de recherche locale proposée repose sur une exploration des solutions du problème du suiveur évaluées en terme de revenu du meneur par la résolution d'un POI. L'algorithme est décrit plus en détails dans ce qui suit :

Méthode de recherche locale pour PTR

PAS 0 *Initialisation*

- Soit S^0 une solution du problème du suiveur et (Z^0, T^0) le revenu et le plan tarifaire optimal obtenus en résolvant le POI avec S^0.
- Initialiser le compteur d'itération $l = 0$.

PAS 1 *Exploration du voisinage*

- $(Z^*, T^*) = (0, +\infty)$
- Soit $L(S^l)$ l'ensemble des solutions voisines de S^l.
- Pour chaque $S \in L(S^l)$,
  - Calculer (Z, T) en résolvant le POI avec S.
  - **Si** $Z > Z^*$ **alors** $S^* = S, Z^* = Z, T^* = T$.

- aller au pas 2.

PAS 2 *Critère d'arrêt*

- **Si** $Z^* > Z^l$ **alors**
  - $l = l + 1$.
  - $S^l = S^*, Z^l = Z^*, T^l = T^*$.
  - Aller au pas 1.

  sinon Arrêt.

5.2 Voisinage d'une solution du problème du suiveur

Dans cette section, nous présentons la structure de voisinage utilisée par la méthode de recherche locale. Pour des raisons de simplicité, nous considérons d'abord le cas mono produit où une solution du problème du suiveur est un chemin reliant l'origine o à la destination d du produit. Le voisinage est un ensemble de chemins entre l'origine o et la destination d du produit dont la taille peut être exponentielle. Afin de limiter cette taille, les nouveaux chemins sont construits à partir d'un arbre de recouvrement $\mathcal{T}_o$ de racine égale à l'origine du produit considéré. Plus précisément, un chemin voisin est obtenu en ajoutant un arc $(i, j) \in \mathcal{A}$ tel que :

- $(i, j) \notin \mathcal{T}_o$,
- il existe un chemin $p_{\mathcal{T}_o}(o, i)$ dans $\mathcal{T}_o$ de o vers i,
- il existe un chemin $p_{\mathcal{T}_o}(j, d)$ dans $\mathcal{T}_o$ de j vers d.

Le chemin voisin est donné par la concaténation de $p_{\mathcal{T}_o}(o, i)$, (i, j) et $p_{\mathcal{T}_o}(j, d)$.

Ce voisinage est illustré sur le réseau de la figure 5.2(a) comprenant un produit d'origine 1 et de destination 5 et ayant une demande de 1. Les arcs $(1, 3)$, $(3, 5)$ et $(2, 5)$ sont tarifables. L'arbre de plus courts chemins, obtenu lorsque les tarifs du meneur sont fixés à zéro, est représenté à la figure 5.2(b). La solution courante du problème du suiveur est définie par le chemin $p = 1 - 3 - 2 - 5$. L'ajout de

l'arc $(1,4)$ dans l'arbre ne permet pas de construire un nouveau chemin de 1 à 5. L'ajout de l'arc $(5,1)$ implique la création d'un chemin de 1 à 5 non élémentaire. Seul l'ajout des arcs $(4,5)$, $(1,2)$, $(5,1)$ permet de construire les nouveaux chemins de 1 à 5. Finalement, les chemins voisins sont $1-3-4-5$, $1-3-5$ et $1-2-5$.

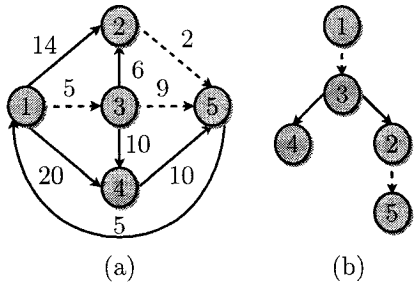


Figure 5.2 Exemple de voisinage d'une base du problème du suiveur

La généralisation de ce voisinage au cas multi-produits s'effectue en considérant un arbre par produit ($\mathcal{T}_{o(k)}$). Un voisin est obtenu en ajoutant un $(i,j) \in \mathcal{A}$ dans l'arbre de recouvrement $\mathcal{T}_{o(k)}$ d'un produit $k \in \mathcal{K}$ tel que

- $(i,j) \notin \mathcal{T}_{o(k)}$,
- il existe un chemin $p_{\mathcal{T}_{o(k)}}(o(k),i)$ dans $\mathcal{T}_{o(k)}$ de $o(k)$ vers i,
- il existe un chemin $p_{\mathcal{T}_{o(k)}}(j,d(k))$ dans $\mathcal{T}_{o(k)}$ de j vers $d(k)$.

Le chemin du produit k est donné par la concaténation de $p_{\mathcal{T}_o}(o,i)$, (i,j) et $p_{\mathcal{T}_o}(j,d)$ que nous noterons p_k. De plus, pour ne pas visiter de solutions sous optimales, le chemin $p_{k'}$ des produits $k' \neq k$ sont modifiés si p_k et $p_{k'}$ relient, tous les deux, les noeuds u à v avec des sous chemins différents. Si une telle situation se présente, les produits k et k' sont en "interactions" et le sous chemin de u à v de $p_{k'}$ est remplacé par celui de p_k. L'exemple suivant illustre cette situation.

Considérons l'exemple de la figure 5.3 où le réseau comprend deux produits $(1,6)$ et $(2,6)$. Les chemins courants sont $1-3-5-6$ (resp. $2-3-5-6$) pour le produit $(1,6)$ (resp. $(2,6)$). L'ajout de l'arc $(4,6)$ dans l'arbre du produit $(2,6)$ crée le nouveau chemin $2-3-4-6$. L'utilisation de ce chemin est en contradiction avec l'utilisation

du chemin du produit $(1, 6)$ puisque les sous-chemins reliant les noeuds 3 à 6 sont différents pour les produits. Comme prouvé par Dewez (2004), il suit qu'une telle solution est sous-optimale. Nous proposons donc de remplacer les sous-chemins des produits en contradiction par celui utilisé par le nouveau chemin. Ainsi, le chemin du noeud 3 au noeud 6 du produit $(1, 6)$ est remplacé par le sous-chemin $3 - 4 - 5$ emprunté par le produit $(2, 6)$.

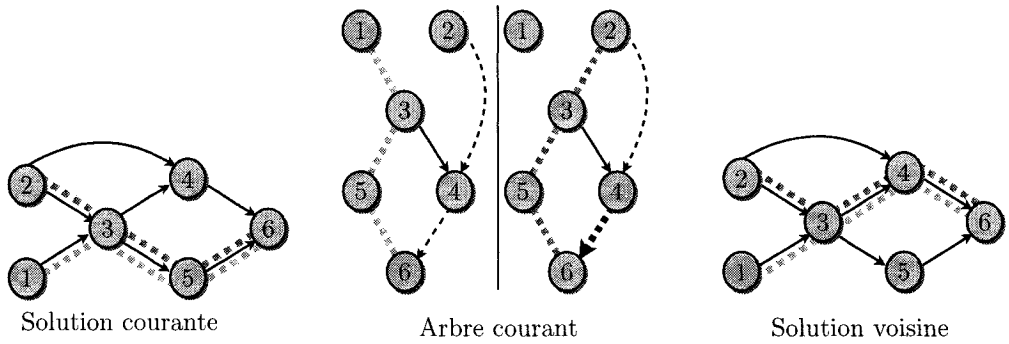


Figure 5.3 Exemple d'interaction entre les produits

5.3 Mise à jour de la solution courante

Une fois qu'une solution améliorante est trouvée, les arbres de recouvrement doivent être mis à jour. Nous proposons deux approches possibles que nous avons testées à la section 5.5.1.2.

Une première consiste à chaque itération à calculer l'arbre de plus courts chemins correspondant à chaque produit obtenu en fixant les tarifs optimaux associés à la solution courante. Même si cette approche permet de maintenir un arbre de recouvrement adapté aux tarifs du meneur, elle requiert l'utilisation d'un algorithme de plus court chemin pour chaque produit.

Une deuxième approche consiste à effectuer des pivots de l'algorithme du simplexe réseaux. Plus précisément, l'arc (i, j) ayant permis de construire le nouveau chemin

d'un produit est ajouté à l'arbre et l'ancien arc permettant d'arriver au noeud j est supprimé. Par exemple, si la solution améliorante est le chemin $1-3-4-5$ obtenu avec l'arc $(4,5)$, l'arbre est mis à jour en ajoutant l'arc $(4,5)$ et en supprimant l'arc $(2,5)$ de l'arbre de recouvrement. Cette approche permet de mettre rapidement à jour l'arbre de recouvrement mais ne tient pas compte des tarifs du meneur.

5.4 Réduction du voisinage

La méthode de recherche locale, présentée à la section 5.1, nécessite l'évaluation du voisinage de chaque produit à chaque itération. Cette évaluation étant coûteuse en temps, nous proposons une version où un seul produit est considéré à chaque itération.

Plus précisément, les produits ayant été évalués après la découverte de la meilleure solution sont marqués. Ainsi, lorsque tous les produits sont marqués, nous savons qu'il n'existe plus aucune solution améliorante dans aucun voisinage et la méthode de recherche locale peut être stoppée. Par contre, lorsqu'une solution améliorante est découverte, elle remplace la solution courante, tous les produits sont considérés comme "non marqués" et nous reprenons la recherche d'une telle solution améliorante. L'algorithme de cette méthode de recherche locale est décrit dans ce qui suit.

Méthode de recherche locale basée sur l'évaluant du voisinage d'un produit à chaque itération

PAS 0 *Initialisation*

- Soient S^0 une solution du problème du suiveur et (Z^0, T^0) : le revenu et le plan tarifaire optimal obtenus en résolvant le POI avec S^0.
- Initialiser le compteur d'itération $l = 1$.

- Initialiser tous les produits comme "non marqués".

PAS 1 *Exploration des produits*

- **Si** tous les produits sont marqués **alors** arrêt.

  Sinon

  - *Choisir* un produit $k^l \in \mathcal{K}$ non marqué.
  - Soit L^{k^l} l'ensemble des solutions voisines de S^l obtenues à partir du produit k.
  - $(Z^*, T^*) = (0, +\infty)$,
  - Pour chaque $S \in L^{k^l}$.
    - Soit (Z, T) le résultat du problème d'optimisation inverse appliqué à S.
    - **Si** $Z > Z^*$ **alors** $S^* = S, Z^* = Z, T^* = T$.

PAS 2 *Mise à jour de la meilleure solution*

- **Si** $Z^* > Z^l$ **alors**
  - $S^l = S^*, Z^l = Z^*, T^l = T^*$.
  - Mise à jour des arbres de recouvrement de chaque produit.
  - Réinitialiser tous les produits à "non marqués".
  - $l = l + 1$.

  Sinon

  - $S^{l+1} = S^l, Z^{l+1} = Z^l, T^{l+1} = T^l$.
  - Marquer le produit k^l.
- Aller au pas 1.

Plusieurs critères de choix des produits (cf. PAS 1) sont présentés dans la section consacrée aux résultats numériques 5.5.1.3.

5.5 Résultats numériques

Les résultats numériques sont divisés en deux parties. La première partie vise à sélectionner les meilleures stratégies pour la méthode de recherche locale. La deuxième partie permet d'étudier le comportement de la méthode de recherche locale (basée sur les meilleures stratégies) sur un grand nombre d'instances.

Avant la présentation des résultats numériques, nous précisons que les détails sur l'implémentation des méthodes sont identiques à ceux présentés à la section 4.7.

Les instances utilisées sont des réseaux sous forme de grille de 60 noeuds et 208 arcs dont le nombre de produits varie entre 10, 20, 30 et 40 produits. Pour chaque type d'instances, la sélection des arcs tarifables est effectuée par la méthode introduite par Brotcorne et al. (2000). Pour plus de détails sur la génération de ces instances, nous renvoyons le lecteur à l'annexe I.

5.5.1 Tests des différentes stratégies de la méthode de recherche locale

Les premières expérimentations numériques s'intéressent à l'impact sur les temps de calcul de la méthode de génération de colonnes pour la résolution du problème d'optimisation inverse présentée à la section 3.2.2. Nous étudions ensuite l'influence sur la qualité des solutions des mises à jour des arbres de recouvrement de la solution courante proposée à la section 5.3. Finalement, les stratégies de sélections des produits pour la méthode de recherche locale à voisinage réduit présentée à la section 5.4 sont analysées.

Pour ces tests, nous nous sommes limités à des instances ayant 15% d'arcs tarifables et un nombre de produits variant entre 10, 20, 30 et 40 et nous supposons que les tarifs sont non bornés.

Tableau 5.1 Efficacité de la méthode de décomposition pour la résolution du problème d'optimisation inverse

Instances			RL			RLGC			
#OD	$\%T$	T	%	#it	CPU	%	#it	CPU	ratio
10	15	libre	99.69	3.40	1.72	99.69	3.40	0.39	4.41
20	15	libre	95.16	5.60	20.57	95.16	5.60	1.52	13.53
30	15	libre	93.49	10.80	127.16	93.49	10.80	5.02	25.33
40	15	libre	94.45	10.80	343.43	94.45	10.80	8.68	39.56

Les résultats sont reportés dans les tableaux 5.1 à 5.3 où sont indiqués le nombre de produits (#OD), le pourcentage d'arcs tarifables ($\%T$) et les contraintes de borne inférieure sur les tarifs (T), la qualité (%) de la solution par rapport à celle fournie par la méthode exacte, dont le temps de calcul a été borné à 2 heures et le temps de calcul (CPU) en secondes. Ces résultats sont une moyenne sur 5 instances.

5.5.1.1 Efficacité de la résolution de la méthode de génération de colonnes pour la résolution du problème d'optimisation

L'efficacité de la méthode de génération de colonnes pour la résolution du POI est comparée à celle de la méthode de recherche locale où le voisinage de chaque produit est évalué à chaque itération et où les arbres de recouvrement sont mis à jour par des pivots de l'algorithme du simplexe.

Nous avons considéré deux versions de la méthode de recherche locale : une utilisant la méthode de génération de colonnes pour résoudre le POI, notée RLGC, et l'autre ne l'utilisant pas, notée RL c'est-à-dire que nous résolvons la formulation linéaire du POI avec un logiciel commercial. Les résultats numériques sont reportés dans le tableau 5.1 auquel nous avons ajouté une colonne (ratio) représentant le ratio entre les temps de calcul de RL et ceux de RLGC.

Les temps de calcul montrent l'efficacité de la méthode de génération de colonnes. Par exemple, nous observons que RLGC est 39 fois plus rapide que RL sur les instances de 40 produits. La performance de RLGC est due en particulier à la stratégie de gestion des colonnes utilisée. En effet, nous avons remarqué que la méthode de génération de colonnes effectue très peu d'itérations pour résoudre le POI. Ceci est dû au fait que les solutions différent très peu entre deux résolutions consécutives. Dans ce cas, les colonnes nécessaires pour trouver la solution optimale sont déjà présentes et elles n'ont donc plus besoin d'être générées.

Finalement, dans la suite des expérimentations numériques, nous utilisons la méthode de décomposition pour résoudre le POI intervenant dans la méthode de recherche locale. Pour ne pas alourdir les notations, nous renommons RLGC en RL.

5.5.1.2 Stratégie de mise à jour de la solution courante

Deux stratégies de mise à jour de l'arbre de recouvrement de chaque produit de la solution courante ont été proposées à la section 5.3. Afin d'étudier l'influence de ces mises à jour, nous comparons la méthode de recherche locale, notée RL-SP, mettant à jour les arbres de recouvrement à l'aide d'un arbre de plus courts chemins et la méthode de recherche locale, notée RL-PIV, effectuant un pivot de l'algorithme du simplexe réseau. Les résultats numériques sont reportés dans le tableau 5.2.

Les résultats numériques mettent tout d'abord en évidence la supériorité de la qualité des solutions fournies par la méthode RL-SP par rapport à PL-PIV. Cette différence de qualité augmente avec le nombre de produits. Ainsi, pour les instances avec 20 produits, la différence est d'environ 3,5% alors qu'elle est d'environ 10% pour les instances de 40 produits. D'autre part, RL-SP nécessite moins d'itéra-

Tableau 5.2 Comparaisons des stratégies de mise à jour de la solution courante

Instances			RL-PIV			RL-SP		
#OD	%T	T	%	#it	CPU	%	#it	CPU
10	15	libre	99.69	3.40	0.39	99.62	2.60	0.38
20	15	libre	95.16	5.60	1.52	98.70	4.60	1.49
30	15	libre	93.49	10.80	5.02	100.21	6.80	3.12
40	15	libre	94.45	10.80	8.68	104.43	7.40	5.28

tions pour trouver ces meilleures solutions. Pour les instances avec 20 produits, la différence est d'environ 1 itération alors qu'elle est d'environ 3 itérations pour les instances de 40 produits. Cette réduction du nombre d'itérations implique une réduction des temps de calcul pour RL-SP malgré une stratégie de mises à jour plus coûteuse en temps.

Notons que les performances de RL-SP sont dues à la mise à jour des arbres de chaque produit en calculant à chaque itération l'arbre de plus courts chemins correspondant aux tarifs optimaux de la solution courante. Cependant, comme nous l'avons dit, cette mise à jour dépend des tarifs et en particulier de ceux des arcs tarifables non utilisés dans la solution du problème du suiveur. Il est donc impératif d'utiliser la formulation du POI, présentée à la section 3.3.1. Cette formulation permet d'obtenir des tarifs les plus petits possibles pour les arcs tarifables non utilisés garantissant une attractivité pour les solutions voisines pour ces arcs.

Dans la suite, la mise à jour de la solution courante est effectuée à l'aide des arbres de plus courts chemins.

5.5.1.3 Stratégie de sélection des produits pour la méthode de recherche locale évaluant le voisinage d'un produit à chaque itération

Nous testons la sélection du produit devant être évalué dans la méthode de recherche locale (RL-1OD) présentée à la section 5.4. Plusieurs stratégies de sélection ont été considérées.

La première stratégie de sélection (DEM) est basée sur la demande associée à chaque produit. Plus précisément, les produits sont visités en ordre décroissant des valeurs de demande. En effet, les chemins du produit possédant une demande élevée ont plus d'influence sur les tarifs que ceux avec une petite demande. La deuxième stratégie (SUP) est identique à la première mais l'ordre de sélection est basée sur les bornes supérieures sur le revenu du meneur dans le cas mono-produit présentée à la section 2.3. Une autre stratégie intervient lorsqu'une solution améliorante est trouvée car deux possibilités peuvent être envisagées. La première consiste à visiter les produits de manière cyclique (CYC) en utilisant une liste circulaire de produits initialisée suivant l'ordre de sélection des produits. Ainsi, à chaque fois qu'une solution améliorante est trouvée, le produit suivant celui venant d'être exploré est choisi. La deuxième (NOCYC) consiste à revisiter les produits suivant l'ordre choisi à chaque fois qu'une solution améliorante est trouvée.

Les résultats numériques sont reportés dans le tableau 5.3.

Dans un premier temps, nous nous intéressons à l'influence de l'ordonnancement des produits. Pour une stratégie de parcours des produits (CYC ou NOCYC), la qualité des solutions obtenues avec RL-1OD-SUP est meilleure que celles obtenues avec RL-1OD-DEM. Par exemple, pour les instances de 40 produits, la qualité des solutions de RL-1OD-SUP sont 1 à 2% meilleures que celles de RL-1OD-DEM. Par

Tableau 5.3 Comparaison des stratégies de sélections des produits pour le méthode de recherche locale

Instances			NOCYC					
			RL-1OD-DEM			RL-1OD-SUP		
#OD	%T	T	%	#it	CPU	%	#it	CPU
10	15	libre	99.44	17.20	0.30	99.59	21.40	0.34
20	15	libre	97.44	49.60	0.86	98.40	53.00	0.94
30	15	libre	98.82	107.40	1.89	100.05	107.80	2.09
40	15	libre	103.44	189.40	3.63	104.53	148.20	3.24

Instances			CYC					
			RL-1OD-DEM			RL-1OD-SUP		
#OD	%T	T	%	#it	CPU	%	#it	CPU
10	15	libre	99.44	14.80	0.27	99.59	18.80	0.31
20	15	libre	97.31	34.40	0.63	97.30	36.60	0.67
30	15	libre	98.81	69.60	1.20	100.05	72.20	1.30
40	15	libre	103.27	126.00	2.44	104.48	105.20	2.17

contre, les temps de calcul et le nombre d'itérations restent comparables.

Dans un deuxième temps, nous analysons l'influence du parcours des produits. Pour une stratégie d'ordonnancement des produits, la qualité des solutions est en moyenne identique pour les deux types de parcours. Cependant, une divergence des temps de calcul qui s'accentue avec l'augmentation du nombre de produits est observée. La stratégie CYC s'avère être légèrement plus rapide que la stratégie NOCYC.

Ce phénomène s'explique par l'intensification sur les "meilleurs" produits (c'est-à-dire les produits avec une plus forte demande pour RL-1OD-DEM ou avec une meilleure borne supérieure pour RL-1OD-SUP). Plus précisément, la recherche d'une solution améliorante pour un produit "moins bon" (ayant une demande ou une borne supérieure moins élevée) nécessite d'explorer le voisinage des meilleures produits ce qui implique un nombre d'itérations plus important. Ce phénomène ne

se produit pas avec la stratégie CYC car l'exploration cyclique permet de visiter chaque produit avec une même fréquence.

Dans la suite, nous considérerons la méthode de recherche RL-1OD avec les stratégies SUP et CYC.

5.5.2 Résultats numériques de la méthode de recherche locale

Nous présentons les résultats numériques de la méthode de recherche locale explorant le voisinage de chaque produit à chaque itérations, notée RL, et celle utilisant le voisinage réduit présenté à la section 5.4, notée RL-1OD. Pour cette dernière, les produits sont sélectionnés cycliquement suivant la borne supérieure mono-produit

Pour ces tests, nous avons considéré des instances dont le nombre de produits varie entre 10, 20, 30 et 40 produits et le pourcentage d'arcs tarifables varie entre 5%, 10%, 15% et 20% et en considérant des tarifs non bornés et des tarifs positifs.

Les résultats numériques présentés dans chaque ligne du tableau 5.4 sont des moyennes sur 5 instances. Dans le tableau sont reportés le nombre de produits (#OD), le pourcentage d'arcs tarifables ($\%T$) et les contraintes de borne inférieure sur les tarifs (T). Les colonnes suivantes sont consacrées aux résultats obtenus avec les méthodes exactes. Si les tarifs sont positifs (resp. libres), la résolution exacte est effectuée par la résolution de la formulation MIP par le logiciel commercial CPLEX (2006) (resp. par la méthode exacte présentée au chapitre 4(DMS)) où les temps de résolutions ont été bornés à 2 heures. Pour chaque méthode exacte, nous reportons l'écart (%) entre la borne supérieure Z_{sup} et la borne inférieure Z_{inf} donnée par $(Z_{sup} - Z_{inf})/Z_{inf}$ et le temps de calcul en secondes (CPU). Pour les méthodes de recherche locale testées, nous reportons la qualité des solutions obtenues (%) par rapport à la solution fournie par la méthode exacte, la pire qualité obtenue (%min)

parmi les 5 instances, le nombre d'itération (#it) et le temps de calcul en secondes (CPU).

Nous remarquons d'abord que les deux méthodes de recherche locale se comportent de manière similaire quelles que soient les contraintes sur les tarifs et les temps de calcul des deux méthodes sont négligeables par rapport à ceux des méthodes exactes. Cependant, RL-1OD est 2 à 3 fois plus rapide que RL étant donné le nombre restreint de solutions visitées à chaque itération.

La qualité des solutions se situe en moyenne à 6% des meilleures solutions fournies par les méthodes exactes. Notons que les solutions de moins bonne qualité se trouvent en moyenne à 16%. Ceci s'explique par le fait que les méthodes de recherche locale s'arrêtent prématurément à cause de l'absence de solutions améliorantes dans le voisinage immédiat.

5.6 Conclusions

Dans ce chapitre, nous avons proposé une méthode de recherche locale pour le PTR. Cette méthode repose sur une exploration d'optima locaux caractérisés à partir des solutions du problème du suiveur. Après avoir défini le voisinage exploitant la structure de réseau du problème du suiveur, des stratégie de gestion du voisinage et de la mise à jour de la solution courante ont été envisagées. Les expérimentations numériques ont permis de valider la structure de voisinage pour une exploration locale efficace. Dans le chapitre suivant, cette méthode de recherche locale est utilisée dans une méthode de recherche avec tabous afin de contrecarrer l'arrêt prématuré de celle-ci.

Tableau 5.4 Résultats numériques obtenus avec les méthodes de recherche locale RL et RL-1OD

			MIP		RL				RL-1OD			
#OD	%T	T	%	CPU	%min	%	#it	CPU	%min	%	#it	CPU
10	5	≥ 0	0.00	0.72	95.65	98.52	3.00	0.08	95.65	98.52	17.40	0.05
10	10	≥ 0	0.00	3.35	95.49	97.54	4.40	0.17	95.49	97.54	23.00	0.10
10	15	≥ 0	0.00	33.90	95.83	98.46	4.60	0.27	95.83	98.46	23.60	0.16
10	20	≥ 0	0.00	181.64	95.07	98.68	4.80	0.39	97.12	99.09	23.00	0.22
20	5	≥ 0	0.00	4.26	87.31	93.10	4.80	0.27	87.18	93.02	43.40	0.13
20	10	≥ 0	0.00	266.84	78.21	93.16	7.40	0.72	78.21	92.95	62.40	0.34
20	15	≥ 0	3.48	5522.79	75.68	89.80	7.60	1.01	75.68	89.59	63.20	0.49
20	20	≥ 0	11.41	7263.25	89.85	94.72	8.00	1.70	88.12	95.06	63.20	0.76
30	5	≥ 0	0.00	20.00	81.77	92.22	5.00	0.44	65.96	87.50	81.60	0.27
30	10	≥ 0	0.00	785.25	68.58	81.52	9.20	1.67	66.85	81.08	72.80	0.48
30	15	≥ 0	7.86	7269.17	69.56	82.71	10.40	2.82	69.49	83.53	96.40	0.95
30	20	≥ 0	22.72	7248.54	81.76	93.48	13.40	4.73	79.96	92.05	88.80	1.19
40	5	≥ 0	0.00	86.86	58.71	81.54	6.60	1.10	67.09	88.49	111.20	0.49
40	10	≥ 0	9.86	7257.81	78.36	87.52	9.20	2.85	78.36	87.86	147.60	1.20
40	15	≥ 0	46.47	7239.42	66.46	104.37	11.20	5.12	71.41	103.71	140.80	1.65
40	20	≥ 0	63.01	7235.11	102.41	112.20	13.00	7.95	99.91	109.95	109.20	1.79
	Moyenne				82,54	93,72		1,96	82,02	93,65		0,64

			K-chemins		RL				RL-1OD			
#OD	%T	T	%	CPU	%min	%	#it	CPU	%min	%	#it	CPU
10	5	libre	0.00	1.64	95.65	99.13	3.20	0.11	95.65	99.13	19.40	0.07
10	10	libre	0.00	202.23	94.00	97.51	4.20	0.29	94.00	97.51	28.00	0.21
10	15	libre	0.00	2041.01	97.77	99.30	4.20	0.51	97.77	99.30	24.40	0.33
10	20	libre	0.26	4669.08	96.78	98.69	5.00	0.96	93.65	98.25	27.20	0.62
20	5	libre	0.00	40.64	87.18	93.02	4.80	0.40	87.18	93.02	43.40	0.20
20	10	libre	6.68	7200.01	83.18	94.38	7.20	1.13	83.18	96.24	65.20	0.59
20	15	libre	12.95	7200.02	94.00	98.26	7.20	1.95	94.00	98.23	67.00	1.03
20	20	libre	14.62	7200.02	96.73	98.31	8.00	3.42	93.75	98.54	60.00	1.49
30	5	libre	0.00	1152.10	83.99	93.33	6.00	0.82	68.04	90.75	85.00	0.40
30	10	libre	13.12	7200.01	68.99	85.29	8.80	2.95	69.72	80.41	72.60	0.90
30	15	libre	26.07	7200.03	83.98	95.02	12.00	6.13	83.98	95.19	108.60	1.97
30	20	libre	20.58	7200.24	85.08	93.99	12.60	9.32	85.08	93.21	118.20	3.24
40	5	libre	9.82	7097.45	58.32	81.72	5.00	1.27	66.34	87.39	112.00	0.71
40	10	libre	32.48	7200.03	85.52	97.67	8.40	4.62	85.52	98.64	162.20	2.14
40	15	libre	26.49	7200.05	68.56	88.66	10.20	8.88	65.29	87.01	115.00	2.74
40	20	libre	33.14	7200.14	82.78	94.42	12.80	17.87	74.24	92.65	124.40	4.83
	Moyenne				85,16	94,29		3,47	83,59	94,09		1,34

CHAPITRE 6

MÉTHODE DE RECHERCHE AVEC TABOUS POUR LE PROBLÈME DE TARIFICATION SUR UN RÉSEAU

Dans le chapitre précédent, nous avons validé l'approche de recherche locale basée sur les solutions du problème du suiveur. Celle-ci nous permet de trouver des solutions de bonne qualité en des temps de calcul très courts mais son arrêt prématuré l'empêche de découvrir des solutions de meilleure qualité. Pour améliorer la qualité de ces solutions, nous proposons une méthode de recherche avec tabous exploitant le voisinage de la méthode de recherche locale Brotcorne et al. (2006a).

Ce chapitre se compose de la manière suivante, nous présentons tout d'abord le principe générale de la méthode de recherche avec tabous. Puis, nous adaptons cette méthode au PTR. Finalement, nous présentons les résultats numériques avant de conclure.

6.1 Méthode de recherche avec tabous

La méthode de recherche avec tabous a été proposée par Glover (1986) et indépendamment par Hansen (1986). Cette métaheuristique est une méthode de recherche locale possédant des mécanismes permettant au processus d'exploration de dégrader la solution courante. Elle a été appliquée dans de nombreux problèmes combinatoires de grande taille pour obtenir des solutions de qualité. À notre connaissance, la seule utilisation de cette méthode sur un problème mathématiques à deux niveaux a été proposée par Gendreau et al. (1996) pour la famille des problèmes linéaire-linéaire de grande taille.

La méthode de recherche avec tabous permet d'éviter l'arrêt prématuré de l'algorithme suite à l'absence de solution améliorante dans le voisinage immédiat. L'idée est donc d'accepter une dégradation de la valeur de la solution courante afin de poursuivre l'exploration de l'espace des solutions (appelé aussi espace de recherche). Pour cela, la méthode de recherche avec tabous vise à se déplacer d'une solution vers une autre en choisissant la solution qui améliore le plus ou détériore le moins la valeur de la fonction objectif de la solution courante dans un voisinage.

Cette stratégie élitiste de sélection des solutions conduit souvent à la visite répétée des mêmes solutions. Il faut donc garder une trace du cheminement passé et s'y référer pour ne pas visiter de nouveau des solutions. Ainsi, une mémoire à court terme, appelée *liste des tabous*, conserve les dernières solutions visitées. Ces solutions sont dites *taboues* c'est-à-dire qu'elles ne peuvent pas être revisitées pendant un certain nombre d'itérations.

La manière la plus simple de définir les solutions taboues est de stocker, dans une liste L, les $|L|^{ème}$ dernières solutions rencontrées. Cependant, ce stockage peut s'avérer coûteux en terme de quantité d'informations mémorisées. Ainsi, plutôt que de conserver des solutions complètes, seules les transformations permettant de passer d'une solution à une autre sont mémorisées. Plus précisément, les transformations inverses sont stockées dans la liste tabou L pour interdire les retours en arrière. Notons que ces transformations sont appelées *mouvements tabous.*

Dans certains cas, il arrive que le statut tabou empêche la découverte d'une solution intéressante. Par exemple, il existe une solution améliorant la meilleure solution trouvée jusqu'à présent mais le mouvement pour l'obtenir est tabou. Dans ce cas, le statut tabou peut être levé pour atteindre cette solution. Les conditions sous lesquelles le caractère tabou d'un mouvement n'est pas pris en compte sont appelées *critère d'aspiration.*

Pour étendre la recherche dans des régions intéressantes (*intensification*) ou éviter de rester dans une seule région de l'espace de recherche (*diversification*), une mémoire à long terme est ajoutée à la méthode de recherche avec tabous. Par exemple, la mémoire à base de fréquence attribue des pénalités à des caractéristiques des solutions plusieurs fois visitées au cours de la recherche. Cette technique simple permet de diversifier la recherche facilement. Par ailleurs, les mouvements ayant conduit à des bonnes solutions peuvent être aussi encouragés en gardant en mémoire une liste de meilleures solutions autour desquelles une recherche plus approfondie peut être effectuée.

Nous allons présenter un cadre général de la méthode de recherche avec tabous. Soit le problème d'optimisation consistant à maximiser une fonction $Z(S)$ sur un domaine $\mathcal{X}$ et soit S^* la meilleure solution obtenue jusqu'à présent et $Z^* = Z(S^*)$ sa valeur. Notons $N(S, i)$ l'ensemble des solutions admissibles à partir de S à l'itération i c'est-à-dire l'ensemble des solutions voisines non taboues et des solutions voisines taboues vérifiant le critère d'aspiration. Dans ce cas, la méthode de recherche avec tabous s'écrit de la manière suivante :

Recherche avec tabous

Pas 1 : *Initialisation*

- Soit une solution réalisable $S^0 \in \mathcal{X}$.
- Initialiser la meilleure solution : $S^* = S^0, Z^* = Z(S^*)$.
- Initialiser le compteur d'itération $i = 1$.
- Initialiser la liste des tabous $L = \emptyset$.

Pas 2 : *Exploration du voisinage*

- Sélectionner $S^i = \arg\min\{Z(S'), S' \in N(S^{i-1}, i)\}$.

Pas 3 : *Mise à jour de la meilleure solution*

- **Si** $(Z(S^i) > Z^*)$ **alors** $S^* = S^i, Z^* = Z(S^*)$.
- Mise à jour de la liste des tabous L et de la mémoire à long terme .

- $i = i + 1$.

Pas 4 : *Critère d'arrêt*

- **Si** Critère d'arrêt satisfait **alors** arrêt.

Pas 5 : *Intensification*

- **Si** Critère d'intensification satisfait **alors** intensifier la recherche en exploitant la mémoire à long terme.

Pas 6 : *Diversification*

- **Si** Critère de diversification satisfait **alors** diversifier la recherche en exploitant la mémoire à long terme.
- Aller au pas 2.

Cet algorithme est une version simplifiée de la méthode de recherche avec tabous. Cependant, elle permet de mettre en évidence les différentes phases de cette méthode. Dans la suite, nous utilisons cet algorithme pour décrire et présenter les choix de conception d'une méthode de recherche avec tabous pour le PTR.

6.2 Méthode de recherche avec tabous pour le PTR

Nous présentons la méthode de recherche avec tabous en commençant, dans la sous-section suivante, par aborder cette méthode sans mécanisme de diversification. La sous-section suivante sera consacrée à cette stratégie.

6.2.1 Mise en oeuvre de la méthode de recherche avec tabous

6.2.1.1 Voisinage

Comme nous l'avons dit précédemment, la méthode de recherche avec tabous est une méthode de recherche locale améliorée. Nous avons donc choisi d'utiliser le voisinage de la méthode de recherche locale présentée à la section 5.2.

6.2.1.2 Gestion des solutions taboues

Pour présenter la gestion des solutions taboues, nous considérons d'abord le cas mono-produit que nous généraliserons par la suite au cas multi-produits. Rappelons tout d'abord qu'une solution voisine est obtenue en ajoutant un et un seul arc u dans l'arbre de recouvrement associé à la solution courante du problème du suiveur S^i. Pour maintenir la structure d'arbre, il est nécessaire de faire sortir un arc v de l'arbre lors de l'ajout de u. À partir du couple (u, v), nous pouvons caractériser le passage d'une solution à l'autre. Pour cela, nous avons retenu trois attributs pour la mémoire à court terme répartis dans trois listes taboues :

- L_e : Liste de tabous des arcs entrants u dans l'arbre de recouvrement
- L_s : Liste de tabous des arcs sortants v de l'arbre de recouvrement
- L_p : Liste de tabous des pivots (u, v).

Le choix de trois listes est motivé par ce qui suit. La troisième liste L_p permet de caractériser la trajectoire de recherche dans l'espace des solutions grâce aux pivots. Cependant, cette liste n'est pas suffisante pour ne pas cycler dans l'espace de recherche. Nous considérons les listes L_e et L_s interdisant l'utilisation inverse des derniers mouvements alors que L_p interdit la répétition des pivots. Ces attributs sont maintenus tabous pendant un nombre d'itérations généré aléatoirement et uniformément dans les intervalles $[\underline{\theta}_e, \overline{\theta}_e]$,$[\underline{\theta}_s, \overline{\theta}_s]$ et $[\underline{\theta}_p, \overline{\theta}_p]$.

À chaque nouvelle visite de la meilleure solution, les bornes des intervalles sont augmentées de 1 pour induire un éloignement suffisant de cette solution et éviter d'y revenir. Notons que si une nouvelle solution est découverte, ces bornes sont réinitialisées.

Pour chaque solution voisine de S obtenue en pivotant u avec v, notée $S(u,v)$, le revenu du meneur généré est calculé en résolvant le POI avec $S(u,v)$. Le mouvement (u,v) implanté est donc celui fournissant le meilleur revenu pour le meneur parmi les solutions voisines non taboues et celles améliorant la meilleure solution connue. Précisons qu'une solution $S(u,v)$ est taboue si un des ses attributs est tabou.

Si aucun pivot ne satisfait les critères de sélection précédents, la solution $S(u,v)$ minimisant la fonction suivante est sélectionnée :

$$\Pi(u,v) = [t_e(u) - t]^+ + [t_s(v) - t]^+ + 2[t_p(u,v) - t]^+ \text{ avec } [a]^+ = \max\{0, a\},$$

t est le nombre courant d'itérations, $t_e(u)$ est l'itération où l'arc u a été inséré dans la liste L_e, $t_s(v)$ est l'itération où l'arc v a été inséré dans la liste L_s, et $t_p(u,v)$ est l'itération où le pivot (u,v) a été supprimé de la liste des tabous des pivots. Notons qu'initialement, les valeurs $t_r(u)$, $t_s(v)$ et $t_p(u,v)$ sont nulles pour tout $(u,v) \in \mathcal{A}$.

Pour étendre la mémoire à court terme au cas multi-produits, nous remarquons qu'un mouvement est défini par un arc entrant u, un arc sortant v et un produit k indiquant sur quel arbre le pivotage (u,v) doit être appliqué. Ainsi, les couples (u,k), (v,k) et les triplets (u,v,k) doivent être stockés respectivement dans la liste des tabous L_e, L_s et L_p. De plus, comme les interactions entre les produits sont prises en compte, les listes taboues sont mises à jour pour chaque produit dont le chemin emprunté par le suiveur a été modifié.

6.2.1.3 Mise à jour des arbres de recouvrement

Pour la mise à jour des arbres de recouvrement, nous exploitons les deux stratégies proposées pour la méthode de recherche locale (cf section 5.5.1.2). Le choix de la stratégie de mise à jour dépend de l'amélioration de la meilleure solution.

Lorsque la solution sélectionnée dans le voisinage n'améliore pas la meilleure solution connue, la mise à jour de l'arbre de recouvrement de chaque produit est effectuée par des pivots de l'algorithme du simplexe. Bien que cette stratégie n'ait pas permis d'obtenir de bons résultats avec la méthode de recherche locale, nous l'avons retenue car elle est adaptée pour l'exploration de l'espace de recherche en tenant compte des informations stockées dans les listes taboues.

Dans le cas contraire, la solution sélectionnée dans le voisinage améliore la meilleure solution connue, la mise à jour est effectuée en calculant les arbres de plus courts chemins de chaque produit en considérant les tarifs du meneur associés à la solution améliorante. Ce choix est motivé par les résultats numériques obtenus avec cette stratégie pour la méthode recherche locale.

6.2.1.4 Accélération de la méthode de recherche avec tabous

Pour ne pas avoir à visiter le voisinage des $|\mathcal{K}|$ produits à chaque itération, nous nous limitons à un sous-ensemble de produits. Une liste de k_1 produits explorés à chaque itération est maintenue. Initialement, cette liste est composée des k_1 produits ayant la plus forte demande. Puis, cette liste est mise à jour avec les $k_1/2$ produits fournissant les meilleures solutions à l'itération précédente et les $k_1/2$ produits ayant été le moins visités parmi ceux restant.

L'évaluation de chaque solution voisine nécessite la résolution d'un POI. Pour accé-

lérer l'évaluation du voisinage, nous proposons de n'effectuer qu'une seule itération de la méthode de génération de colonnes utilisée pour résoudre le POI présentée à la section 3.2.2.

6.2.2 Diversification

Une mémoire à long terme, conservant les solutions élites rencontrées pendant le processus de recherche, a été considérée afin de permettre des stratégies de diversification. À partir de ces solutions, nous essayons d'identifier les caractéristiques communes à ces solutions pour diriger la recherche. Pour cela, nous stockons les j meilleures solutions dans une liste. Lorsqu'une nouvelle solution améliorante est rencontrée, elle est ajoutée à la liste. Cependant, pour avoir une certaine diversité dans l'ensemble des solutions élites, nous imposons que les solutions contenues dans cette mémoire diffèrent deux à deux d'un certain nombre de chemins. Grâce à cette liste, nous calculons des statistiques sur la fréquence d'apparition des chemins de chaque produit dans les "bonnes" solutions. Ces informations sont ensuite utilisées pour guider les stratégies de diversification.

Une première stratégie de diversification repose sur l'utilisation de la méthode exacte présentée au chapitre 4. Ce choix est motivé par l'efficacité de celle-ci sur des instances ayant peu de produits. Notre but est donc de fixer des parties de solutions (c'est-à-dire des chemins pour un certain nombre de produits). Les parties sont fixées de la manière suivante. À partir d'une solution du suiveur, nous calculons pour chaque chemin la fréquence d'apparition par rapport à l'ensemble des meilleures solutions. Ensuite, nous fixons les chemins dont la fréquence est supérieure à un certain seuil et nous appliquons la méthode exacte. Avec cette stratégie, nous garantissons que la solution finale est au moins aussi bonne que la solution courante. Cependant, cette stratégie n'a pas été retenue car elle n'a pas permis

d'obtenir des bons résultats dans les tests préliminaires. En effet, elle nécessite de fixer un trop grand nombre de chemins, ce qui réduit la diversification.

Nous avons choisi une deuxième approche plus naïve pour la diversification. Son principe est de s'éloigner de la solution courante dans le but d'explorer des nouvelles régions. Pour cela, nous nous servons de la méthode de recherche avec tabous mais en considérant une nouvelle fonction objectif. Soit une solution du suiveur S et S^* la meilleure solution courante, la fonction objectif $Z_{div}(S, S^*)$ est la somme du revenu $Z(S)$ et du produit de la distance entre les solutions S à S^*, noté $D(S, S^*)$[1], par la valeur du meilleur revenu trouvée $Z(S^*)$. Nous avons ainsi

$$Z_{div}(S) = Z(S) + Z(S^*)D(S, S^*).$$

Cette fonction permet à la méthode de recherche avec tabous de se diriger vers des régions éloignées mais vers des solutions procurant un bon revenu pour le meneur.

Pour renforcer la diversification, les informations collectées par la mémoire à long terme sont utilisées. Pour chaque produit, si un chemin est utilisé par 90% des solutions élites, nous le fixons pendant tout la phase de diversification.

6.3 Algorithme de la méthode avec tabous pour le PTR

Nous décrivons l'algorithme de la méthode de recherche avec tabous pour le PTR dont les paramètres d'entrée sont $[\underline{\theta}, \overline{\theta}]$, k_1, $maxiter$, it_{div}. L'algorithme alterne entre deux phases : une phase de diversification (Phase_Diversification == vrai) et

[1]Soit (x, y) (resp. (x^*, y^*)) les variables du suiveur associées à la solution S (resp. S^*). La distance $D(S, S^*)$ entre S et S^* est définie par

$$D(S, S^*) = ||(x, y) - (x^*, y^*)||$$

une phase d'exploration locale (Phase_Diversification == faux).

Dans l'algorithme, le triplet (S^{**}, Z^{**}, T^{**}) représente la meilleure solution trouvée (S^{**} est la solution du problème du suiveur, T^{**} est le vecteur des tarifs du meneur et Z^{**} la valeur du revenu du meneur). De la même manière, (S^*, Z^*, T^*) représente la meilleure solution trouvée dans la phase courante en considérant la fonction objectif appropriée. Le compteur d'itérations est défini par l. i représente le nombre d'itérations sans amélioration de (S^*, Z^*, T^*) pour la phase d'exploration locale ou le nombre d'itérations effectuées par la phase de diversification. Finalement, L_{od} est la liste des produits considérés pour l'évaluation du voisinage.

Phase_Tabou($[\underline{\theta}, \overline{\theta}]$, k_1, *maxiter*, it_{div})

PAS 0 *Initialisation*

- Soient S^0 une solution du problème du suiveur et (Z^0, T^0) le revenu et le plan tarifaire optimal obtenus en résolvant le POI avec S^0.
- $l = i = 0$.
- $(S^*, Z^*, T^*) = (S^{**}, Z^{**}, T^{**}) = (S^0, Z^0, T^0)$.
- Initialiser L_{od} avec les k_1 produits ayant la plus forte demande.
- Phase_Diversification = faux.

PAS 1 *Exploration du voisinage*

- $(Z^+, T^+) = (-\infty, +\infty)$.
- Pour chaque $k \in L_{od}$,
  - Soit $L^k(S^l)$ l'ensemble des solutions voisines de S^l pour le produit k.
  - **Pour** chaque $S \in L^k(S^l)$,
    - Soit $\tilde{Z}$ la valeur obtenu en effectuant une itération de la méthode génération de colonne pour résoudre POI avec S.
      - **Si** Phase_Diversification == faux et $\tilde{Z} > Z^+$,
        alors $S^+ = S, Z^+ = \tilde{Z}$.
      - **Si** Phase_Diversification == vrai et $\tilde{Z} + Z(S^*)D(S, S^*) > Z^+$,

alors $S^+ = S, Z^+ = \tilde{Z} + Z(S^*)D(S, S^*)$.

- Mise à jour de L_{od} (cf section 6.2.1.4).

PAS 2 *Mise à jour de la solution courante*

- $l = l + 1$.
- $S^l = S^+$.
- Mise à jour des listes taboues L_e, L_s etL_p.
- Soient (Z^l, T^l) le revenu et le plan tarifaire optimal obtenus en résolvant le POI avec S^l.
- **Si** $S^l == S^*$ **alors** $[\underline{\theta}, \overline{\theta}] = [\underline{\theta} + 1, \overline{\theta} + 1]$.
- **Si** Phase_Diversification == faux **alors**
  - **Si** $Z^l > Z^*$ **alors**
    - $(S^*, Z^*, T^*) = (S^l, Z^l, T^l)$.
    - Réinitialiser $[\underline{\theta}, \overline{\theta}]$.
    - Mise à jour des arbres de recouvrement de chaque produit en calculant un arbre de plus court chemin en considérant les tarifs T^l.
    - $i = 0$.

    Sinon
    - Mise à jour des arbres de recouvrement de chaque produit en effectuant des pivots la méthode du simplexe.
    - $i = i + 1$.

  Sinon {*phase de diversification* }
  - Mise à jour des arbres de recouvrement de chaque produit en effectuant des pivots la méthode du simplexe.
  - $i = i + 1$.
  - **Si** $Z^l + Z^* D(S^l, S^*) > Z^*$

    alors $(S^*, Z^*, T^*) = (S^l, Z^l + Z^* D(S^l, S^*), T^l)$.

PAS 3 *Gestion des solutions élites*

- **Si** $Z^l > Z^*$ **alors** $(S^{**}, Z^{**}, T^{**}) = (S^l, Z^l, T^l)$.

- Mise à jour de la liste des solutions élites (c.f. section 6.2.2)

PAS 4 *Critère d'arrêt*

- **Si** $l \geq maxiter$ **alors** arrêt avec la solution (S^{**}, Z^{**}, T^{**}).

PAS 5 *Diversification*

- **Si** Phase_Diversification == faux et $i \geq it_{div}$

  alors

  - Phase_Diversification = vrai.
  - $i = 0$.
  - Réinitialiser $[\underline{\theta}, \overline{\theta}]$.
  - Fixer les chemins de S^l en se basant sur les solutions élites conservées (c.f. section 6.2.2)
  - $(S^*, Z^*, T^*) = (S^l, Z^l, T^l)$.
- **Si** Phase_Diversification == vrai et $i == |\mathcal{K}|$ **alors**
  - Phase_Diversification = faux.
  - $i = 0$.
  - Mise à jour des arbres de recouvrement de chaque produit en calculant un arbre de plus court chemin en considérant les tarifs T^l.
  - $(S^*, Z^*, T^*) = (S^l, Z^l, T^l)$.
- Aller au pas 1.

6.4 Résultats numériques de la méthode de recherche avec tabous

Les résultats numériques sont répartis dans deux sections. Dans la première partie, nous étalonnons la méthode de recherche avec tabou. Dans la deuxième partie, la méthode de recherche avec tabous est testée sur un grand nombre d'instances.

Avant de présenter les résultats numériques, précisons que les détails d'implantation des méthodes sont identiques à ceux présentés à la section 4.7.

6.5 Étalonnage de la méthode de recherche avec tabous

Dans cette section, nous étudions l'influence des différents paramètres de la méthode de recherche avec tabous. Plus précisément, les paramètres considérés sont :

- Le nombre d'itérations pendant lequel un mouvement est tabou. Pour cela, l'intervalle de sélection est le même pour les trois listes taboues c'est-à-dire

$$\begin{aligned} \underline{\theta} &= \underline{\theta}_{in} = \underline{\theta}_{out} = \underline{\theta}_{piv}, \\ \overline{\theta} &= \overline{\theta}_{in} = \overline{\theta}_{out} = \overline{\theta}_{piv}. \end{aligned}$$

- Le nombre d'itérations it_{div} sans amélioration de la meilleure solution requis avant d'effectuer une phase de diversification.
- Le nombre d'itérations maximal *maxiter* avant l'arrêt de la recherche.
- Le nombre de produits k_1 considérés à chaque exploration du voisinage.

Les instances utilisées dans ces tests sont des réseaux sous forme de grille de 60 noeuds et 208 arcs ayant, d'une part, 10, 20, 30 et 40 produits avec 15% d'arcs tarifables que nous dénoterons comme les instances de type A et, d'autre part, 20 produits avec un pourcentage d'arcs tarifables de 5%, 10%, 15% et 20% que nous dénoterons comme les instances de type B. Pour chaque type d'instances, les arcs tarifables ont été sélectionnés à l'aide de la méthode proposée par Brotcorne et al. (2000) (cf. annexe I).

Les résultats numériques sont donnés dans les tableaux 6.1 à 6.4. La première ligne du tableau indique les valeurs des paramètres fixés et la deuxième détaille les valeurs du paramètre testé. Les trois premières colonnes définissent les caractéristiques des instances, c'est-à-dire le nombre de produits (#OD), le pourcentage d'arcs tarifables ($\%T$) et les contraintes de borne inférieure sur les tarifs (T). Pour chaque méthode de recherche avec tabous testée, nous reportons la qualité de la solution (%) définie par Z/Z_{inf} où Z (resp. Z_{inf}) est la valeur de la meilleure solution obtenue avec la

méthode de recherche avec tabous (resp. la méthode exacte dont les temps de calcul ont été bornés à 2 heures) et le temps de calcul en secondes (CPU). Les résultats sont des moyennes sur 5 instances.

6.5.1 Influence des intervalles pour la longueur de la liste des tabous

Nous testons l'influence des intervalles pour la longueur de la liste des tabous sur la qualité des solutions fournies par la méthode de recherche avec tabous. Pour cela, les paramètres suivants ont été fixés $maxiter = 500$, $k_1 = 25\% \times |\mathcal{K}|$, $it_{div} = +\infty$ (c'est-à-dire qu'il n'y a pas de diversification) et le paramètre $\underline{\theta}$ varie entre les valeurs 3, 7, 11 et 15. $\overline{\theta}$ est fixé à $\underline{\theta} + 5$. Les résultats numériques sont reportés dans le tableau 6.1.

Pour les instances de type A, les meilleurs résultats ont été obtenus avec le paramètre $\underline{\theta} = 3$ sauf pour les instances possédant 20% d'arcs tarifables où le paramètre $\underline{\theta} = 7$ permet d'obtenir les meilleurs solutions. Notons toutefois que cette différence de qualité est inférieure à 0.1%.

Pour les instances de type B, le paramètre $\underline{\theta} = 3$ permet d'obtenir les meilleures solutions à l'exception des instances avec 40 produits où elles sont obtenues avec $\underline{\theta} = 7$. Cependant, contrairement aux instances de types A, la différence de qualité de ces solutions est de l'ordre de 2%.

Nous avons choisi de fixer le paramètre $\underline{\theta}$ à 3 car il permet une exploration locale de l'espace des solutions. De plus, la stratégie de diversification va permettre d'améliorer la qualité des solutions en explorant de nouvelles régions de l'espace des solutions.

Tableau 6.1 Influence des intervalles pour la longueur de la liste des tabous

paramètres $[\underline{\theta}, \overline{\theta}]$			$k_1 = 25\% \times \lvert\mathcal{K}\rvert$, $maxiter = 500$, $it_{div} = +\infty$							
			[3, 8]		[7, 12]		[11, 6]		[15, 20]	
#OD	$\%T$	T	%	CPU	%	CPU	%	CPU	%	CPU
20	5	≥ 0	100.00	5.51	99.07	5.37	98.16	5.18	98.45	5.18
20	10	≥ 0	100.00	8.68	100.00	8.53	99.76	8.41	99.44	8.16
20	15	≥ 0	99.79	12.41	99.54	11.97	99.51	12.40	99.08	12.26
20	20	≥ 0	101.77	18.88	101.83	19.79	101.44	19.96	101.19	19.46
10	15	≥ 0	100.00	3.92	99.35	3.94	98.60	3.89	94.61	3.81
20	15	≥ 0	99.79	12.41	99.54	11.97	99.51	12.40	99.08	12.26
30	15	≥ 0	99.88	20.57	99.19	19.26	99.75	20.53	99.71	20.98
40	15	≥ 0	109.71	34.17	111.43	33.15	111.13	33.57	111.18	33.71
moyenne			101.37	14.57	101.24	14.25	100.98	14.54	100.34	14.48

6.5.2 Influence de la fréquence des phases de diversification

Nous étudions maintenant l'impact de la phase de diversification présentée à la section 6.2.2 sur la qualité des solutions fournies par la méthode de recherche avec tabous. Plus précisément, nous voulons déterminer la fréquence d'application de ces phases. Une phase de diversification est effectuée si pendant un certain nombre d'itérations, noté it_{div}, il n'y a pas eu d'amélioration de la meilleure solution. Nous avons donc fait varier it_{div} entre les valeurs $|\mathcal{K}|/2$, $|\mathcal{K}|$, $1.5|\mathcal{K}|$, $2|\mathcal{K}|$, $+\infty$. Notons que la valeur $+\infty$ correspond au cas où aucune diversification n'est effectuée. Pour ces tests, les autres paramètres ont été considérés $[\underline{\theta}, \overline{\theta}] = [3, 8]$, $k_1 = 25\% \times |\mathcal{K}|$ et $maxiter = 500$. Les résultats numériques sont reportés dans le tableau 6.2.

Pour les instances de type A, nous remarquons qu'en moyenne les meilleurs résultats ont été obtenus sans l'utilisation de phase de diversification. Ces résultats s'expliquent par le fait que la valeur it_{div} est trop petite (à cause du nombre de produits) ce qui implique que la méthode de recherche avec tabous n'a pas le temps d'explorer suffisamment la région de l'espace des solutions courantes qu'elle doit

Tableau 6.2 Influence du paramètre it_{div}

paramètres			$[\underline{\theta}, \overline{\theta}] = [3,8]$, $k_1 = 25\% \times \|\mathcal{K}\|$, $maxiter = 500$									
it_{div}			$\|\mathcal{K}\|/2$		$\|\mathcal{K}\|$		$1.5\|\mathcal{K}\|$		$2\|\mathcal{K}\|$		$+\infty$	
#OD	$\%T$	T	%	CPU	%	CPU	%	CPU	%	CPU	%	CPU
20	5	≥ 0	99.14	5.67	99.14	5.64	99.95	5.64	99.14	5.57	100.00	5.51
20	10	≥ 0	95.28	8.58	99.89	8.74	99.89	8.73	100.00	8.76	100.00	8.68
20	15	≥ 0	99.54	12.02	99.89	12.43	99.79	12.42	99.89	12.34	99.79	12.41
20	20	≥ 0	101.61	17.70	101.71	18.35	101.76	18.69	101.35	17.66	101.77	18.88
10	15	≥ 0	98.91	4.12	98.15	4.02	99.76	4.07	99.92	3.99	100.00	3.92
20	15	≥ 0	99.54	12.02	99.89	12.43	99.79	12.42	99.89	12.34	99.79	12.41
30	15	≥ 0	99.91	19.75	99.93	20.42	100.04	20.22	99.93	20.07	99.88	20.57
40	15	≥ 0	109.20	33.30	109.52	34.01	111.31	32.95	111.29	33.88	109.71	34.17
moyenne			100.39	14.15	101.02	14.51	101.54	14.39	101.43	14.33	101.37	14.57

commencer l'exploration d'une autre.

Pour les instances de type B, lorsque la valeur it_{div} est trop grande, l'exploration stagne dans une région de l'espace (par exemple pour les instances possédant 40 produits). À contrario, lorsqu'elle est trop petite, nous nous retrouvons dans le même cas que celui évoqué pour les instances de types A. Les meilleures solutions ont été obtenues en fixant le paramètre à it_{div} à $1.5|\mathcal{K}|$.

Nous avons choisi de fixer le paramètre it_{div} à $1.5|\mathcal{K}|$ car la méthode de recherche avec tabous est destinée à la résolution d'instances comprenant un grand nombre de produits.

6.5.3 Influence du paramètre *maxiter*

Nous nous intéressons à présent au critère d'arrêt de la méthode de recherche avec tabous. Nous faisons donc varier la valeur du paramètre *maxiter* entre 500, 2000, 4000, 8000 et 16000. Les autres paramètres ont été fixés à $[\underline{\theta}, \overline{\theta}] = [3,8]$, $k_1 = 25\% \times |\mathcal{K}|$, $it_{div} = 1.5|K|$. Les résultats numériques sont reportés dans le

Tableau 6.3 Influence du paramètre *maxiter*

paramètres			$[\underline{\theta}, \overline{\theta}] = [3, 8]$, $k_1 = 25\% \times \lvert\mathcal{K}\rvert$, $it_{div} = 1.5\lvert K\rvert$									
maxiter			100		500		2000		4000		16000	
#OD	%T	T	%	CPU	%	CPU	%	CPU	%	CPU	%	CPU
20	5	≥ 0	99.95	1.26	99.95	5.67	100.00	16.24	100.00	42.85	100.00	171.91
20	10	≥ 0	99.89	1.97	99.89	8.52	99.89	24.62	99.94	65.56	100.00	266.16
20	15	≥ 0	99.31	2.78	99.79	12.36	100.00	36.38	100.00	96.93	100.00	399.25
20	20	≥ 0	101.31	4.00	101.76	18.83	101.78	56.79	101.82	152.01	101.86	601.38
10	15	≥ 0	99.76	0.89	99.76	4.07	99.79	11.92	99.79	31.80	100.00	130.36
20	15	≥ 0	99.31	2.78	99.79	12.36	100.00	36.38	100.00	96.93	100.00	399.25
30	15	≥ 0	99.01	4.55	100.04	21.00	100.09	61.06	100.14	164.07	100.14	676.51
40	15	≥ 0	107.95	7.23	111.31	32.96	111.33	95.24	111.42	257.92	111.44	1070.24
moyenne			100.81	3.18	101.54	14.47	101.61	42.33	101.62	113.35	101.68	464.38

tableau 6.3.

Pour les instances de type A et B, nous constatons que la qualité des solutions semble augmenter de manière logarithmique en fonction du paramètre *maxiter* et l'évolution des temps de calcul en fonction de *maxiter* semble être linéaire. Ce paramètre *maxiter* semble surtout sensible au nombre de produits. Ce phénomène s'explique par le fait que pour une valeur fixée de *maxiter*, lorsque le nombre de produits augmente, il y a une diminution du nombre de diversification, donc un risque de laisser des parties de l'espace des solutions contenant des "bonnes solutions" inexplorées. Nous avons choisi de fixer le paramètre *maxiter* à $50|\mathcal{K}|$ pour les expérimentations numériques.

6.5.4 Influence du nombre de produits évalués à chaque itération

Finalement, nous nous sommes intéressés à l'influence du nombre de produits considérés sur la qualité des solutions fournies par la méthode de recherche avec tabous. Pour cela, nous avons testé cette méthode en faisant varier le nombre de produits entre $25\% \times |\mathcal{K}|$, $50\% \times |\mathcal{K}|$, $75\% \times |\mathcal{K}|$ et $100\% \times |\mathcal{K}|$. Les durées taboues ont été

Tableau 6.4 Influence du paramètre k_1

paramètres			$[\underline{\theta}, \overline{\theta}] = [12\|\mathcal{K}\|/k_1, 12\|\mathcal{K}\|/k_1 + 5]$, $maxiter = 2000$, $it_{div} = 1.5\|\mathcal{K}\|$							
k_1			25% × $\|\mathcal{K}\|$		50% × $\|\mathcal{K}\|$		75% × $\|\mathcal{K}\|$		100% × $\|\mathcal{K}\|$	
#OD	%T	Borne	%	CPU	%	CPU	%	CPU	%	CPU
20	5	≥ 0	100.00	21.57	100.00	38.35	100.00	54.29	100.00	63.69
20	10	≥ 0	99.89	32.89	100.00	58.66	100.00	87.36	100.00	107.24
20	15	≥ 0	100.00	48.44	100.00	87.55	100.00	133.75	100.00	164.81
20	20	≥ 0	101.78	75.90	101.87	140.19	101.87	209.31	101.79	260.78
10	15	≥ 0	99.79	15.89	100.00	34.70	100.00	48.13	100.00	64.54
20	15	≥ 0	100.00	48.44	100.00	87.55	100.00	133.75	100.00	164.81
30	15	≥ 0	100.14	80.47	100.14	157.53	100.14	228.85	100.09	292.24
40	15	≥ 0	111.33	125.78	111.23	243.39	110.95	352.38	111.38	452.45
moyenne			101.62	56.17	101.66	105.99	101.62	155.98	101.66	196.32

ajustées en fonction de la taille des voisinages. Plus précisément, la durée taboue associée à 100% des produits est 4 fois supérieure à celle associés à 25% des produits. Les autres paramètres considérés sont $maxiter = 2000$ et $it_{div} = 1.5|\mathcal{K}|$. Les résultats numériques sont reportés dans le tableau 6.4.

Pour les instances de type A et B, la qualité des solutions est peu influencée par le paramètre k_1. Cependant, les temps de calcul sont logiquement proportionnels au paramètres k_1, c'est-à-dire qu'ils sont doublés quand le nombre de produits considéré est doublé. Nous choisissons de fixer le paramètre à 25% pour les tests effectués par la suite sur la méthode de recherche avec tabous.

6.6 Résultats numériques pour le PTR

Dans cette section, nous présentons les résultats numériques de la méthode avec tabous calibrée. Pour ces tests, nous avons considéré les 5 types d'instances présentées à l'annexe I regroupant un total de 640 instances.

Les résultats numériques sont des moyennes sur 5 instances. Ils sont repris dans les tableaux 6.5 à 6.9. Les trois premières colonnes définissent les caractéristiques des instances résolues c'est-à-dire le nombre de produits (#OD), le pourcentage d'arcs tarifables ($\%T$) et les contraintes de borne inférieure sur les tarifs (T). Les colonnes suivantes sont consacrées aux résultats obtenus avec les méthodes exactes. Si les tarifs sont positifs (resp. libres), la résolution exacte est effectuée par la résolution de la formulation MIP par le logiciel commercial CPLEX (2006) (MIP^+)(resp. par la méthode exacte présentée au chapitre 4 (DMS)) où les temps de résolution ont été bornés à 12 heures (43200 secondes). Pour chaque méthode exacte, nous avons reporté le nombre d'instances résolues à l'optimalité (NOpt), la valeur de la borne inférieure Z_{inf}, la qualité de la solution (%), le nombre d'itérations (# it) et le temps de calcul en secondes (CPU). La qualité des solutions est donnée par $(Z_{sup} - Z_{inf})/Z_{inf}$ où Z_{sup} est la valeur de la borne supérieure sur le revenu du meneur obtenue à la fin de la résolution. Ensuite, pour la méthode de recherche avec tabous, nous reportons la valeur de la meilleure solution trouvée (Z), la qualité de cette solution (%) par rapport à celle fournie par la méthode exacte, la qualité de la moins bonne solution obtenue (%min) parmi les 5 instances, le nombre d'itérations (#it) et le temps de calcul en secondes (CPU). Nous indiquons en plus, le nombre d'itérations (*it) et le temps de calcul (*CPU) pour la découverte de la meilleure solution. La qualité des solutions est définie par Z/Z_{inf}.

Nous observons tout d'abord que les temps de calcul de la méthode de recherche avec tabous sont négligeables par rapport à ceux de la méthode exacte pour l'ensemble des instances. De plus, le nombre d'itérations requis pour trouver la solution avec le meilleur revenu est en moyenne deux fois plus petit que le nombre d'itérations maximal fixé.

La méthode de recherche avec tabous fournit des solutions de très bonne qualité sur l'ensemble des familles d'instances. Plus précisément, lorsque les instances sont

résolues à l'optimalité, les solutions de la méthode de recherche avec tabous se situent à moins de 1% de la solution optimale. Pour les instances où la solution optimale n'a pu être trouvée ou prouvée en moins de 12 heures, les solutions de la méthode de recherche avec tabous sont meilleures et, dans le pire des cas, identiques à celles obtenues avec les méthodes exactes.

Finalement, nous avons représenté, sur les figures 6.1 et 6.2, l'évolution de la valeur de la meilleure solution pendant le processus de résolution des méthodes exactes et des méthodes de recherche avec tabous. Notons que l'échelle de l'axe des abscisses est logarithmique. Ces courbes montrent la rapidité de la méthode de recherche avec tabous pour la découverte de solutions de bonne qualité par rapport aux méthodes exactes.

6.7 Conclusions

Dans ce chapitre, nous avons présenté une méthode de recherche avec tabous pour le PTR basée sur la méthode de recherche locale proposée au chapitre précédent. À la gestion des mouvements tabous, des stratégies d'accélération pour l'évaluation du voisinage ont été mise en place comme l'utilisation heuristique de la méthode de génération de colonne pour la résolution du problème d'optimisation inverse et la réduction du nombre de produits évalué à chaque itération. De plus, nous avons proposé une stratégie de diversification utilisant une mémoire à long terme conservant les solutions élites.

Cette méthode a été testée en deux phases. La première est une phase d'étalonnage où l'influence de chaque paramètre sur le comportement de la méthode est analysée. Dans la deuxième phase, les performances de la méthode de recherche avec tabous exploitant les résultats de la phase d'étalonnage sont étudiés sur une collection

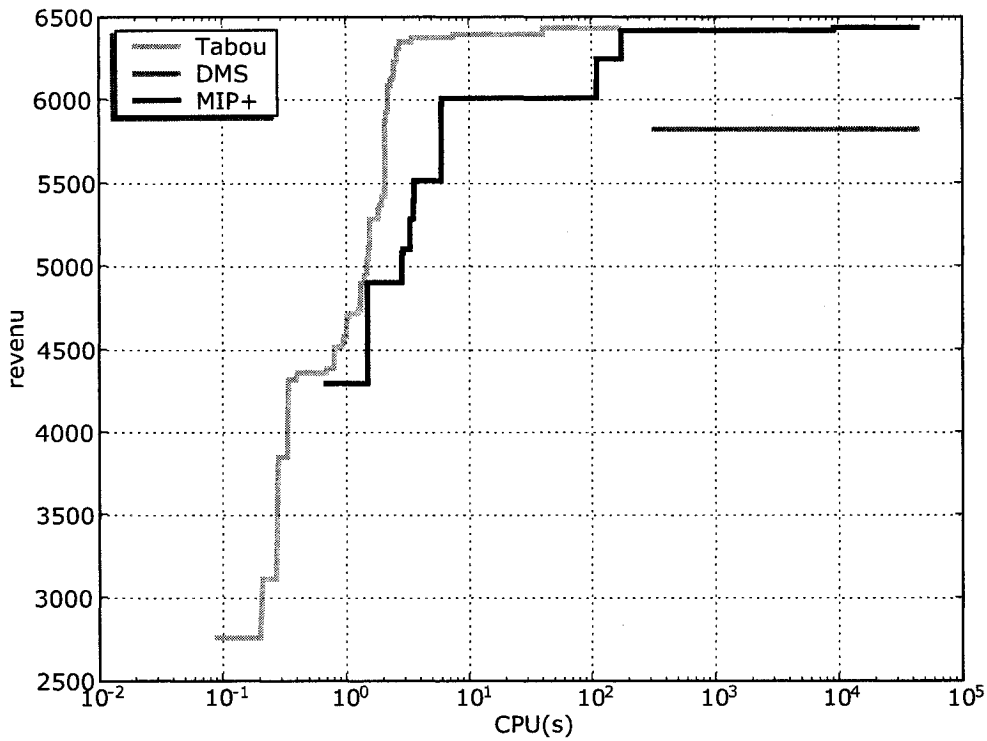


Figure 6.1 Évolution de la meilleure solution en fonction des temps de calcul de la méthode de recherche avec tabous et des méthodes exactes sur une instance de Brotcorne *et al.* de 30 produits avec 15% d'arcs tarifables lorsque les tarifs sont positifs

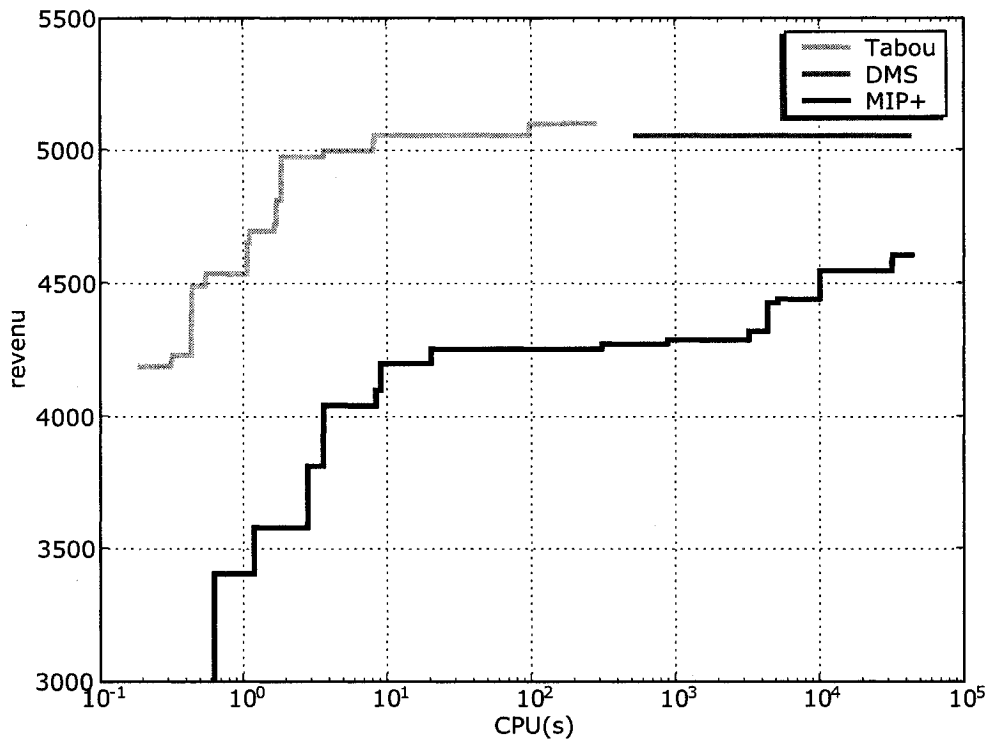


Figure 6.2 Évolution de la meilleure solution en fonction des temps de calcul de la méthode de recherche avec tabous et des méthodes exactes sur une instances de Brotcorne *et al.* de 30 produits avec 15% d'arcs tarifables lorsque les tarifs sont libres

d'instance du PTR. Les résultats numériques ont montré que cette méthode fournit des solutions de très bonne qualité en des temps de calcul très courts par rapport aux méthodes exactes sur l'ensemble des familles d'instances.

Tableau 6.5 Résultats numériques obtenus par la méthode de recherche avec tabous sur les instances de Brotcorne *et al*

			MIP$^+$				Méthode de recherche avec tabous $[\underline{\theta}, \overline{\theta}] = [3, 8]$, $maxiter = 50\|K\|$ $k_1 = 25\%$, $it_{div} = 1.5\|K\|$							
#OD	$\%T$	T	NOpt	Z_{inf}	%	#it	CPU	Z	min%	%	#it	CPU	*it	*CPU
10	5	≥ 0	5	1246.36	0.00	17.67	0.75	1238.13	97.71	99.34	500.00	2.17	144.98	0.65
10	10	≥ 0	5	2204.96	0.00	94.69	3.70	2199.75	99.52	99.76	500.00	3.03	82.21	0.56
10	15	≥ 0	5	3088.08	0.00	1870.58	30.98	3030.21	93.24	98.13	500.00	4.29	83.22	0.74
10	20	≥ 0	5	3894.22	0.00	9677.69	170.67	3876.10	97.77	99.53	500.00	6.13	113.45	1.40
20	5	≥ 0	5	1720.77	0.00	53.56	4.66	1718.43	99.09	99.86	1000.00	9.81	158.47	1.54
20	10	≥ 0	5	3483.78	0.00	12318.03	311.92	3483.78	100.00	100.00	1000.00	15.75	274.72	4.50
20	15	≥ 0	4	4847.57	0.91	602916.67	16689.36	4847.36	99.98	100.00	1000.00	23.62	447.27	10.44
20	20	≥ 0	1	6386.51	5.79	613583.33	38975.47	6420.28	100.00	100.53	1000.00	33.65	220.79	7.44
30	5	≥ 0	5	2075.21	0.00	295.00	16.63	2070.55	98.08	99.78	1500.00	24.37	85.76	1.40
30	10	≥ 0	5	3928.73	0.00	14052.13	620.10	3928.73	100.00	100.00	1500.00	40.68	55.62	1.72
30	15	≥ 0	2	5904.41	3.37	610416.67	39566.84	6002.15	100.00	100.13	1500.00	58.64	592.75	22.48
30	20	≥ 0	0	8040.76	13.38	307361.11	43558.65	8215.96	100.97	102.18	1500.00	88.33	190.89	10.90
40	5	≥ 0	5	2430.08	0.00	345.59	74.68	2430.08	100.00	100.00	2000.00	44.72	290.99	6.71
40	10	≥ 0	1	4724.06	4.65	346722.22	37079.00	4733.26	100.00	100.19	2000.00	80.81	344.83	13.80
40	15	≥ 0	0	6607.13	17.53	182277.78	43492.37	6782.20	101.79	102.65	2000.00	119.48	251.04	15.63
40	20	≥ 0	0	8597.03	23.43	116583.33	43440.05	8916.57	100.93	103.72	2000.00	188.67	632.37	59.47

			DMS				Méthode de recherche avec tabous $[\underline{\theta}, \overline{\theta}] = [3, 8]$, $maxiter = 50\|K\|$ $k_1 = 25\%$, $it_{div} = 1.5\|K\|$							
#OD	$\%T$	T	NOpt	Z_{inf}	%	#it	CPU	Z	min%	%	#it	CPU	*it	*CPU
10	5	Libre	5	1259.91	0.00	346.61	1.74	1256.76	99.09	99.75	500.00	2.61	182.54	1.00
10	10	Libre	5	2241.93	0.00	57753.76	211.46	2229.77	98.16	99.46	500.00	4.08	133.83	1.18
10	15	Libre	5	3140.41	0.00	480524.02	2117.46	3116.82	98.01	99.25	500.00	6.51	115.18	1.48
10	20	Libre	5	4020.19	0.00	1076958.53	6780.29	3980.86	97.12	99.02	500.00	9.28	82.68	1.56
20	5	Libre	5	1735.35	0.00	12940.92	41.77	1735.35	100.00	100.00	1000.00	11.48	44.85	0.61
20	10	Libre	1	3533.29	1.03	6586320.56	39414.43	3539.22	99.67	100.17	1000.00	21.66	214.38	4.82
20	15	Libre	0	4919.28	11.03	4866872.74	43200.00	5039.22	100.29	102.44	1000.00	40.20	198.15	7.83
20	20	Libre	0	6464.59	13.48	3334660.38	43200.00	6681.63	100.00	103.36	1000.00	56.13	298.07	15.92
30	5	Libre	5	2077.30	0.00	345027.20	935.33	2070.30	97.13	99.66	1500.00	29.21	54.02	1.22
30	10	Libre	0	3870.88	9.85	5200749.67	43200.00	3954.67	100.00	102.16	1500.00	56.83	103.61	4.01
30	15	Libre	0	5538.67	25.10	2384113.74	43200.00	6161.24	100.04	111.24	1500.00	93.68	221.19	12.87
30	20	Libre	0	8149.57	20.25	1115621.37	43200.58	8531.31	100.18	104.68	1500.00	152.46	391.74	40.80
40	5	Libre	4	2428.74	0.76	3373667.09	15714.19	2432.17	100.00	100.14	2000.00	56.91	330.93	9.46
40	10	Libre	0	4289.78	30.58	3072971.89	43200.00	4735.63	100.51	110.39	2000.00	118.97	574.66	37.82
40	15	Libre	0	6635.79	25.31	1814703.23	43200.00	6960.63	100.00	104.90	2000.00	213.36	635.84	64.70
40	20	Libre	0	8494.05	32.47	1058196.89	43200.91	9267.24	104.10	109.10	2000.00	368.85	358.37	57.28

Tableau 6.6 Résultats numériques obtenus par la méthode de recherche avec tabous avec une structure de grilles

			MIP$^+$					Méthode de recherche avec tabous $[\underline{\theta}, \overline{\theta}] = [3, 8]$, $maxiter = 50\|\mathcal{K}\|$ $k_1 = 25\%$, $it_{div} = 1.5\|\mathcal{K}\|$							
#OD	$\%T$	T	NOpt	Z_{inf}	%	#it	CPU	Z	min%	%	#it	CPU	*it	*CPU
10	5	≥ 0	5	909.05	0.00	2.48	0.21	906.55	96.25	99.72	500.00	1.59	53.41	0.20
10	10	≥ 0	5	1647.22	0.00	11.04	0.49	1647.22	100.00	100.00	500.00	2.06	50.17	0.22
10	15	≥ 0	5	2306.28	0.00	24.00	0.98	2306.28	100.00	100.00	500.00	2.68	260.30	1.33
10	20	≥ 0	5	3073.72	0.00	70.47	1.83	3052.17	97.69	99.30	500.00	3.34	123.32	0.82
50	5	≥ 0	5	2674.59	0.00	528.31	44.50	2674.59	100.00	100.00	2500.00	63.83	506.10	12.44
50	10	≥ 0	5	5741.34	0.00	93495.28	5918.32	5683.79	94.83	99.00	2500.00	103.56	169.45	7.08
50	15	≥ 0	2	7709.40	3.21	199595.56	26960.66	7730.65	100.00	100.28	2500.00	145.76	566.50	33.59
50	20	≥ 0	0	11069.07	7.65	278463.06	43663.27	11151.28	100.17	100.74	2500.00	183.67	669.16	51.94
100	5	≥ 0	5	4536.22	0.00	1725.09	208.72	4506.34	96.79	99.34	5000.00	308.99	885.09	50.94
100	10	≥ 0	1	8156.28	11.68	125805.56	41128.73	8240.58	99.94	101.03	5000.00	587.97	1692.94	205.75
100	15	≥ 0	0	13016.29	18.46	77770.28	43471.72	13318.53	99.75	102.32	5000.00	858.42	1802.75	283.33
100	20	≥ 0	0	16440.22	23.90	51708.06	43414.58	17293.59	101.77	105.19	5000.00	1190.63	1785.24	375.75

			DMS					Méthode de recherche avec tabous $[\underline{\theta}, \overline{\theta}] = [3, 8]$, $maxiter = 50\|\mathcal{K}\|$ $k_1 = 25\%$, $it_{div} = 1.5\|\mathcal{K}\|$						
#OD	$\%T$	T	NOpt	Z_{inf}	%	#it	CPU	Z	min%	%	#it	CPU	*it	*CPU
10	5	Libre	5	932.27	0.00	17.11	1.71	932.27	100.00	100.00	500.00	1.80	119.54	0.42
10	10	Libre	5	1671.01	0.00	234.91	2.70	1670.16	99.77	99.95	500.00	2.65	183.89	0.94
10	15	Libre	5	2370.04	0.00	830.87	7.47	2367.29	99.58	99.88	500.00	3.68	71.97	0.62
10	20	Libre	5	3186.15	0.00	3090.15	16.47	3179.70	99.52	99.80	500.00	5.32	181.71	1.95
50	5	Libre	4	2772.36	2.58	2702511.53	16070.86	2769.94	99.38	99.91	2500.00	85.88	426.09	15.40
50	10	Libre	0	5397.19	16.47	2932134.44	43200.00	5752.89	101.33	106.59	2500.00	151.19	674.92	36.27
50	15	Libre	0	7621.95	19.67	1036278.89	43211.59	8051.47	100.92	105.64	2500.00	234.43	770.26	74.47
50	20	Libre	0	10821.84	19.40	998294.22	43205.89	11581.90	100.71	107.02	2500.00	359.81	869.13	112.71
100	5	Libre	2	4380.56	7.67	2618425.88	32292.67	4484.93	92.28	102.38	5000.00	412.90	1092.28	94.95
100	10	Libre	0	7846.57	29.00	888248.30	43200.00	8346.93	101.52	106.38	5000.00	850.57	807.23	131.67
100	15	Libre	0	12491.86	32.40	553459.99	43200.02	13698.38	101.10	109.66	5000.00	1424.08	2282.19	692.35
100	20	Libre	0	16850.21	26.54	300559.46	43200.06	17735.67	101.04	105.25	5000.00	2234.62	2502.20	1076.78

Tableau 6.7 Résultats numériques obtenus par la méthode de recherche avec tabous sur instances avec une structure de Didi *et al*

			MIP$^+$					Méthode de recherche avec tabous $[\underline{\theta}, \overline{\theta}] = [3, 8]$, $maxiter = 50\|\mathcal{K}\|$ $k_1 = 25\%$, $it_{div} = 1.5\|\mathcal{K}\|$					
#OD	$\%T$	T	NOpt	Z_{inf}	%	#it	CPU	Z	min%	%	#it	CPU	*it	*CPU
10	5	≥ 0	5	2295.96	0.00	0.00	0.04	2295.96	100.00	100.00	500.00	1.40	36.82	0.10
10	10	≥ 0	5	4652.86	0.00	0.00	0.06	4652.86	100.00	100.00	500.00	1.61	57.96	0.20
10	15	≥ 0	5	5193.26	0.00	0.43	0.15	5159.86	98.05	99.36	500.00	1.59	78.76	0.27
10	20	≥ 0	5	9430.84	0.00	1.40	0.24	9430.84	100.00	100.00	500.00	1.86	70.75	0.29
50	5	≥ 0	5	7866.90	0.00	0.00	1.00	7866.90	100.00	100.00	2500.00	37.05	315.69	4.70
50	10	≥ 0	5	15766.32	0.00	13.59	4.71	15749.78	99.55	99.90	2500.00	44.07	229.14	4.02
50	15	≥ 0	5	21532.19	0.00	95.97	16.96	21525.32	99.83	99.97	2500.00	50.31	128.74	2.80
50	20	≥ 0	5	32054.05	0.00	1697.78	184.52	32020.05	99.58	99.89	2500.00	58.62	307.40	7.77
100	5	≥ 0	5	14990.16	0.00	8.42	4.97	14888.57	96.83	99.32	5000.00	164.84	163.81	5.60
100	10	≥ 0	5	28699.48	0.00	395.18	61.80	28697.18	99.96	99.99	5000.00	217.35	1032.16	47.32
100	15	≥ 0	5	37431.06	0.00	3812.73	597.15	37192.48	97.34	99.36	5000.00	246.02	2541.20	123.71
100	20	≥ 0	2	55997.78	4.02	84178.89	28239.38	56185.76	99.33	100.34	5000.00	320.42	380.01	26.43

			DMS					Méthode de recherche avec tabous $[\underline{\theta}, \overline{\theta}] = [3, 8]$, $maxiter = 50\|\mathcal{K}\|$ $k_1 = 25\%$, $it_{div} = 1.5\|\mathcal{K}\|$					
#OD	$\%T$	T	NOpt	Z_{inf}	%	#it	CPU	Z	min%	%	#it	CPU	*it	*CPU
10	5	Libre	5	2318.46	0.00	2.00	7.83	2318.46	100.00	100.00	500.00	1.86	77.86	0.31
10	10	Libre	5	4737.86	0.00	5.61	1.90	4737.86	100.00	100.00	500.00	2.80	49.59	0.32
10	15	Libre	5	5280.55	0.00	2.97	2.16	5251.35	98.34	99.45	500.00	3.99	177.22	1.41
10	20	Libre	5	9654.57	0.00	175.13	5.78	9654.57	100.00	100.00	500.00	5.34	57.45	0.62
50	5	Libre	5	7885.59	0.00	91.03	120.32	7885.59	100.00	100.00	2500.00	49.81	42.79	0.96
50	10	Libre	4	15803.34	0.04	280267.24	7861.66	15803.34	100.00	100.00	2500.00	75.84	437.42	12.92
50	15	Libre	1	21765.93	3.59	888458.20	34597.63	21762.49	99.61	99.98	2500.00	119.90	330.12	15.04
50	20	Libre	0	31192.76	16.22	896025.68	43200.02	32525.81	100.00	104.27	2500.00	172.72	325.66	19.35
100	5	Libre	5	14990.16	0.00	73445.02	4069.33	14990.16	100.00	100.00	5000.00	250.23	1020.52	53.24
100	10	Libre	0	26540.76	20.59	955007.84	43200.00	28568.09	100.00	107.64	5000.00	421.45	614.45	55.87
100	15	Libre	0	35422.34	26.78	395328.20	43200.04	37566.05	100.34	106.05	5000.00	593.81	1690.93	197.72
100	20	Libre	0	54513.61	33.38	201067.76	43200.06	56515.82	100.00	103.67	5000.00	913.30	1312.70	208.88

Tableau 6.8 Résultats numériques obtenus par la méthode de recherche avec tabous sur instances avec une structure de Delaunay

			MIP$^+$					Méthode de recherche avec tabous $[\underline{\theta}, \overline{\theta}] = [3, 8]$, $maxiter = 50\|\mathcal{K}\|$ $k_1 = 25\%$, $it_{div} = 1.5\|\mathcal{K}\|$						
#OD	$\%T$	T	NOpt	Z_{inf}	%	#it	CPU	Z	min%	%	#it	CPU	*it	*CPU
10	5	≥ 0	5	2438.71	0.00	0.73	0.30	2438.71	100.00	100.00	500.00	2.64	206.89	1.13
10	10	≥ 0	5	4453.15	0.00	2.47	0.39	4432.08	98.08	99.53	500.00	3.15	75.36	0.54
10	15	≥ 0	5	5107.57	0.00	0.17	0.37	5098.28	99.11	99.82	500.00	3.71	89.20	0.69
10	20	≥ 0	5	5527.38	0.00	0.00	0.46	5527.38	100.00	100.00	500.00	4.33	105.88	0.98
50	5	≥ 0	5	8780.36	0.00	35.24	18.77	8780.36	100.00	100.00	2500.00	78.33	484.74	14.92
50	10	≥ 0	5	15182.66	0.00	722.50	173.98	15182.66	100.00	100.00	2500.00	106.79	606.37	27.94
50	15	≥ 0	5	19285.80	0.00	2525.56	513.62	19285.80	100.00	100.00	2500.00	137.88	1422.29	79.63
50	20	≥ 0	3	22566.28	0.30	101769.72	21027.24	22555.19	99.80	99.95	2500.00	176.66	569.47	38.68
100	5	≥ 0	5	12143.63	0.00	796.81	269.86	12139.57	99.82	99.97	5000.00	346.20	896.53	64.43
100	10	≥ 0	4	23115.40	0.74	44136.11	20052.50	23100.46	99.75	99.94	5000.00	496.26	369.50	36.17
100	15	≥ 0	1	29234.82	5.68	53682.78	42976.56	29467.93	100.00	100.80	5000.00	735.92	1480.40	221.57
100	20	≥ 0	0	34226.27	9.04	29572.22	43385.53	34728.17	100.89	101.47	5000.00	1037.87	1544.29	319.39

			DMS					Méthode de recherche avec tabous $[\underline{\theta}, \overline{\theta}] = [3, 8]$, $maxiter = 50\|\mathcal{K}\|$ $k_1 = 25\%$, $it_{div} = 1.5\|\mathcal{K}\|$						
#OD	$\%T$	T	NOpt	Z_{inf}	%	#it	CPU	Z	min%	%	#it	CPU	*it	*CPU
10	5	Libre	5	2532.86	0.00	6.64	7.86	2532.86	100.00	100.00	500.00	3.17	118.08	0.78
10	10	Libre	5	4544.37	0.00	3.21	1.52	4542.02	99.78	99.95	500.00	4.51	143.78	1.39
10	15	Libre	5	5245.72	0.00	2.21	2.34	5245.72	100.00	100.00	500.00	6.25	77.86	1.10
10	20	Libre	5	5607.38	0.00	61.96	5.61	5607.38	100.00	100.00	500.00	9.95	109.80	2.13
50	5	Libre	5	8864.13	0.00	446008.52	4934.84	8864.13	100.00	100.00	2500.00	99.12	168.13	6.61
50	10	Libre	0	15019.57	5.51	1838456.01	43200.00	15256.45	100.00	101.58	2500.00	169.77	173.06	10.72
50	15	Libre	0	19224.06	5.95	1215454.63	43200.97	19385.84	100.00	100.84	2500.00	313.60	684.73	80.23
50	20	Libre	0	22657.94	6.09	947519.55	43200.00	22771.54	99.73	100.50	2500.00	479.12	1519.39	288.95
100	5	Libre	0	11698.23	10.87	1657471.03	43200.00	12219.89	100.11	104.46	5000.00	411.76	671.36	58.28
100	10	Libre	0	22793.78	15.83	501145.10	43200.00	23321.15	100.00	102.31	5000.00	839.72	819.52	138.17
100	15	Libre	0	28783.86	19.17	288167.36	43200.02	29647.40	101.26	103.00	5000.00	1633.58	2198.87	686.96
100	20	Libre	0	34199.63	20.76	167939.59	43200.08	34990.42	101.44	102.31	5000.00	3448.37	2976.48	1980.47

Tableau 6.9 Résultats numériques obtenus par la méthode de recherche avec tabous sur instances avec une structure de Voronoï

			MIP+					Méthode de recherche avec tabous $[\underline{\theta}, \overline{\theta}] = [3, 8]$, $maxiter = 50\|\mathcal{K}\|$ $k_1 = 25\%$, $it_{div} = 1.5\|\mathcal{K}\|$						
#OD	%T	T	NOpt	Z_{inf}	%	#it	CPU	Z	min%	%	#it	CPU	*it	*CPU
10	5	≥ 0	5	2368.73	0.00	0.00	0.67	2368.73	100.00	100.00	500.00	1.93	31.59	0.14
10	10	≥ 0	5	4665.81	0.00	80.29	2.11	4661.81	99.54	99.91	500.00	2.42	29.24	0.19
10	15	≥ 0	5	6642.13	0.00	112.73	3.08	6639.63	99.79	99.96	500.00	2.90	128.21	0.87
10	20	≥ 0	5	10315.62	0.00	715.31	11.10	10300.32	99.53	99.85	500.00	3.53	221.57	1.68
50	5	≥ 0	5	8909.47	0.00	472.50	59.16	8898.64	99.33	99.88	2500.00	72.87	592.26	16.03
50	10	≥ 0	1	17258.37	5.33	233234.17	38679.99	17290.81	99.15	100.19	2500.00	116.94	748.53	35.18
50	15	≥ 0	0	30899.79	10.47	137925.28	43523.17	31052.34	98.62	100.49	2500.00	157.45	782.11	45.66
50	20	≥ 0	0	47480.59	14.81	81613.06	43444.53	48104.64	99.83	101.31	2500.00	209.41	446.49	33.54
100	5	≥ 0	3	15760.46	0.43	106395.83	23615.18	15673.41	97.28	99.45	5000.00	413.65	1529.13	133.09
100	10	≥ 0	0	28249.23	19.98	50359.72	43475.47	28644.30	93.86	101.40	5000.00	681.76	1239.45	156.53
100	15	≥ 0	0	43601.57	31.11	25456.67	43380.90	45008.09	96.96	103.23	5000.00	1039.09	2559.26	532.20
100	20	≥ 0	0	72427.82	30.34	18755.56	43363.03	76784.50	103.66	106.02	5000.00	1411.75	1995.76	567.27

			DMS					Méthode de recherche avec tabous $[\underline{\theta}, \overline{\theta}] = [3, 8]$, $maxiter = 50\|\mathcal{K}\|$ $k_1 = 25\%$, $it_{div} = 1.5\|\mathcal{K}\|$						
#OD	%T	T	NOpt	Z_{inf}	%	#it	CPU	Z	min%	%	#it	CPU	*it	*CPU
10	5	Libre	5	2531.58	0.00	1310.67	2.77	2531.58	100.00	100.00	500.00	2.16	47.41	0.27
10	10	Libre	5	5086.45	0.00	7169.05	20.56	5066.95	97.80	99.62	500.00	2.78	155.47	0.91
10	15	Libre	5	7232.52	0.00	3277.03	12.98	7226.72	99.68	99.92	500.00	3.68	236.52	1.77
10	20	Libre	5	11101.38	0.00	7523.13	33.00	11062.37	98.45	99.65	500.00	5.05	193.65	2.18
50	5	Libre	3	8978.98	1.07	3653036.29	22646.06	8968.14	99.33	99.88	2500.00	91.65	155.13	5.62
50	10	Libre	0	17686.23	23.01	2519521.72	43200.00	18128.14	98.26	102.50	2500.00	149.10	906.85	54.22
50	15	Libre	0	28256.18	41.76	1382847.71	43200.00	33513.42	103.10	118.61	2500.00	215.11	501.57	40.19
50	20	Libre	0	40129.28	52.86	941072.68	43200.00	50714.16	113.78	126.38	2500.00	296.72	1026.56	122.17
100	5	Libre	0	15234.40	16.53	2900387.54	43200.00	16029.16	99.99	105.22	5000.00	538.12	1061.46	117.66
100	10	Libre	0	27468.93	40.58	771199.92	43200.02	29093.23	102.15	105.91	5000.00	943.32	1388.44	238.32
100	15	Libre	0	40788.95	60.87	419706.82	43200.04	46357.10	102.82	113.65	5000.00	1509.31	3013.82	986.93
100	20	Libre	0	59852.12	74.33	176218.53	43200.08	79256.08	105.44	132.42	5000.00	2249.78	2888.06	1241.92

CONCLUSION

Dans cette thèse, nous avons étudié un problème de tarification sur un réseau (PTR). Plus précisément, la situation où un meneur veut maximiser son revenu en tarifant des tronçons du réseau est considéré. Pour cela, il est nécessaire que le meneur tienne compte de la réaction des utilisateurs du réseau c'est à dire un suiveur devant acheminer des produits d'une origine vers une destination à moindre coût à travers le réseau. Ce problème est formulé sous forme d'un problème mathématique à deux niveaux bilinéaire-bilinéaire de façon à modéliser l'interaction hiérarchique existante entre les deux agents de décision (meneur, suiveur).

Le problème de détermination de tarifs reproduisant une solution du problème du suiveur connue et maximisant le revenu du meneur (appelé problème d'optimisation inverse (POI)) constitue une étape majeure pour les méthodes de résolution développées dans cette thèse. Le chapitre 3 est consacré au développement d'une méthode de génération de colonnes efficace pour ce problème. Elle généralise au cas multi-produits celle proposée par Labbé et al. (1998) pour le cas mono-produit. La rapidité de résolution de la méthode de génération de colonnes est mise en évidence au travers des expérimentations numériques.

La première méthode de résolution du PTR que nous avons développée au chapitre 4 est une méthode exacte basée sur le concept de K-chemins proposé par Didi et al. (1999). Elle consiste à énumérer de façon efficace les solutions du problème du suiveur (K-chemins) évaluées en résolvant un POI. D'une part, le nombre de solutions non réalisables visitées par la méthode a été réduit grâce à un prétraitement appliqué pendant la génération des K-chemins. D'autre part, une nouvelle structure de données a été introduite afin d'éliminer la visite des solutions redondantes. Cette structure a également été exploitée pour améliorer la qualité des bornes supé-

rieures sur le revenu du meneur. Les performances de la méthode exacte proposée dans cette thèse ont été comparées à celles du solveur commercial CPLEX pour la résolution de la formulation MIP. Au vu des résultats numériques, la méthode exacte est très efficace pour résoudre des instances du PTR où aucune contrainte n'est imposée sur les tarifs.

Le deuxième méthode de résolution du PTR proposée au chapitre 5 est une méthode de recherche locale basée sur l'exploration d'optima locaux du PTR. Ces optima locaux se caractérisent à partir des solutions du problème du suiveur et du POI. La méthode de recherche locale explore à chaque itération un voisinage des solutions du problème du suiveur évalué en terme du revenu du meneur par la résolution du POI. Plusieurs stratégies ont été proposées pour la mise à jour des solutions courantes et la réduction de la taille du voisinage exploré à chaque itération. Après les avoir testées, nous avons mis en évidence l'importance des mises à jour avec des arbres de plus court chemin sur la qualité des solutions et des évaluations partielles du voisinage pour la réduction des temps de calcul. Néanmoins, cette méthode s'arrête prématurément suite à l'absence de solutions améliorantes dans le voisinage immédiat.

Le dernière méthode de résolution du PTR présentée au chapitre 6 est une méthode de recherche avec tabous basée sur la méthode de recherche locale décrite au chapitre 5. Après avoir mis en place les mécanismes liés à la métaheuristique tabou, des améliorations accélérant l'exploration du voisinage ont été proposées et une stratégie de diversification basée sur des solutions élites a été introduite. Lors d'une phase d'échantillonnage, les valeurs de plusieurs paramètres ont pu être fixées. Nous avons ensuite testé cette méthode sur une collection d'instances. Nous avons observé l'efficacité de la méthode de recherche avec tabous pour la découverte de solutions de très bonne qualité en des temps de calculs très courts par rapport aux résultats obtenus par les méthodes exactes sur des instances de 10 à

100 produits et avec 5% à 20% d'arcs tarifables.

Dans les futures recherches, nous allons travailler sur l'amélioration des méthodes de résolutions exactes et heuristiques développées dans cette thèse. Tout d'abord, pour la méthode exacte basée sur le concept de K-chemins, nous voulons réduire les temps de calcul requis pour le calcul des bornes supérieures sur le revenu du meneur généré par les suffixes en développant une stratégie de décomposition.

D'autre part, nous désirons développer une approche par décomposition pour le calcul des bornes supérieures intervenant dans l'algorithme de séparation et d'évaluation pour la résolution des formulations linéaires mixtes du PTR. Cette dernière approche nous semble être la plus prometteuse pour la résolution de problèmes de plus grande taille du PTR.

Finalement, pour la méthode de recherche avec tabous, nous voulons intégrer une stratégie d'intensification afin de sonder plus efficacement les régions de l'espace de recherche prometteuses. Au final, nous désirons adapter cette méthode pour la résolution de problèmes plus réalistes comme par exemple lorsque toute la demande d'un produit n'est pas affectée au plus court chemin reliant son origine et sa destination.

RÉFÉRENCES

AIYOSHI, E. ET SHIMIZU, K. (1984). A solution method for the static constrained stackelberg problem via penalty method. *IEEE Transactions on Automatic Control*, **29**, 1111–1114.

AL-KHAYAL, F., HORST, R., ET P.M.PARDALOS (1992). Global optimization of concave functions subject to quadratic constraints : An application in nonlinear bilevel programming. *Annals of Operations Research*, **34**, 125–147.

BARD, J. (1991). Some properties of the bilevel programming problem. *Journal of Optimization Theory and Applications*, **68**, 371–378.

BARD, J. ET FALK, J. (1982). An explicit solution to the multi-level programming problem. *Computers and Operations Research*, **9**, 77–100.

BEN-AYED, O. (1993). Bilevel linear programming. *Computers and Operations Research*, **20**, 485–501.

BIALAS, W. ET KARWAN, M. (1984). Two-level linear programming. *Management Science*, **30**, 1004–1020.

BOUHTOU, M., ERBS, G., ET MINOUX, M. (2005). Princing and ressource allocation for point to point telecommunication services in a competitive market : a bilevel optimization approach. *Telecommunications Planning : Innovation in Princing, Networks design and Management*, **Springer**, 1–16.

BOUHTOU, M., HOESEL, S., KRAAIJ, A., ET LUTTON, J.-L. (2003). Tariff optimization in networks. *Meteor Research Memorandum RM03011, Maastricht University.*

BRACKEN, J. ET MCGILL, J. (1973). Mathematical programs with optimization problems in the contraints. *Operations Research*, **21**, 37–44.

BROTCORNE, L., CIRINEI, F., MARCOTTE, P., ET SAVARD, G. (2004a). A local search method for a pricing problem on a transportation. *TRISTAN V Conference, Guadeloupe, juin.*

BROTCORNE, L., CIRINEI, F., MARCOTTE, P., ET SAVARD, G. (2004b). Méthode de recherche locale pour un problème de tarification sur un réseau. *Les Journées de l'Optimisation, Montreal, Canada, mai.*

BROTCORNE, L., CIRINEI, F., MARCOTTE, P., ET SAVARD, G. (2005a). An exact algorithm for a pricing problem on a network. *Les Journées de l'Optimisation, Montreal, Canada, mai.*

BROTCORNE, L., CIRINEI, F., MARCOTTE, P., ET SAVARD, G. (2005b). An exact algorithm for bilevel toll setting. *INFORMS, San Francisco, novembre.*

BROTCORNE, L., CIRINEI, F., MARCOTTE, P., ET SAVARD, G. (2006a). A tabu search heuristic for a pricing problem on a network. *INFORMS, Pittsburgh, US, novembre.*

BROTCORNE, L., LABBÉ, M., MARCOTTE, P., ET SAVARD, G. (2000). A bilevel model and solution for a freight tariff setting problem. *Transportation Science*, **34**, 289–302.

BROTCORNE, L., LABBÉ, M., MARCOTTE, P., ET SAVARD, G. (2001). A bilevel model for toll optimization on a multicommodity transportation network. *Transportation Science*, **35**, 1–14.

BROTCORNE, L., LABBÉ, M., MARCOTTE, P., ET SAVARD, G. (2003). Joint design and pricing on a network. *Soumis à Operations Research.*

BROTCORNE, L., MARCOTTE, P., SAVARD, G., ET WIART, M. (2005c). Joint pricing and network capacity setting problem. *In Advanced OR and AL Methods in Transportation, Jaszkiewicz, Kaczmarek, Zak and Kubiak eds, Publishing House of Poznan University od Technology.*

BROTCORNE, L., P.MARCOTTE, ET WIART, M. (2006b). A bilevel approach for the hazardous materials routing problem. *Third International Workshop on Freight Transportation and Logistics (Odysseus).*

CANDLER, W. ET NORTON, R. (1977). Multilevel programming. *Tech. Rep., World Bank Development Research Center*, **20**.

CANDLER, W. ET TOWNSLEY, R. (1982). A linear two-level programming problem. *Computers and Operations Research*, **9**, 59–76.

CHOUMAN, M., MARCOTTE, P., ET SAVARD, G. (2005). Bilevel model for pricing telecommunications services. presented at Informs San Francisco.

COLSON, B., MARCOTTE, P., ET SAVARD, G. (2005a). A trust-region method for nonlinear bilevel programming : Algorithm and computational experience. *Computational Optimization and Applications*, **30(3)**, 211–227.

COLSON, B., MARCOTTE, P., ET SAVARD, S. (2005b). Bilevel programming : A survey. *4OR*, **3**, 87–107.

CÔTÉ, J., MARCOTTE, P., ET SAVARD, G. (2003). A bilevel modeling approach to pricing and fare optimization in the airline industry. *Journal of Revenue and Pricing Management*, **2**, 23–36.

CPLEX (2006). Using the cplex callable library and cplex mixed integer programming.

DEMPE, S. (2002). Foundations of bilevel programming. *Kluwer Academic Publisher, Dordrecht et al.*

DEWEZ, S. (2004). *On the Toll Setting Problem.* PhD thesis, Université Libre de Bruxelles.

DIDI, M., MARCOTTE, P., ET SAVARD, G. (1999). Optimal and heuristic approaches to a multicommodity toll-setting problem. *Rapport Interne du GERAD.*

EDMUNDS, T. ET BARD, J. (1991). Algorithms for nonlinear bilevel mathematical programming. *IEEE Transactions on Systems, Man, and Cybernetics*, **21**, 83–89.

FORTIN, M. (2005). Tarification avec segmentation de la demande et congestion. Master's thesis, École polytechnique.

FORTUNE, S. (1992). Voronoi diagrams and delaunay triangulations. *Computing in Euclidean Geometry*, pages 193–233.

FORTUNY-AMAT, J. ET MCCARL, B. (1981). A representation and economic interpretation of a two-level programming problem. *Journal of the Operational Research Society*, **32**, 783–792.

GAUVIN, J. ET SAVARD, G. (1994). The steepest descent direction for the nonlinear bilevel programming problem. *Operations Research Letters*, **15**, 275–282.

GENDREAU, M., MARCOTTE, P., ET SAVARD, G. (1996). A hybrid tabu-ascent algorithm for the linear bilevel programming problem. *Journal of Global Optimization*, **8**, 217–232.

GLOVER, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, **13**, 533–549.

GRIGORIEV, A., HOESEL, S. V., DER KRAAIJ, A. V., UETZ, M., ET BOUHTOU, M. (2005). Pricing network edges to cross a river. *Lecture Notes in Computer Science*, **3351/2005**, 140–153.

HANSEN, P. (1986). The steepest ascent mildest descent heuristic for combinatorial programming. *Congress on Numerical Methods in Combinatorial Optimization*, **Capri, Italy**.

HANSEN, P., JAUMARD, B., ET SAVARD, G. (1992). New branch-and-bound rules for linear bilevel programming. *Journal on Scientific and Statistical Computing*, **13**, 1194–1217.

ISHIZUKA, Y. ET AIYOSHI, E. (1992). Double penalty method for bilevel optimization problems. *Annals of Operations Research*, **34 n.1-4**, 73–88.

JÚDICE, J. ET FAUSTINO, A. (1992). A sequential lcp method for bilevel linear programming. *Annals of Operations Research*, **34**(1-4), 89–106.

JULSAIN, H. (1999). Tarification dans les réseaux de télécommunications : une approche par programmation mathématique à deux niveaux. Master's thesis, École polytechnique.

KOLSTAD, C. ET LASDON, L. (1990). Derivative estimation and computational experience with large bilevel mathematical programs. *Journal of Optimization Theory & Applications*, **65**, 485–499.

LABBÉ, M., MARCOTTE, P., ET SAVARD, G. (1998). A bilevel model of taxation and its application to optimal highway pricing. *Management Science*, pages 1595–1607.

LORIDAN, P. ET MORGAN, J. (1989a). New results on approximate solutions in two-level optimization. *Optimization*, **20**, 819– 836.

LORIDAN, P. ET MORGAN, J. (1989b). A theoretical approximation scheme for stackelberg problems. *Journal of Optimization Theory and Applications*, **61**, 95–110.

MARTINS, E. ET PASCOAL, M. (2003). A new implementation of yen's ranking loopless paths algorithm. *Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, **1(2)**, 121–134.

ROCH, S., MARCOTTE, P., ET SAVARD, G. (2003). Design and analysis of an approximation algorithm for stackelberg network pricing. *Cahiers du GERAD, G-2002-61, HEC Montreal, Montreal, Canada.*

TARJAN, R. (1981). Shortest paths. Technical report, AT&T Bell Laboratories, Murray Hill, NJ.

THOAI, N., YAMAMOTO, Y., ET YOSHISE, A. (2002). Global optimization method for solving mathematical programs with linear complementarity constraints.

Discussion Paper No. 987, Institute of Policy and Planning Sciences, University of Tsukuba,.

TUY, H., MIGDALAS, A., ET VARBRAND, P. (1993). A global optimization approach for the linear two-level program. *Journal of Global Optimization*, **3**, 1–23.

VICENTE, L., SAVARD, G., ET JÚDICE, J. (1994). Descent approaches for quadratic bilevel programming. *Journal of Optimization Theory and Applications*, **81**, 379–399.

VON STACKELBERG, H. (1952). *The Theory of the Market Economy*. PhD thesis, Oxford University Press.

WANG, I. (2003). *Shortest Paths and Multicommodity Network Flows*. PhD thesis, School of Industrial and Systems Engineering, Georgia Institute of Technology.

YEN, J. (1971). Finding the k-shortest loopless paths in a network. *Management Science*, **17**, 712–716.

ZUYEV, S., DESNOGUES, P., ET RAKOTOARISOA, H. (1996). Simulations of large telecommunication networks based on probabilistic modeling. Technical report, INRIA Sophia Antipolis.

ANNEXE I

COLLECTION D'INSTANCES POUR LE PROBLÈME DE TARIFICATION SUR UN RÉSEAU

Nous présentons une collection d'instances pour le problème de tarification sur un réseau. Nous discutons d'abord des différentes stratégies utilisées dans la littérature pour leurs constructions et nous décrivons ensuite celles de notre collection.

I.1 Stratégies de construction des instances du PTR

Nous supposons connu un graphe $\mathcal{G} =< \mathcal{N}, \mathcal{A} >$. À partir de celui-ci, une instance pour le PTR est obtenue en générant un ensemble de produits et un ensemble d'arcs tarifables.

I.1.1 Sélection des produits

Le choix des produits est important pour influencer l'utilisation de sous-chemins communs pour acheminer la demande de certains produits créant ainsi une interaction entre eux. Deux stratégies de sélection de produits ont été relevées dans la littérature et une nouvelle est proposée dans cette section.

La première stratégie (Didi et al. (1999); Grigoriev et al. (2005)) repose sur un choix totalement aléatoire de l'origine, la destination et la demande des produits. Plus précisément, l'origine et la destination sont sélectionnées parmi les noeuds de $\mathcal{N}$ et la demande est définie en suivant une distribution uniforme dans un intervalle. Cette

stratégie, nommée DEM_ALEA, a le désavantage de ne pas garantir d'interactions entre les produits ce qui implique que ces instances sont généralement faciles à résoudre.

Une autre approche, notée DEM_BLMS, a été proposée par Brotcorne *et al.* Brotcorne et al. (2001). Elle se base sur la géométrie du graphe utilisé. Plus précisément, les auteurs choisissent eux-même les produits de manière à engendrer des interactions. La demande est choisie aléatoirement en suivant une distribution uniforme dans un intervalle. En pratique, les instances résultantes sont plus difficiles à résoudre à cause de la forte interaction entre les produits.

Nous proposons une approche intermédiaire pouvant être automatisée pour la sélection des produits. Nous dénoterons cette stratégie DEM_BLMS_ALEA. Pour cela, nous supposons que les coordonnées de chaque noeud du graphe sont connues. Dans une première phase, nous cherchons les noeuds définissant l'enveloppe convexe de l'ensemble $\mathcal{N}$. À partir de ces noeuds, nous choisissons 25% des produits en sélectionnant la paire de noeuds (i, j) maximisant la distance les séparant et nous créons le nouveau produit ayant pour origine i et destination j. Ensuite, les 75% produits restants sont choisis suivant la stratégie DEM_ALEA. Finalement, la demande de chaque produit est choisie aléatoirement suivant une distribution uniforme dans un intervalle. En pratique, ces instances sont de difficulté intermédiaire.

I.1.2 Sélection des arcs tarifables

Une fois que les produits sont identifiés, le choix des arcs tarifables reste très important car ce sont eux qui attirent les flots et créent les sous-chemins communs entre les produits. Dans la littérature, nous avons relevé deux types de sélection.

La première Didi et al. (1999); Grigoriev et al. (2005) est une sélection aléatoire

des arcs tarifables. Nous la dénoterons TAR_ALEA. En pratique, ces instances sont faciles à résoudre car peu d'arcs tarifables sont attractifs pour plusieurs produits.

Une approche plus sophistiquée est proposée par Brotcorne *et al.* Brotcorne et al. (2001). Nous la nommerons TAR_BLMS. Son principe est de choisir des arcs susceptibles d'être utilisés par plusieurs produits. Pour cela, la fréquence d'apparition des arcs dans les plus courts chemins de chaque produit est calculée. Plus la fréquence d'un arc est élevée, plus son attractivité est grande par rapport aux produits. Les auteurs ont imposé que 66% des arcs tarifables sont choisis parmi les arcs ayant une fréquence d'apparition élevée et les 34% restants sont sélectionnés suivant la stratégie TAR_ALEA.

I.2 Famille des instances

Nous présentons à présent les familles composant notre collection d'instances. Elle différent par la structure du graphe $\mathcal{G}$ et par la stratégie de constructions des instances du PTR.

I.2.1 Instances de Didi, Marcotte et Savard Didi et al. (1999)

La première famille est basée sur un graphe $\mathcal{G}$ (60 noeuds et 200 arcs) construit de la manière suivante. Un cycle reliant chaque noeud du graphe est créé afin de garantir la connexité du réseau. Ensuite, des arcs sont ajoutés aléatoirement dans le réseau. La figure I.1(a) représente un exemple de ce graphe où les arcs du cycles sont en noir et les arcs ajoutés en gris.

Le coût des arcs varie de 2 à 20. À partir de ce graphe, les stratégies DEM_ALEA et TAR_ALEA sont appliquées. La demande de chaque produit varie entre 20 et

100, et le coût des arcs tarifables est divisé par 3 afin de les rendre plus attractifs.

I.2.2 Instances sous forme de Grille

Ces instances sont des grilles (12×5) de 60 noeuds et 206 arcs du même types que celles représentés sur la figure I.2.3(b). La ligne reliant deux noeuds sur la figure est équivalent à la représentation des arcs à double sens. Le coût des arcs sur ces grilles varie entre 2 à 20. Ces instances sont divisées en deux sous-familles.

Une première regroupe les instances proposées par de Brotcorne *et al.* Brotcorne et al. (2001) c'est-à-dire que les produits sont choisis suivant les stratégie DEM_BLMS et TAR_BLMS. La demande de chaque produit varie entre 5 et 10 et le coût des arcs tarifables est divisé par 2 afin de les rendre plus attractifs.

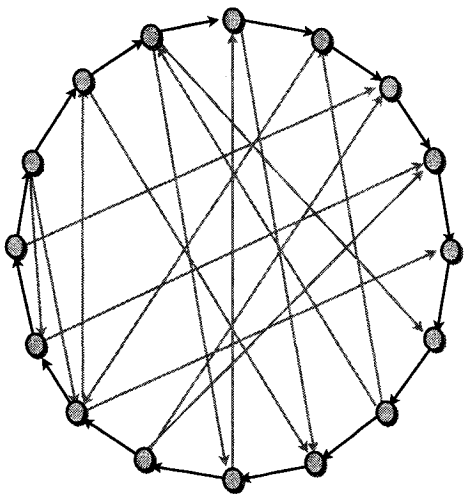
La deuxième regroupe les instances qui sont semblables à celles Brotcorne *et al.* mais la selection des produits est effectuée avec la stratégie DEM_BLMS_ALEA et la demande varie entre 1 et 5.

I.2.3 Instances sous forme de Delaunay

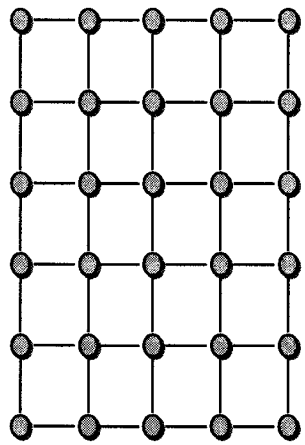
Nous proposons l'utilisation d'une nouvelle structure de graphe pour le PTR basée la triangularisation de Delaunay. Le choix de cette structure est motivé par le fait qu'elle est très souvent utilisée pour modéliser les réseaux de télécommunications comme par exemple dans Zuyev et al. (1996).

Rappelons qu'une triangularisation de Delaunay $\mathcal{T}_{\mathcal{N}}$ d'un ensemble de points $\mathcal{N}$ est la triangulation telle qu'aucun triangle de $\mathcal{T}_{\mathcal{N}}$ ne contient un autre point de $\mathcal{N}$ à l'intérieur de son cercle circonscrit. Nous utiliserons l'algorithme proposé par Fortune Fortune (1992) permettant de calculer cette triangularisation pour un

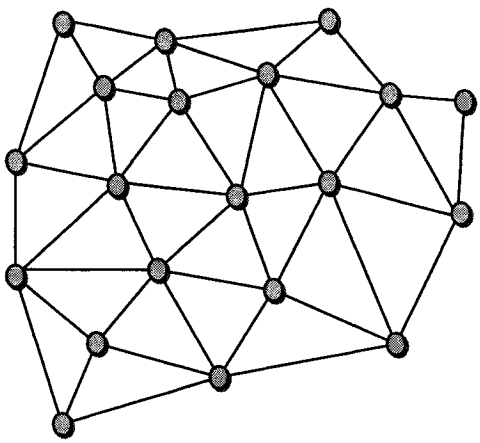
Figure I.1 Structure de graphes des différentes familles d'instances



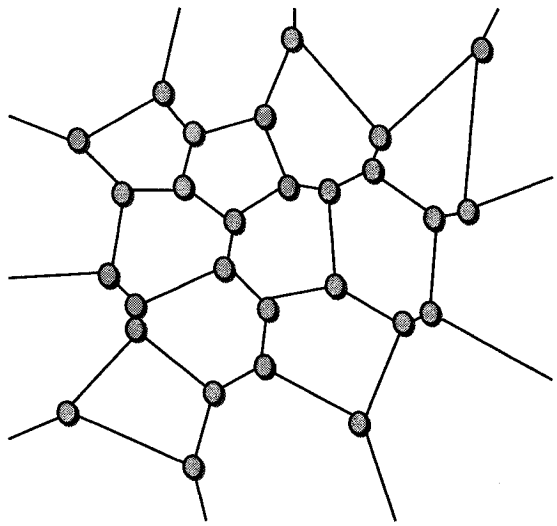
a) Structure de Didi, Marcotte et Savard



b) Structure de Grille



c) Structure de Delaunay



d) Structure de Voronoï

ensemble de point $\mathcal{N}$ en $O(|\mathcal{N}|\log(|\mathcal{N}|))$.

À partir de la triangularisation $\mathcal{T}_{\mathcal{N}}$ d'un ensemble de 60 points répartis uniformément, le graphe est obtenu en considérant les arrêts des triangles comme des arcs à double sens. Le coût de ses arcs varie entre 2 et 20. Un exemple de graphe est représentée à la figure I.2.3(c).

Avec ces graphes, les instances du PTR sont générées en considérant les stratégies DEM_BLMS_ALEA et TAR_BLMS. La demande varie entre 1 et 5 et le coût des arcs tarifables est divisé par 2 pour renforcer l'interaction entre les produits.

I.2.4 Instances sous forme de Voronoï

Nous avons retenu une autre structure de graphe basée sur les diagrammes de Voronoï. La particularité de ces graphes est que les noeuds sont de degrés 3. En regardant une carte routière, nous constatons que les points d'intersections des grands axes routiers sont aussi de degrés 3. Cette observation a motivé le choix de ces graphes.

Le diagramme de Voronoï est connu comme étant le dual d'une triangulations de Delaunay. Plus précisément, les noeuds de ces diagrammes sont les centres des cercles circonscrits des triangles appartenant à la triangularisation de Delaunay et les arcs à double sens sont obtenus en reliant les centres des cercles circonscrits de 2 triangles adjacents. Le diagramme de Voronoï associé à la triangularisation de la figure I.2.3(c) est représenté à la figure I.2.3(d). Notons que l'algorithme de Fortune Fortune (1992) permet de générer ces diagrammes.

Les graphes de cette famille d'instances sont générés à partir de triangularisations de Delaunay obtenue sur des ensembles de 60 points reparties uniformément. Les

stratégies DEM_BLMS_ALEA et TAR_BLMS sont appliquées. La demande varie entre 1 et 5 et le coût des arcs entre 2 à 20 et le coût des arcs tarifables est divisé par 2 pour renforcer l'interaction entre les produits.