N° d'Ordre : D.U. 1718
EDSPIC : 366

Université Blaise Pascal - Clermont-Ferrand II

École Doctorale
Sciences Pour l'Ingénieur de Clermont-Ferrand

THÈSE

présentée par

Sylvie BORNE

pour obtenir le grade de

Docteur d'université
Spécialité : Informatique

Sécurisation et dimensionnement de réseaux multicouches : modèles et polyèdres

Soutenue publiquement le 13 décembre 2006 devant le jury :

A.R.	Mahjoub	Directeur de thèse
A.	Quilliot	Président du jury
J.-F.	Maurras	Rapporteur
M.	Pinar	Rapporteur
F.	Semet	Rapporteur
E.	Gourdin	Examinateur
H.	Kerivin	Examinateur

Remerciements

Je voudrais tout d'abord remercier Monsieur A. Ridha Mahjoub, Professeur à l'Université Blaise Pascal de Clermont-Ferrand, pour la confiance qu'il m'a accordée en me permettant d'effectuer une thèse sous sa direction. Je tiens à lui exprimer ma plus profonde gratitude pour sa constante disponibilité, ses conseils toujours pertinents et son soutien dans les moments difficiles. J'ai particulièrement apprécié les longues heures que nous avons passées à travailler ensemble. Il a su me transmettre ses connaissances, son expérience et sa passion pour la recherche et l'enseignement, et ce depuis le tout premier jour. Pour tout cela, je lui témoigne ma plus sincère reconnaissance.

J'ai été très honorée que Monsieur Jean-François Maurras, Professeur à l'Université de la Méditerranée de Marseille, ait accepté de rapporter ma thèse. Je tiens à lui exprimer ma plus profonde reconnaissance.

Je remercie également Monsieur Mustafa Pinar, Professeur à Bilkent University d'Ankaras (Turquie), pour l'intérêt qu'il a bien voulu porter à ce travail et pour m'avoir fait l'honneur d'accepter la charge de rapporteur.

Ma gratitude va ensuite à Monsieur Frédéric Semet, Professeur à l'Université de Valenciennes et du Hainaut-Cambrésis, qui m'a fait l'honneur de rapporter ma thèse. Je le remercie pour l'attention toute particulière qu'il a porté à mon mémoire.

Mes remerciements vont également à Monsieur Alain Quilliot, Professeur à l'Université Blaise Pascal de Clermont-Ferrand, pour avoir accepté de présider le jury de cette thèse. De plus, je remercie Monsieur Hervé Kerivin, Maître de Conférences à l'Université Blaise Pascal de Clermont-Ferrand, d'avoir bien voulu examiner mes travaux et participer au jury.

Je suis également très reconnaissante envers Monsieur Eric Gourdin de France Télécom division Recherche et Développement, qui m'a fait le plaisir de participer au jury. Je lui adresse mes plus sincères remerciements pour sa collaboration aussi fructueuse qu'agréable.

Un immense merci, bien qu'insuffisant, à Pierre Fouilhoux et Pierre Pesneau pour m'avoir montré la route à suivre. Je leur suis extrêmement reconnaissante pour tout ce qu'ils ont fait même après leur départ. Je les remercie également pour leur constante disponibilité, pour m'avoir fait répéter sans relâche les différents exposés que j'ai présentés et pour leur lecture minutieuse de ce mémoire.

Un autre merci à Hervé Kerivin pour ses nombreux conseils et les discussions passionnantes que nous avons pu avoir.

Je voudrais également remercier plus globalement tous les membres de l'équipe EPOC (Equipe Polyèdre et Optimisation Combinatoire) du laboratoire LIMOS de Clermont-Ferrand, et en particulier Denis Cornaz, Ibrahima Diarrassouba, Mathieu Lacroix et Lise Slama pour tous les bons moments que nous avons passés ensemble ces dernières années. Je les remercie pour leur gaîté, leur bonne humeur et leur amitié. Je leur exprime ici tout ma sympathie. Enfin, j'adresse mes plus vifs remerciements à Fatiha et Jean Mailfert qui, tout au long de ces années, ont toujours fait preuve d'énormément de gentillesse et n'ont jamais été avares d'encouragements.

Enfin, ces remerciements seraient bien incomplets si je n'exprimais pas ma profonde reconnaissance à mes parents, à Pascale, Christophe, Arnaud et Clément, qui m'ont supportée durant ces quatre années de thèse. Et je n'oublie pas non plus mes amis que je remercie chalheureusement pour leur soutien inconditionnel, y compris durant mes périodes de doute. Ce sont dans les moments difficiles que l'on reconnait ses véritables amis.

Résumé

Les réseaux de télécommunications sont actuellement en plein essor. Les nouveaux systèmes optiques permettent une grande capacité de transmission et la gestion de données hétérogènes. Cela a entrainé une augmentation significative du trafic. Les réseaux de télécommunications évoluent vers des modèles qui consistent en un certain nombre de routeurs IP interconnectés par un réseau optique intelligent. De plus, il y a maintenant un consensus général sur le fait que le plan de contrôle des réseaux optiques devrait utiliser les protocoles et mécanismes développés pour IP comme MPLS et GMPLS, afin de fournir des circuits optiques de façon automatique et d'en assurer la restauration dynamique. L'interaction entre les routeurs IP et les brasseurs optiques permet d'assurer des connexions de bout-en-bout. Et les circuits optiques fournissent la topologie du réseau virtuel interconnectant les routeurs IP. Cette nouvelle infrastructure multicouche nécessite un haut niveau de fiabilité, de telle manière que les services du réseau puissent être rétablis rapidement en cas de panne.

Dans cette thèse, nous considérons différents problèmes liés à cette infrastructure multicouche des réseaux de télécommunications dans le cas de réseaux IP sur optique.

Après une présentation bibliographique de derniers développements des réseaux de télécommunications, nous considérons dans un premier temps un problème d'optimisation concernant la sécurisation de la couche IP. Nous donnons une formulation en termes de programmation linéaire en nombres entiers pour ce problème, et nous discutons du polytope associé. Nous décrivons plusieurs classes d'inégalités valides et étudions les conditions pour qu'elles définissent des facettes. Nous discutons de procédures de séparation pour ces inégalités et introduisons des opérations de réduction. Nous développons un algorithme de coupes et branchements basé sur nos résultats et présentons une étude expérimentale sur des instances aléatoires mais aussi sur des instances réelles.

Par la suite, nous introduisons une extension de ce problème qui considère également le dimensionnement du réseau IP en plus de la sécurisation. Nous présentons plusieurs

formulations en termes de programmes linéaires mixtes. Nous identifions différentes classes d'inégalités valides pour le problème et développons des algorithmes de coupes et branchements et de coupes, génération de colonnes et branchements pour résoudre le problème. Ceux-ci ont été utilisés pour résoudre des instances aléatoires et des instances réelles ayant un grand intérêt pour France Télécom.

Enfin, nous nous intéressons à deux derniers problèmes concernant la sécurisation multicouche du réseau optique et la sécurisation simultanée des deux couches. Nous proposons également une formulation en termes de programmes linéaires en nombres entiers pour ces problèmes. Aussi, nous développons une métaheuristique pour résoudre ces problèmes de manière approchée et présentons une étude expérimentale.

Mots clés : Réseaux de télécommunications multicouches, sécurisation, dimensionnement, polytope, facette, algorithme de coupes et branchements, génération de colonnes.

Table des matières

Introduction

Les réseaux de télécommunications sont actuellement en plein essor. Leur croissance se situe aussi bien au niveau du volume d'informations échangées qu'à l'étendue toujours plus importante de la couverture globale. Nous nous tournons de plus en plus vers l'Internet de haut débit. Les données véhiculées se diversifient davantage. Les réseaux actuels permettent d'acheminer de manière indifférenciée de la voix, des données et de l'image.

Un réseau de télécommunications peut être vu comme la superposition de deux couches : une couche cliente et une couche transport. Le traffic dominant les réseaux de télécommunications est basé de plus en plus sur des clients IP (Internet Protocol). De même, pour la couche transport, la technologie basée sur les brasseurs et les fibres optiques semble la plus adaptée pour effectuer la transmission d'informations à grande échelle. Des interfaces UNI (User-Network Interface) permettent la communication entre les deux couches. Actuellement, les deux couches sont gérées indépendemment. Les plans de commande et de transfert des deux couches sont séparés.

L'IETF (Internet Engineering Task Force) a identifié trois modèles d'interconnexion entre les deux couches, appelés Overlay, Augmented et Peer. Chacun se définit par l'échange de plus ou moins d'informations entre les couches. Le modèle Overlay est le plus couramment utilisé à l'heure actuelle. La topologie de chacune des couches est invisible pour l'autre. La gestion du réseau s'effectue donc séparément sur chaque couche. Les modèles Augmented et Peer permettent une certaine visibilité entre les couches, mais demandent un plan de commande commun. Un tel plan de commande est actuellement à l'étude. GMPLS (Generalized MultiProtocol Label Switching) est une généralisation du plan de commande MPLS permettant de gérer la signalisation et le routage sur la couche cliente. Il pourra être utilisé sur les deux couches et permettra donc une interaction entre les couches, en particulier pour le modèle Peer.

L'utilisation de cette nouvelle infrastructure de réseaux multicouches pose de nouveaux problèmes d'optimisation de réseaux. La sécurisation (fiabilité) reste un objec-

tif primordial pour les réseaux de télécommunications. Le problème de sécurisation consiste à déterminer les liaisons devant être installées dans un réseau de telle sorte que si une panne survient sur ces liaisons, le trafic puisse être rerouté et le réseau continue à fonctionner. On distingue également les problèmes concernant le dimensionnement de réseaux. Ce problème consiste à déterminer les liaisons à installer dans un réseau et la capacité de celles-ci de telle manière que le réseau puisse écouler des demandes fixées entre des sommets du réseau.

Les outils de l'optimisation combinatoire, en particulier la méthode dite *polyédrale*, se sont avérés très puissants pour aborder des problèmes combinatoires difficiles. Cette technique, initiée par Edmonds [29] dans le cadre du problème de couplage, consiste à ramener le problème en question à la résolution d'un programme linéaire (ou d'une séquence de programmes linéaires), par la description complète (ou partielle) de son polytope des solutions par un système d'inégalités linéaires. L'approche polyédrale a été appliquée avec succès à plusieurs problèmes d'optimisation combinatoire comme le problème du voyageur de commerce et celui de la coupe maximale.

Cette thèse s'inscrit dans le cadre des méthodes exactes de l'optimisation et la conception de réseaux de télécommunications. Nous présentons différents problèmes liés à l'infrastructure multicouche des réseaux de télécommunications. Nous proposons des modèles exacts pour ces problèmes. Nous présentons une étude polyédrale pour certains de ces problèmes et nous développons des algorithmes de coupes et branchements ou de coupes, génération de colonnes et branchements pour les résoudre. Nous développons également une métaheuristique de type recuit simulé pour résoudre certains de ces problèmes.

Dans le premier chapitre, nous introduisons quelques notions de base et quelques notations qui seront utiles tout au long de cette thèse. Nous présentons également un rapide état de l'art sur les problèmes de sécurisation et de dimensionnement de réseaux. Le chapitre 2 présente quelques notions de base sur l'architecture des réseaux de télécommunications multicouches. Les chapitre 3 et 4 concernent un premier problème de sécurisation du réseau IP. Les chapitre 5 et 6 discutent d'une extension du premier problème qui considère non plus uniquement la sécurisation du réseau mais également son dimensionnement. Le chapitre 7 porte sur deux autres problèmes. Le premier concerne la sécurisation du réseau optique et le deuxième la sécurisation simultanée des deux couches du réseau. Nous introduisons pour ces problèmes une métaheuristique de type recuit simulé pour les résoudre d'une manière approchée.

Chapitre 1

Notions préliminaires et état de l'art

Dans ce chapitre, nous donnons quelques notions de base sur l'optimisation combinatoire, la théorie de la complexité et les polyèdres combinatoires. Nous expliquons brièvement la méthode de coupes et branchements ainsi que la méthode de génération de colonnes et branchements. Nous donnons également des définitions et des notations qui seront utilisées tout au long de ce mémoire. Nous présentons également un rapide état de l'art sur des problèmes de conception de réseaux fiables et dimensionnement de réseaux.

1.1 Notions préliminaires

1.1.1 Optimisation Combinatoire

L'*Optimisation Combinatoire* est une des branches de l'informatique et des mathématiques appliquées. Elle concerne les problèmes pouvant se formuler de la façon suivante : soit $E = \{e_1, \ldots, e_n\}$ un ensemble fini appelé *ensemble de base*, où chaque élément e_i possède un *poids* $c(e_i)$. Soit $\mathcal{F}$ une famille de sous-ensembles de E. Si $F \in \mathcal{F}$, alors $c(F) = \sum_{e_i \in F} c(e_i)$ est le poids de F. Le problème consiste à déterminer un élément de $\mathcal{F}$, ayant le plus petit (ou le plus grand) poids. Un tel problème est appelé un *problème d'optimisation combinatoire*. L'ensemble $\mathcal{F}$ est appelé l'*ensemble des solutions* du problème.

Le mot *combinatoire* qui désigne la discipline des mathématiques concernée par les structures discrètes ou finies, évoque l'existence d'une structure sous-jacente discrète

(généralement un graphe). L'ensemble de solutions $\mathcal{F}$ est défini dans cette structure et peut avoir un nombre exponentiel d'éléments. Le mot *optimisation* veut dire que l'on recherche le meilleur élément de l'ensemble de solutions.

De nombreux problèmes dans la pratique peuvent se formuler comme des problèmes d'optimisation combinatoire, comme par exemple le problème du sac à dos, du voyageur de commerce, de conception de réseaux, de transport, de localisation, ... L'optimisation combinatoire se trouve au carrefour de la théorie des graphes, de la programmation linéaire et de la programmation linéaire en nombres entiers. Elle est très liée à la théorie de la complexité d'algorithmes.

1.1.2 Rappels de la théorie de la complexité

La théorie de la complexité est basée sur les travaux d'Edmonds [28] et de Cook [17]. Elle permet de classer un problème donné parmi les problèmes faciles ou difficiles. Dans cette section, nous allons en rappeler quelques éléments de base. La théorie de la NP-complétude est présentée en détail dans Garey et Johnson [35].

Un *problème* est une question générale possédant des paramètres dont la valeur n'est pas connue. Un problème est décrit en donnant : une description générale de tous les paramètres, et une énumération des propriétés que la solution doit satisfaire. Une *instance* d'un problème est obtenue en spécifiant la valeur de chaque paramètre du problème. Un *algorithme* de résolution d'un problème donné est une procédure, décomposable en opérations élémentaires, qui pour chaque instance du problème, produit une solution. La *taille* d'un problème reflète le nombre de données nécessaires pour décrire une instance.

Un algorithme est dit *polynomial* si le nombre d'opérations élémentaires nécessaire pour résoudre une instance de taille n est borné par une fonction polynomiale en n. Un problème est dit appartenir à la *classe* P s'il existe un algorithme polynomial pour le résoudre. Les problèmes de la classe P sont dits *faciles*.

Un *problème de décision* est un problème ayant deux réponses possibles : *oui* ou *non*. Soient $\mathcal{P}$ un problème de décision et $\mathcal{I}$ les instances de ce problème pour lesquelles la réponse est oui. $\mathcal{P}$ appartient à la *classe NP* (Nondeterministic Polynomial) s'il existe un algorithme polynomial qui permet de vérifier que la réponse est oui pour toute instance de $\mathcal{I}$. Il est clair que la classe P est contenue dans la classe NP (voir figure 1.1). La différence entre P et NP n'a pas été prouvée, mais la conjecture est considérée comme hautement probable.

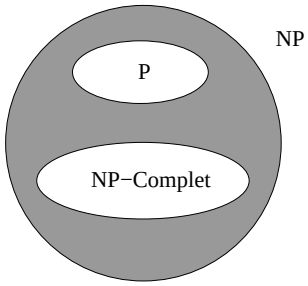


FIG. 1.1 – Relations entre P, NP et NP-Complet

Nous distinguons également dans la classe NP, la classe des problèmes NP-complets. La NP-complétude s'appuie sur la notion de réduction polynomiale. Un problème de décision P_1 se réduit polynomialement en un problème de décision P_2 s'il existe une fonction polynomiale f telle que, pour toute instance I de P_1, la réponse est oui si et seulement si la réponse de $f(I)$ pour P_2 est oui. Nous noterons alors $P_1 \alpha P_2$. Un problème $\mathcal{P}$ est *NP-complet*, s'il appartient à la classe NP et s'il existe un problème Q connu comme étant NP-complet tels que $Q \alpha P$. Cook a été le premier à montrer la NP-complétude d'un problème, celui de la satisfiabilité [17].

A tout problème d'optimisation combinatoire peut être associé un problème de décision. Tout problème d'optimisation combinatoire dont le problème de décision associé est NP-complet est dit *NP-difficile*.

1.1.3 Approche polyédrale et méthode de coupes et branchements

Un grand nombre de problèmes issus du monde réel peuvent être formulés sous la forme de problèmes d'optimisation combinatoire. A première vue, leur résolution semble facile. Nous pouvons par exemple faire une énumération de toutes les solutions possibles et choisir la meilleure. Cependant, le nombre de ces solutions devient vite exponentiel et la méthode énumérative n'est plus possible. D'où la nécessité de développer des techniques permettant de résoudre ces problèmes plus efficacement. Une des méthodes les plus puissantes est la méthode dite *polyédrale* qui a été introduite par Edmonds en 1965 [29] pour le problème du couplage. Nous allons à présent présenter brièvement cette approche. Pour plus de détails, voir par exemple [78, 65].

1.1.3.1 Définitions

Nous allons tout d'abord rappeler quelques définitions et propriétés liées à la théorie des polyèdres.

Soit $n \in \mathbb{N}$. Le symbole $\mathbb{R}^n$ représente l'ensemble des vecteurs ayant n composantes réelles. L'ensemble des nombres réels non négatifs sera noté $\mathbb{R}_+$.

Étant donné un ensemble de points $x^1, \ldots, x^m \in \mathbb{R}^n$, un point $x \in \mathbb{R}^n$ est dit *combinaison linéaire* de $x^1, \ldots, x^m$ s'il existe $\lambda_1, \ldots, \lambda_m \in \mathbb{R}$ tels que

$$x = \sum_{i=1}^{m} \lambda_i x^i.$$

Si de plus

$$\sum_{i=1}^{m} \lambda_i = 1$$

$$\text{(resp. } \lambda_i \in \mathbb{R}_+ \text{ pour i=1,...,m et } \sum_{i=1}^{m} \lambda_i = 1),$$

x est dit *combinaison affine* (resp. *combinaison convexe*) de ces points.

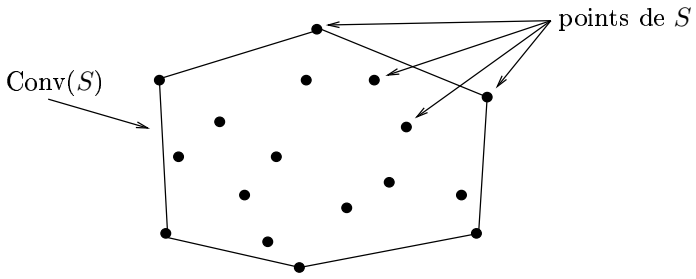


FIG. 1.2 – Enveloppe convexe

Étant donné un ensemble S de points $x^1, \ldots, x^m \in \mathbb{R}^n$, l'*enveloppe convexe* de $x^1, \ldots, x^m$ est

$$conv(S) = \{x \in \mathbb{R}^n \mid x \text{ combinaison convexe de } x^1, \ldots, x^m\}.$$

La figure 1.2 illustre cette notion.

Des points $x^1, \ldots, x^m \in \mathbb{R}^n$ sont *linéairement indépendants* (resp. *affinement indépendants*) si le système

$$\sum_{i=1}^{m} \lambda_i x^i = 0$$

$$(\text{resp. } \sum_{i=1}^{m} \lambda_i x^i = 0 \text{ et } \sum_{i=1}^{m} \lambda_i = 0)$$

admet une solution unique $\lambda_i = 0$ pour $i = 1, \ldots, m$.

Un *polyèdre* P est l'ensemble des solutions d'un système linéaire $Ax \leq b$ c'est-à-dire $P = \{x \in I\!R^n \mid Ax \leq b\}$, où A est une matrice à m lignes et n colonnes, et b un vecteur à m composantes. Un *polytope* est un polyèdre borné.

Un polyèdre $P \subseteq I\!R^n$ est dit de *dimension* p si le nombre maximum de points de P affinement indépendants est $p+1$. Nous noterons alors $dim(P) = p$. Un polyèdre P de $I\!R^n$ est *de pleine dimension* si $dim(P) = n$.

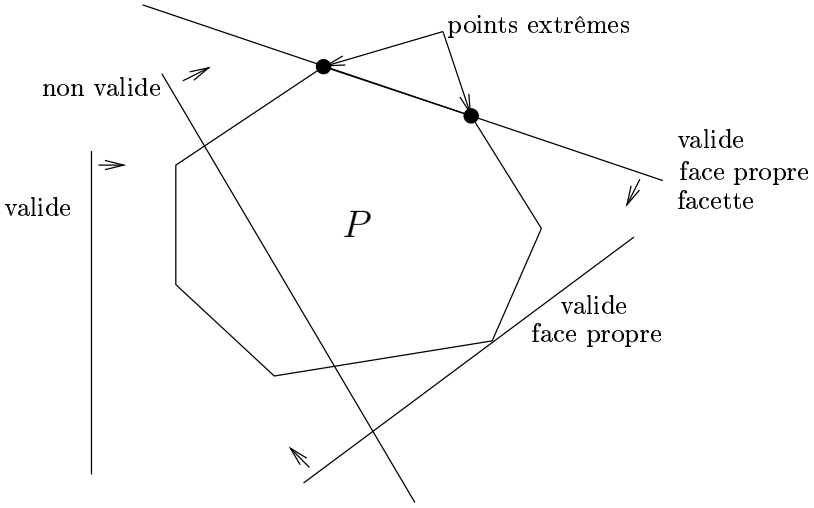


FIG. 1.3 – Contraintes valides, faces et facettes

Si a et x sont deux vecteurs colonne à n composantes, on note par $a^T x$, le produit scalaire de a et x. Une contrainte $a^T x \leqslant \alpha$ est dite *valide* pour un polyèdre P de $I\!R^n$ si elle est vérifiée pour toute solution de P. Soit $x^* \in I\!R^n$. L'inégalité $a^T x \leqslant \alpha$ est dite *serrée* pour x^* si $a^T x^* = \alpha$. Une contrainte est dite *violée* par x^* si x^* ne satisfait pas la contrainte (voir figure 1.3).

Étant donné un polyèdre P et une contrainte $a^T x \leqslant \alpha$ valide pour P, le sous-ensemble $F = \{x \in P \mid a^T x = \alpha\}$ est appelé *face* de P *définie* par $a^T x \leqslant \alpha$. De plus nous avons $dim(F) \leqslant dim(P)$. Une face F est dite *propre* si $F \neq P$ et $F \neq \emptyset$. Une face propre F est une *facette* de P si $dim(F) = dim(P) - 1$.

Un *point extrême* d'un polyèdre P est une face de P de dimension 0. Il est facile de voir qu'un point $x \in I\!R^n$ est un point extrême d'un polyèdre P, s'il ne peut pas être écrit comme combinaison convexe d'autres points de P.

1.1.3.2 Approche polyédrale

Soient $\mathcal{P}$ un problème d'optimisation combinatoire, $\mathcal{S}$ l'ensemble des solutions de $\mathcal{P}$, E l'ensemble de base de $\mathcal{P}$ et c la fonction poids associée aux variables du problème. Le problème $\mathcal{P}$ s'écrit donc $max\{cx \mid x \in \mathcal{S}\}$.

Si T est un sous-ensemble de E, le vecteur $x^T \in I\!R^E$ (ayant $|E|$ composantes associées aux éléments de E) tel que

$$x^T(e) = \begin{cases} 1 & \text{si } e \in T, \\ 0 & \text{sinon,} \end{cases}$$

est appelé *vecteur d'incidence* de T. Le polyèdre

$$P(\mathcal{S}) = conv\{x^S \mid S \in \mathcal{S}\}$$

est appelé *polyèdre des solutions* de $\mathcal{P}$ (ou *polyèdre associé* à $\mathcal{P}$).

Le problème $\mathcal{P}$ est donc équivalent au programme linéaire $max\{cx \mid x \in P(\mathcal{S})\}$. $P(\mathcal{S})$ peut être caractérisé par un ensemble de contraintes linéaires où chaque contrainte définit une facette. Si on peut décrire entièrement le polyèdre $P(\mathcal{S})$ par un système d'inégalités linéaires, le problème $\mathcal{P}$ se ramène donc à la résolution d'un programme linéaire.

L'*approche polyédrale* consiste à ramener la résolution du problème à la résolution d'un (ou d'une séquence) de programmes linéaires. Cette transformation nécessite, par conséquent, une étude approfondie du polyèdre associé au problème. Néanmoins, la caractérisation complète de ce polyèdre est généralement difficile à établir (voire impossible si le problème est NP-difficile). De plus, le nombre de contraintes nécessaires pour décrire le polyèdre, est souvent exponentiel. Cependant, en utilisant une méthode de coupes et branchements (Branch-and-Cut method), une description partielle peut être suffisante pour résoudre le problème à l'optimum. Cette méthode combine la méthode de génération de nouvelles contraintes (Cutting plane method) et la méthode de *séparations et évaluations* (Branch & Bound method). Dans la suite, nous discutons de cette méthode.

1.1.3.3 Méthode de coupes et branchements

La méthode de coupes et branchements pour un problème d'optimisation combinatoire est basée sur le problème dit *de séparation*. Soit P un polyèdre dans $I\!R^n$. Le

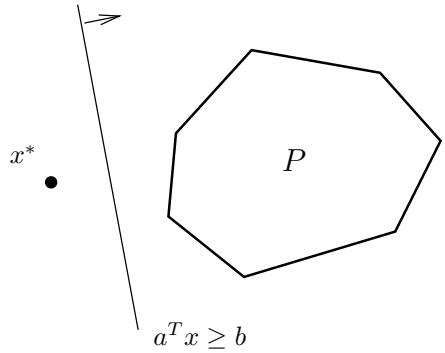


FIG. 1.4 – Hyperplan séparant x^* et P

problème de séparation associé à P consiste à vérifier pour un point $x^* \in \mathbb{R}^n$ si x^* appartient à P, et dans le cas contraire, à trouver une contrainte $a^T x \leqslant b$ valide pour P et violée par x^*. Dans ce deuxième cas, l'hyperplan $a^T x = b$ sépare P et x^* (voir la figure 1.4).

Une des grandes avancées de l'optimisation combinatoire est le parallèle entre séparation et optimisation. Grötschel, Lovász et Schrijver [40] ont montré qu'un problème d'optimisation sur un polyèdre P est polynomial si et seulement si le problème de séparation associé à P peut être résolu en temps polynomial.

Considérons un problème d'optimisation combinatoire $\mathcal{P}$ de la forme $max\{cx \mid Ax \leqslant b, x \text{ entier}\}$ et soit P le polyèdre associé à $\mathcal{P}$. Supposons que l'on dispose d'un système $\bar{A}x \leq \bar{b}$ de contraintes valides pour $\mathcal{P}$ et qui contient comme sous-système les contraintes de base du problème. La méthode de coupes et branchements commence par résoudre un programme linéaire $P_1 = max\{cx \mid \bar{A}_1 x \leq \bar{b}_1\}$ où $\bar{A}_1 x \leq \bar{b}_1$ est un sous-système de $\bar{A}x \leq \bar{b}$ contenant un nombre raisonnable de contraintes. Si la solution optimale disons x_1 de P_1 est entière et solution de $Ax \leq b$, alors elle est optimale pour $\mathcal{P}$. Sinon, on résout le problème de séparation associé à $\bar{A}x \leq \bar{b}$ et x_1. Si l'on trouve une contrainte $a_1 x \leq \alpha_1$ violée par x_1, elle est rajoutée au système $\bar{A}_1 x \leq \bar{b}_1$ et nous obtenons un nouveau programme linéaire $P_2 = max\{cx \mid \bar{A}_1 x \leq \bar{b}_1, a_1 x \leq \alpha_1\}$. On résout P_2. Si la solution optimale x_2 de P_2 est entière et solution de $Ax \leq b$, comme pour x_1 on résout le problème de séparation associé à $\bar{A}x \leq \bar{b}$ et x_2. Si $a_2 x \leq \alpha_2$ est une contrainte violée par x_2, elle est rajoutée au système $\bar{A}_2 x \leq \bar{b}_2$ et ainsi de suite. En continuant ce processus appelé *phase de coupe*, nous pouvons trouver, soit une solution optimale pour $\mathcal{P}$, soit une solution x^* qui soit fractionnaire et pour laquelle nous ne pouvons pas générer de contrainte violée. Dans ce cas, nous commençons une phase dite de *branchement* qui consiste à construire un arbre de Branch & Bound. Nous choisissons une variable fractionnaire x_i^*. Nous résolvons deux nouveaux programmes

linéaires (nouveaux sommets de l'arbre) en ajoutant soit la contrainte $x_i = 0$, soit $x_i = 1$ au programme linéaire obtenu à la fin de la phase de coupe. Pour chacun de ces nouveaux sommets, nous appliquons une procédure de coupes. Si une solution optimale de $\mathcal{P}$ n'a pas été trouvée, nous sélectionnons une feuille de l'arbre et nous recommençons une phase de branchement.

Cette approche est maintenant largement utilisée pour les problèmes d'optimisation combinatoire difficiles. La méthode de coupes et branchements peut être utilisée lorsque le nombre de variables du problème n'est pas très grand (polynomial). Par contre, si le problème comporte un nombre exponentiel de variables, la méthode de génération de colonnes et branchement est plus appropriée pour approcher le problème. La section suivante présente les grandes lignes de cette méthode.

1.1.4 Méthode de génération de colonnes et branchements

Une des méthodes classiques en programmation mathématique est la méthode dite de *génération de colonnes*. Celle-ci permet de résoudre efficacement des programmes linéaires de grande taille en utilisant un nombre réduit de variables. Elle est apparue dans les années 1960 pour résoudre des problèmes dont les données ne pouvaient pas être stockées dans les ordinateurs de l'époque. Dantzig et Wolfe [23] étaient les premiers à introduire cette technique dans le cadre de leur méthode de décomposition. En effet, la technique de génération de colonnes peut être utilisée soit sur des problèmes pouvant être traités par la *décomposition de Dantzig-Wolfe*, soit pour des problèmes dont la modélisation comporte un nombre considérable de variables. Gilmore et Gomory [36, 37] ont utilisé cette approche pour un problème de découpe faisant partie de cette deuxième catégorie de problèmes.

Le principe de la génération de colonnes consiste à résoudre itérativement une séquence de programmes linéaires ayant chacun un nombre limité de colonnes (ou variables). On débute la résolution du programme linéaire en résolvant un programme linéaire réduit ne contenant qu'un ensemble plus petit de variables représentant une base réalisable. Puis à chaque itération, on résout un problème appelé *problème auxiliaire* (pricing problem) qui consiste à déterminer les variables devant entrer en base. Ces colonnes sont celles dont le coût réduit est strictement négatif. Le coût réduit associé à une colonne est calculé à partir des variables duales du problème. Le nouveau problème ainsi obtenu est à son tour résolu et un nouveau problème auxiliaire permet de trouver de nouvelles variables devant entrer en base. On répète ce processus jusqu'à ce que le problème auxiliaire ne permette plus de trouver de variable devant être rajoutée, la solution optimale du problème restreint est alors optimale pour le problème

de départ. Le problème auxiliaire est un problème d'optimisation qui peut être difficile. On peut alors utiliser des heuristiques appropriées pour résoudre le problème auxiliaire. Néanmoins, ce dernier devra être résolu du moins à la dernière itération de manière exacte pour prouver l'optimalité de la solution. Pour plus de détails sur la génération de colonnes, voir [57, 59, 82].

Parmi tous les problèmes pouvant être traités à l'aide de la génération de colonnes, on compte les problèmes de multiflot qui ont été très étudiés dans la littérature [3, 10, 52].

Dans le but de résoudre un programme linéaire en nombres entiers, la technique de génération de colonnes peut être couplée à une phase de branchement similaire à celle décrite précédemment. On parle alors d'algorithme de *génération de colonnes et branchements* (Branch-and-Price algorithm). Le branchement intervient quand le problème auxilliaire ne permet plus de trouver de variables devant entrer en base et la solution du programme linéaire n'est pas entière. On peut de plus coupler cet algorithme avec un algorithme de coupes et branchements. On parle alors d'algorithme de *génération de colonnes, coupes et branchements*. Barnhart et al. [12] présentent deux problèmes de très grande taille résolus à l'aide de cette méthode. De même, Barnhart, Hane et Vance [11] ont utilisé cette même technique pour résoudre le problème du multiflot entier.

1.1.5 Notations et définitions de la théorie des graphes

Un *graphe non orienté* est noté $G = (V, E)$ où V est l'ensemble de *sommets* et E l'ensemble des *arêtes*. Si e est une arête reliant deux sommets u et v, alors u et v seront appelés les *extrémités* de e, et nous écrirons $e = uv$ ou $e = \{u, v\}$. Si u est une extrémité de e, alors u (resp. e) est dit *incident* à e (resp. u). De même, deux sommets u et v formant une arête sont dits *adjacents*.

Un *graphe orienté* est noté $D = (V, A)$ où V est l'ensemble de *sommets* et A l'ensemble des *arcs*. Si a est un arc allant du sommet u au sommet v, alors u et v seront appelés respectivement le *sommet origine* et le *sommet destination* de a, et nous écrirons $a = (u, v)$.

Soit $G = (V, E)$ un graphe non orienté. Si $F \subseteq E$ est un sous-ensemble d'arêtes, alors $V(F)$ représente l'ensemble des extrémités des arêtes de F. Si $W \subseteq V$ est un sous-ensemble de sommets, alors $E(W)$ denote l'ensemble des arêtes ayant leurs deux extrémités dans W.

Soit $W \subseteq V$, $H = (W, E(W))$ est dit sous-graphe de G *induit* par W et sera noté par $G(W)$.

Soient u et v deux sommets de V. Une *chaîne* (resp. *un chemin*) P entre u et v (resp. allant de u à v) est une séquence alternée de sommets et d'arêtes (resp. d'arcs) $(v_0, e_1, v_1, e_2, v_2, \ldots, v_{k-1}, e_k, v_k)$ (resp. $(v_0, a_1, v_1, a_2, v_2, \ldots, v_{k-1}, a_k, v_k)$) où $v_0 = u$, $v_k = v$, $e_i = v_{i-1}v_i$ (resp. $a_i = (v_{i-1}, v_i)$) pour $i = 1, \ldots, k$ et $v_0, \ldots, v_k$ sont des sommets distincts de V. S'il n'y a pas de confusion possible, on peut également noter P par sa séquence d'arêtes $(e_1, \ldots, e_p)$ (resp. sa séquence d'arcs $(a_1, \ldots, a_p)$) ou sa séquence de sommets $(v_1, \ldots, v_{p+1})$. Deux chaînes entre u et v sont dites *arête-disjointes* (resp. *sommet-disjointes*) s'il n'existe pas d'arête (resp. de sommet différent de u et v) apparaissant dans les deux chaînes.

Un graphe G est *connexe* si, pour toute paire de sommets u, v de V, il existe au moins une chaîne entre u et v.

Un *arbre* d'un graphe $G = (V, E)$ est un sous-graphe connexe et sans cycle.

Si $F \subset E$, on notera par $G\backslash F$ le graphe obtenu à partir de G en supprimant les arêtes de F. Si F est réduit à une seule arête e, nous écrirons $G\backslash e$ au lieu de $G\backslash\{e\}$.

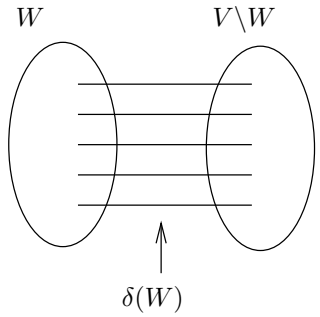


FIG. 1.5 – Une coupe $\delta(W)$

Soit $W \subseteq V$, $W \neq \emptyset$, un sous-ensemble de sommets de V. L'ensemble des arêtes ayant une extrémité dans W et l'autre dans $V\backslash W$ est appelé *coupe* et noté $\delta(W)$ (voir figure 1.5). En posant $\overline{W} = V\backslash W$, nous avons $\delta(W) = \delta(\overline{W})$. Si W est réduit à un seul sommet v, nous écrirons $\delta(v)$ au lieu de $\delta(\{v\})$.

Etant donnés W et W' deux sous-ensembles disjoints de V, alors $\delta_G(W, W')$ représente l'ensemble des arêtes de G qui ont une extrémité dans W et l'autre dans W'. Lorsque W et W' sont réduits à un seul sommet, on notera $[u, u']$ à la place de $\delta_G(\{u\}, \{u'\})$.

Si $V_1, \ldots, V_p$, $(p \geq 2)$ est une partition de V, alors $\delta(V_1, \ldots, V_p)$ représente l'ensemble des arêtes ayant leurs extrémités dans des éléments différents de la partition (*i.e* $e = uv$ telles que $u \in V_i$, $v \in V_j$ et $i \neq j$).

1.2 État de l'art

1.2.1 Conception de réseaux

Dans cette section, nous discutons de plusieurs problèmes ayant trait à la conception de réseaux fiables. Nous présentons tout d'abord un problème général de conception issu du domaine des télécommunications et nous étudions une formulation mathématique de ce problème. De nombreuses variantes de ce modèle ont été traitées dans la littérature. Par la suite, nous étudions différents modèles découlant de ce problème général suivant le type de contraintes de fiabilité imposées. Tous ces problèmes se sont avérés très différents et nécessitant une étude particulière. Parmi ces problèmes, on compte encore de nombreux problèmes ouverts.

1.2.1.1 Le problème général de conception de réseaux fiables

Soit $G = (V, E)$ un graphe fini, non orienté, où V représente les nœuds (ou centres de communication) et E les liaisons possibles entre ces nœuds. Le graphe G peut contenir des arêtes parallèles, mais pas de boucle. Chaque arête $e \in E$ est munie d'un coût $c(e)$ représentant le coût d'installation de la liaison directe. Le coût total d'un sous-graphe $H = (U, F)$ de G est $c(F) = \sum_{e \in F} c(e)$.

A chaque sommet $v \in V$ est associé un entier positif $r(v)$, appelé type de connexité de v, qui représente le nombre minimum d'arêtes qui doivent lier v au reste du réseau. Le vecteur $r = (r(v) \mid v \in V)$ est appelé *vecteur types de connexité* associé aux sommets ou *vecteur types de sommets*. Le concept de *type de connexité* a été introduit par Grötschel et Monma [41].

Un sous-graphe $H = (U, F)$ de G vérifie les *conditions de fiabilité en arêtes* (resp. *en sommets*), si pour toute paire de nœuds $u, v \in V$, il existe au moins

$$r(u, v) = min\ \{r(u), r(v)\}$$

chaînes arête-disjointes (resp. sommet-disjointes) entre u et v. Les conditions de fiabilité en arêtes (resp. en sommets) seront aussi appelées *conditions d'arête-connexité* (resp. *conditions de sommet-connexité*). Ces conditions assurent l'écoulement du trafic entre u et v s'il y a au plus $r(u, v) - 1$ liaisons ou nœuds tombant en panne dans des chemins entre u et v.

Soient un graphe $G = (V, E)$, un vecteur type de sommets $r = (r(v) \mid v \in V)$ et une fonction coût $c : E \rightarrow I\!R$. Nous supposons, sans perte de généralité, qu'il existe au moins deux sommets ayant un degré de connexité k, où $k = max\ \{r(v) \mid v \in V\}$. Le *problème général de conception de réseaux*, introduit par Winter [83] consiste à déterminer un sous-graphe de G de coût minimum vérifiant les conditions d'arête-connexité (resp. de sommet-connexité) (voir aussi [54]). Ce problème sera noté dans la suite par kECON (resp. kNCON).

En établissant une relation entre les chaînes et les coupes, le théorème de Menger [67] suivant permet de donner une formulation mathématique pour le problème kECON.

Théorème 1.1 *Dans un graphe G, il n'existe pas de coupe de cardinalité inférieure ou égale à $k-1$ séparant deux sommets donnés u et v, si et seulement s'il existe au moins k chaînes arête-disjointes entre u et v.*

Soient

$$\begin{array}{rcll} r(W) & = & max\ \{r(u) \mid u \in W\} & \forall\, W \subseteq V, \\ con(W) & = & max\ \{r(u,v) \mid u \in W, v \in V \backslash W\} & \\ & = & min\ \{r(W), r(V \backslash W)\} & \forall\, W \subseteq V,\ \emptyset \neq W \neq V. \end{array}$$

Soit $x \in I\!R^{|E|}$ un vecteur tel que $x(e) = 1$ si l'arête e appartient à la solution et 0 sinon. Par le théorème 1.1 de Menger, le problème kECON est alors équivalent au programme linéaire en nombres entiers suivant :

$$\text{Minimiser} \sum_{e \in E} c(e)x(e)$$

$$\begin{array}{lll} x(\delta(W)) \geq con(W) & \forall\, W \subseteq V,\ \emptyset \neq W \neq V, & (1.1) \\ x(e) \geq 0 & \forall\, e \in E, & (1.2) \\ x(e) \leq 1 & \forall\, e \in E, & (1.3) \\ x(e) \text{ entier} & \forall\, e \in E. & (1.4) \end{array}$$

Les inégalités (1.1) sont appelées *contraintes de coupe*, les inégalités (1.2) et (1.3) sont les *contraintes triviales* et les inégalités (1.4) sont les *contraintes d'intégrité*. Les contraintes de coupe expriment le fait qu'une solution du problème doit posséder au moins $con(W)$ arêtes de la coupe $\delta(W)$.

Soit $G = (V, E)$. Soit $Z \subseteq V$ un sous-ensemble de V. Si $W \subseteq V \backslash Z$, alors on pose $con_{G \backslash Z}(W) = min(r(W), r(V \backslash (Z \cup W)))$. Il n'est pas difficile de voir que les contraintes

$$x(\delta_{G \backslash Z}(W)) \geq con_{G \backslash Z}(W) - |Z| \qquad \forall\, Z \subseteq V,\ \emptyset \neq Z \neq V \text{ et } \forall\, W \subseteq V \backslash Z, \qquad (1.5)$$

sont valides pour le polytope kNCON(G). En ajoutant les contraintes (1.5) au modèle ci-dessus, nous obtenons une formulation du problème kNCON. Les contraintes (1.5) sont également appelées *contraintes de coupe*.

Le polytope des solutions de kECON (resp. kNCON) est le polyèdre

$$k\text{ECON}(G) = conv\{x \in I\!R^E \mid x \text{ satisfait } (1.1) - (1.4)\}$$

$$(\text{resp. } k\text{NCON}(G) = conv\{x \in I\!R^E \mid x \text{ satisfait } (1.1) - (1.5)\}).$$

On remarque que kNCON(G) $\subseteq$ kECON(G).

Dans [41], Grötschel et Monma ont décrit plusieurs familles de contraintes valides pour kECON(G) et kNCON(G). Grötschel, Monma et Stoer [42, 43, 44] ont identifié des classes de facettes et développé des algorithmes de coupes, basés sur une description partielle des polytopes kECON(G) et kNCON(G). Une synthèse complète sur ce modèle est présentée dans [45] et [79].

Magnanti et Raghavan [62] présentent une formulation en termes de flots de ce problème très général et considèrent les versions orientée et non-orientée du problème.

De nombreux problèmes d'optimisation combinatoire, comme le problème de l'arbre de Steiner, le problème du sous-graphe k-arête connexe, connus comme étant NP-difficiles [35], peuvent être vus comme des cas particuliers de ce modèle. Le problème de conception de réseaux fiables est donc NP-difficile dans le cas général. Néanmoins, le problème kECON (kNCON) peut être résolu en temps polynomial pour des conditions de connexité particulières. Par exemple :

- si $r(v) = 1 \quad \forall v \in V$, nous nous ramenons au problème de l'arbre couvrant de poids minimum. Les algorithmes polynomiaux les plus connus pour résoudre ce problème sont ceux de Kruskal [56] et Prim [73].
- si $r(v) = 1$ pour exactement deux sommets de V et $r(v) = 0$ pour les autres et si nous avons des coûts positifs ou nuls, nous obtenons le problème classique du plus court chemin qui peut être résolu en temps polynomial avec l'algorithme de Dijkstra [27].

1.2.1.2 Le problème du sous-graphe k-arête connexe

Dans cette section, nous considérons des problèmes pour lesquels tous les sommets du graphe ont le même type de connexité. Si le type de connexité est uniforme tel que $r(v) = k \quad \forall v \in V$, nous avons le problème dit du sous-graphe k-arête connexe.

Définition 1.2 *Un graphe $G = (V, E)$ est dit* k-arête connexe *(resp.* k-sommet connexe*) si entre deux sommets quelconques de V, il existe au moins k chaînes arête-disjointes (resp. sommet-disjointes).*

Le *problème du sous-graphe k-arête connexe* (kECSP, k-edge connected spanning subgraph problem) consiste à trouver un sous-graphe k-arête connexe couvrant les nœuds de V et qui soit de coût minimum.

D'après le théorème 1.1, le problème kECSP est équivalent au programme linéaire en nombres entiers suivant :

$$\text{Minimiser} \sum_{e \in E} c(e)x(e)$$

$$\begin{aligned}
x(\delta(W)) &\geq k \quad \forall\, W \subseteq V, \emptyset \neq W \neq V, && (1.6)\\
x(e) &\geq 0 \qquad \forall\, e \in E, && (1.7)\\
x(e) &\leq 1 \qquad \forall\, e \in E, && (1.8)\\
x(e)&\text{entier} \qquad \forall\, e \in E. && (1.9)
\end{aligned}$$

Le polytope des solutions de kECSP est

$$k\text{ECP}(G) = conv\{x^F \in I\!R^E \mid (V, F) \text{ est un sous-graphe } k\text{-arête connexe de } G\}.$$

Le cas où $k = 2$ a été très étudié dans la littérature. Le problème du sous-graphe 2-arête connexe est noté TECSP et le polytope des solutions est noté TECP(G).

Le problème du voyageur de commerce consiste à déterminer un cycle Hamiltonien (c'est-à-dire un cycle passant une et une seule fois par chaque sommet du graphe) et qui soit de coût minimum. Ce problème est très lié au problème du sous-graphe 2-arête connexe. En effet, une solution du premier est une solution du deuxième et donc le problème du sous-graphe 2-arête connexe peut être vu comme une relaxation du problème du voyageur de commerce. De plus, le problème consistant à déterminer si un graphe contient ou non un cycle hamiltonien (c'est-à-dire le problème consistant à déterminer si le problème du voyageur de commerce admet une solution ou non) peut être ramené au TECSP [31]. Le TECSP est donc NP-difficile [35].

Définition 1.3 *Un graphe est dit* série-parallèle *s'il peut être obtenu à partir d'un graphe constitué de deux sommets liés par une seule arête, en appliquant d'une manière récursive les deux opérations suivantes :*

– *dupliquer une arête (*i.e. *ajouter une arête parallèle joignant les mêmes sommets),*
– *subdiviser une arête (*i.e. *insérer un nouveau sommet de degré deux).*

Dans [63], Mahjoub a donné une caractérisation complète du polytope TECP(G) dans la classe des graphes série-parallèles. Il a également introduit une large classe d'inégalités valides pour TECSP(G).

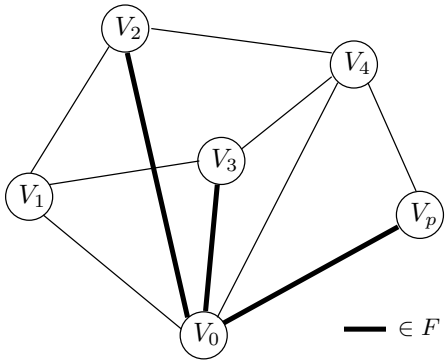


FIG. 1.6 – F-partition

Soient $G = (V, E)$ un graphe, $V_0, V_1, \ldots, V_p$ une partition de V et $F \subseteq \delta(V_0)$ avec $|F| = 2t+1 \quad (t \geq 0)$ (voir figure 1.6). Soit $\Delta = \delta(V_0, V_1, \ldots, V_p) \backslash F$ où $\delta(V_0, V_1, \ldots, V_p)$ représente l'ensemble des arêtes ayant leurs extrémités dans des membres différents de la partition. Mahjoub [63] a montré que l'inégalité

$$x(\Delta) \geq p - t \tag{1.10}$$

est valide pour TECSP(G). Il a également donné des conditions suffisantes pour que ces inégalités définissent des facettes pour TECSP(G). Les contraintes (1.10) sont appelées *contraintes de F-partition.*

Dans [64], Mahjoub a étudié les graphes pour lesquels le polytope (G) est complètement décrit par les contraintes triviales et les contraintes de coupe (1.6) avec $k = 2$.

Barahona et Mahjoub [9] ont étudié le problème TECSP dans la classe des graphes de Halin. (Un graphe de Halin est un graphe $G = (V, T \cup C)$ où T est un arbre sans sommet de degré 2 et C est un cycle dont les sommets sont les sommets pendants de T.) Ils ont montré que les inégalités de F-partition (1.10) associées aux inégalités de coupe (1.6) avec $k = 2$ et aux inégalités triviales (1.7), (1.8) suffisent pour décrire le polytope TECSP(G) dans cette classe de graphes. De plus, les inégalités de F-partition, dans ce cas, définissent toutes des facettes.

Le problème du sous-graphe 2-arête connexe a également été étudié en ajoutant des contraintes de bornes, soit sur la longueur des cycles, soit sur la longueur des chemins [21, 34, 48, 49, 70].

Barahona et Mahjoub [9] ont étudié le problème du sous-graphe 2-sommet connexe (TNCSP). Ce problème consiste à déterminer un sous-graphe 2-sommet connexe de poids minimum couvrant V. Mahjoub et Nocq ont également étudié ce problème [66].

Dans la pratique, nous avons souvent besoin d'imposer de fortes conditions de connexité, c'est-à-dire des types de connexité strictement supérieurs à 2.

Chopra [16] a étudié une relaxation du kECSP. Il a considéré le problème kECSP où chaque arête est autorisée à être utilisée plus d'une fois. Le problème est donc de déterminer un vecteur entier $x \in I\!N^E$ tel que le graphe $(V, E(x))$ où $E(x)$ est l'ensemble d'arêtes obtenu en remplaçant chaque arête $e \in E$ par $x(e)$ arêtes, est k-arête connexe et de poids minimum. Le polytope associé à ce problème est le polyèdre

$$P_k(G) = conv\{x \in I\!N^E \mid (V, E(x)) \text{ } k\text{-arête connexe}\}.$$

Chopra a introduit une nouvelle famille de contraintes valides pour ce problème quand le graphe est outerplanaire. (Un graphe est outerplanaire s'il est planaire et s'il peut être représenté dans le plan de telle manière que tous les sommets soient sur la face extérieure. Les graphes outerplanaires sont série-parallèles.) Soit $V_1, \ldots, V_p$ $(p \geq 2)$ une partition de V telle que le sous-graphe induit $G[V_i]$ soit connexe pour $i = 1, \ldots, p$. Soit l'inégalité

$$x(\delta(V_1, \ldots, V_p)) \geq \left\lceil \frac{k}{2} \right\rceil p - 1. \tag{1.11}$$

Chopra [16] a montré que les contraintes (1.11) sont valides pour $P_k(G)$. Les contraintes (1.11) sont appelées *inégalités de partition outerplanaire.*

Chopra a montré que dans le cas où G est outerplanaire et k est impair, $P_k(G)$ est complètement caractérisé par les contraintes triviales (1.7) et les contraintes de partition outerplanaire (1.11).

Quand G est un graphe série-parallèle et k est pair, Cornuéjols et al. [18] ont montré que $P_k(G)$ est défini entièrement par les contraintes de coupe (1.6) et les contraintes de non négativité (1.7).

Didi Biha et Mahjoub [25] ont étendu la validité des contraintes de partition outerplanaire aux graphes série-parallèles et les ont appelées *contraintes de SP-partition.* Ils ont montré que quand G est un graphe série-parallèle connexe et k est impair, $P_k(G)$

est complètement décrit par les contraintes de non négativité (1.7) et les contraintes de SP-partition (1.11), ce qui généralise le résultat de Chopra [16].

Si G est un graphe série-parallèle, Didi Biha et Mahjoub [25] ont montré que le polytope kECP(G) est décrit par les inégalités triviales (1.7), (1.8) et de coupe (1.6) (resp. les inégalités triviales (1.7), (1.8) et de SP-partition (1.11)) si k est pair (resp. k est impair) (voir aussi [24] et [26]).

Nous allons à présent considérer des problèmes d'optimisation dans les réseaux non plus d'un point de vue topologie et sécurisation mais du côté dimensionnement de réseaux.

1.2.2 Dimensionnement de réseaux

Les problèmes de dimensionnement de réseaux ont de nombreuses applications pratique, comme par exemple pour les réseaux de télécommunications. Ils consistent à déterminer les capacités à installer sur les liaisons d'un réseau pour que celui-ci puisse écouler la demande transitant sur ce réseau. On suppose que des demandes entre des paires d'origines-destinations d'un certain réseau sont données. Les capacités sur les arêtes sont limitées, elles entraînent, en conséquence, un coût d'installation, au même titre que les liaisons. Le problème est donc de déterminer un réseau de telle sorte que les demandes entre les origines-destinations puissent être écoulées et le coût total d'installation soit minimum. Nous présentons ici plusieurs problèmes de dimensionnement.

1.2.2.1 Le problème NLP

Pour le *problème de chargement de réseau* (NLP, Network Loading Problem), la demande point-à-point entre les nœuds du réseau se heurte à l'installation de capacités possibles. La capacité doit être un multiple d'une taille fixe C. Le problème consiste à déterminer la capacité à installer sur chaque arête pour satisfaire la demande avec un coût minimum. Ce problème est NP-difficile [68].

Soit $G = (V, E)$ un graphe fini, non orienté, où V représente les nœuds du réseau et E l'ensemble des arêtes. On note Q l'ensemble des demandes. Soient $k \in Q$, on note $O(k)$ l'origine de la demande, $D(k)$ la destination et u_k le volume. Soit b_{ij} le coût d'installation d'une seule capacité C sur l'arête $\{i, j\}$. Notons par f_{ij}^k le flot du produit k sur l'arête $\{i, j\}$ de i vers j, et par y_{ij} le nombre de capacités de taille C installées sur l'arête $\{i, j\}$.

Le problème NLP est équivalent au programme linéaire mixte suivant :

$$\text{Minimiser} \sum_{\{i,j\}\in E} b_{ij} y_{ij}$$

$$\sum_{j\in V} f_{ij}^k - \sum_{j\in V} f_{ji}^k = \begin{cases} -u_k & \text{si } i = O(k), \\ u_k & \text{si } i = D(k), \\ 0 & \text{sinon,} \end{cases} \qquad \forall\, i \in V, \forall k \in Q, \tag{1.12}$$

$$\sum_{k\in Q} (f_{ij}^k + f_{ji}^k) \leq C y_{ij} \qquad \forall\, i, j \in E, \tag{1.13}$$

$$y_{ij} \geq 0 \qquad \forall\, i, j \in E, \tag{1.14}$$

$$y_{ij} \text{ entier} \qquad \forall\, i, j \in E, \tag{1.15}$$

$$f_{ij}^k, f_{ji}^k \geq 0 \qquad \forall\, i, j \in E, \forall\, k \in Q. \tag{1.16}$$

Les inégalités (1.12) sont appelées *contraintes de conservation de flot*. Les inégalités (1.13) sont les *contraintes de capacité*. Les inégalités (1.14) et (1.16) sont les *contraintes de non négativité* et les inégalités (1.15) sont les *contraintes d'intégrité*.

Dans [60], Magnanti, Mirchandani et Vachani ont étudié deux sous-problèmes du NLP. Le premier consiste à ne considérer qu'une seule arête du graphe. Ils ont introduit de nouvelles inégalités appelées contraintes de capacité résiduelle (arc residual capacity inequalities). Le deuxième sous-problème limite le nombre de nœuds à seulement trois nœuds. Ce problème a été étudié par Mirchandani dans [68]. Magnanti, Mirchandani et Vachani [60] ont introduit une nouvelle classe d'inégalités valides appelées inégalités de 3-partition. L'inégalité de 3-partition peux être généralisée pour des réseaux plus grands en considérant une partition en trois sous-ensembles de l'ensemble des nœuds du réseau.

Bienstock et Muratore [13] ont également étudié le NLP mais en insérant la notion de fiabilité. Ils se sont basés sur les contraintes de coupe et les ont renforcées en supprimant une ou plusieurs arêtes de la coupe. Ils ont étudié les points extrêmes du polyèdre associé et décrit des méthodes de lifting pour construire des facettes générales de ce polyèdre. Ils ont également discuté d'un modèle permettant de réserver des capacités sur chaque arc.

1.2.2.2 Le problème TFLP

Magnanti, Mirchandani et Vachani [61] ont étendu leurs résultats sur le NLP au *problème de chargement de réseau à 2 équipements* (TFLP, Two-Facility Capacitated

Network Loading Problem). Ce problème est similaire au NLP mais on dispose de deux types de capacité possibles. On parle alors de *petite capacité* (LC, low capacity) et de *grande capacité* (HC, high capacity). Ce problème a beaucoup d'utilité dans la pratique et principalement dans les réseaux de télécommunications.

Le problème TFLP est équivalent au programme linéaire mixte suivant :

$$\text{Minimiser} \sum_{\{i,j\}\in E} (a_{ij}x_{ij} + b_{ij}y_{ij})$$

$$\sum_{j\in V} f_{ij}^k - \sum_{j\in V} f_{ji}^k = \begin{cases} -u_k & \text{si } i = O(k), \\ u_k & \text{si } i = D(k), \\ 0 & \text{sinon,} \end{cases} \qquad \forall\, i \in V,\, \forall k \in Q, \tag{1.17}$$

$$\sum_{k\in Q} (f_{ij}^k + f_{ji}^k) \leq x_{ij} + Cy_{ij} \qquad \forall\, i,j \in E, \tag{1.18}$$

$$x_{ij},\ y_{ij} \geq 0 \text{ et entiers} \qquad \forall\, i,j \in E, \tag{1.19}$$

$$f_{ij}^k, f_{ji}^k \geq 0 \qquad \forall\, i,j \in E,\, \forall\, k \in Q. \tag{1.20}$$

Dans cette formulation, le graphe $G = (V, E)$ représente le réseau avec V les sommets et E les arêtes et Q l'ensemble des demandes. Chaque demande $k \in Q$ est définie par son origine $O(k)$, sa destination $D(k)$ et son volume u_k. La formulation comprend deux types de variables, les variables de topologie x_{ij} et y_{ij} représentant le nombre d'équipements de type LC et de type HC à installer sur une arête $\{i, j\}$ et les variables de flot f_{ij}^k représentant le flot de la demande k transitant sur l'arête $\{i, j\}$ de i vers j. a_{ij} et b_{ij} représentent les coûts d'installation unitaire d'un équipement LC et d'un équipement HC.

Magnanti, Mirchandani et Vachani [61] ont étendu les inégalités valides pour le NLP au TFLP et ont donné des conditions nécessaires et suffisantes pour que celles-ci définissent des facettes du problème. En utilisant ces résultats, ils ont développé un algorithme de coupes et branchements pour résoudre le problème.

Pour le problème TFLP, les capacités possibles sont de deux types. Le problème MULTISUN présenté ci-après permet de considérer cette fois des capacités appartenant à un ensemble discret, mais avec des sauts de taille variable.

1.2.2.3 Le problème MULTISUN

Le problème connu sous le nom de MULTISUN (MULTIcommodity SUrvivable Network design) consiste à dimensionner le réseau sous conditions de fiabilité. Il consiste

à trouver quelles arêtes doivent être installées et avec quelle capacité, pour que le réseau puisse écouler la demande, avec une panne possible d'une seule arête ou d'un seul nœud. La capacité peut être étendue de manière discrète, par sauts de taille variable. Ce problème est étudié par Dahl et Stoer dans [22] et [80].

Stoer et Dahl [80] ont donné une formulation du problème MULTISUN. Considérons un graphe $G = (V, E)$ fini, non orienté, sans boucle ni arête multiple, où l'ensemble V représente les nœuds qui doivent être reliés dans le réseau et l'ensemble E représente les liaisons pouvant être construites entre les nœuds. Le graphe G est appelé le *graphe support*.

Nous considérons également un graphe $H = (V, D)$ ayant les mêmes nœuds que le graphe G. L'ensemble D représente les différentes demandes entre les nœuds du réseau. Le volume de ces demandes est donné par une fonction $d : D \to \mathbb{R}_+^*$. Le graphe H est appelé le *graphe demande*.

On suppose que chaque arête $e \in E$ possède un certain nombre de capacités possibles, $M_e^0,\ M_e^1, \ldots, M_e^{T_e}$ où $0 = M_e^0 < M_e^1 < \ldots < M_e^{T_e}$ et $T_e \geq 1$. On suppose que $M_e^{T_e}$ est toujours supérieur à la demande sur cette arête, même si le coût est très grand. Soit $m_e^t = M_e^t - M_e^{t-1}$, $t = 1, \ldots, T_e$. Soit c_e^t le coût pour étendre une capacité de M_e^{t-1} à M_e^t. Soient $I = \{(e,t) \mid t = 1, \ldots, T_e, e \in E\}$ et $x \in \mathbb{R}^I$ tel que $x_e^t = 1$ si on considère l'augmentation de capacité de M_e^{t-1} à M_e^t et 0 sinon. Si y est le vecteur capacité associé aux arêtes du réseau, alors y est tel que $y_e = \sum_{t=1}^{T_e} m_e^t x_e^t \ \forall e \in E$.

Le problème MULTISUN, sans considérer de panne, est équivalent au programme linéaire en nombres entiers suivant :

$$\text{Minimiser} \sum_{e \in E} \sum_{t=1}^{T_e} c_e^t x_e^t$$

$$1 \geq x_e^1 \geq x_e^2 \geq \ldots \geq x_e^{T_e} \qquad \forall\, e \in E, \tag{1.21}$$

$$\sum_{e \in E} \mu_e \sum_{t=1}^{T_e} m_e^t x_e^t \geq \sum_{f \in D} \pi_f^\mu d_f \quad \forall\, (\mu, \pi) \in \Pi, \tag{1.22}$$

$$x_e^t \text{ entier} \qquad \forall\, (e,t) \in I, \tag{1.23}$$

où Π est l'ensemble des rayons extrêmes du cône

$$\{\mu \in \mathbb{R}^E,\ \pi \in \mathbb{R}^D \mid \mu \geq 0,\ \pi_f = \pi_f^\mu \quad \forall\, f \in D\}. \tag{1.24}$$

Les inégalités (1.21) sont appelées *contraintes d'ordre*, les inégalités (1.22) sont appelées *inégalités métriques* et les inégalités (1.23) sont les *contraintes d'intégrité*.

Les inégalités métriques ont été très étudiées dans la littérature. On peut par exemple se référer à [6, 50, 58] pour plus de détails.

Stoer et Dahl [80] ont étendu ce modèle en considérant des contraintes de fiabilité. La capacité choisie y doit alors pouvoir supporter la panne d'une arête ou la défaillance d'un nœud. Ils ont donné des conditions nécessaires et suffisantes pour que les contraintes d'ordre (1.21) et les contraintes d'intégrité (1.23) définissent des facettes du polytope associé au problème.

Dans [80], Stoer et Dahl ont introduit différentes classes d'inégalités appelées inégalités de bande, de k-bande et de bande forte, basées sur les relations étroites du problème MULTISUN avec le problème du sac-à-dos. Ils ont également décrit des inégalités basées sur les inégalités de partition et de coupe, propres au problème de conception de réseaux fiables. Dahl et Stoer [22] ont développé un algorithme de coupes pour ce problème basé sur ces résultats.

Chapitre 2

Réseaux de télécommunications multicouches

Un réseau est un ensemble d'équipements interconnectés qui permettent de transporter de l'information d'un accès usager à un autre accès usager. On distingue différents types de réseaux comme les réseaux informatiques permettant de relier des ordinateurs ou les réseaux de télécommunications qui contrairement aux premiers doivent gérer des applications synchrone où le temps est prépondérant. Des données doivent donc être acheminées d'un équipement terminal vers un autre équipement terminal. Le transport des données s'effectue sous la forme de paquets transitant sur le réseau via des nœuds de transfert et des lignes de communication. De nos jours, les services proposés par les réseaux sont de plus en plus diversifiés et nécessitent des qualités de service différentes. On distingue par exemple la visioconférence, la télévision, la vidéoconférence. Les réseaux actuels devraient être rapidement remplacés par des réseaux NGN (Next Generation Network) qui gèreraient tous les médias utilisés dans le cadre de ces nouveaux services. Ce réseau unique serait basé entre autre sur la technologie GMPLS (Generalized MultiProtocol Label Switching).

Dans ce chapitre, nous présentons tout d'abord la structure en couches d'un réseau et plus particulièrement le modèle OSI et deux de ses niveaux : le niveau physique et le niveau paquet. Pour ce dernier, nous présentons en particulier le protocole MPLS (MultiProtocol Label Switching). Nous considérons ensuite les réseaux comme la superposition des couches physique et paquet et décrivons l'interaction entre ces deux couches. On évoquera par exemple les plans de transfert et de commande et les modèles de services et d'interconnexion. On terminera par une description du plan de commande GMPLS. Un glossaire regroupant les différents sigles utilisés est donné à la fin de ce document (voir aussi [74] pour plus de détails).

2.1 Une structure en couches

2.1.1 Le modèle OSI

L'information sous forme de paquets doit être acheminée à travers le réseau. Pour cela, on utilise différents protocoles. Un protocole est une description formelle de règles et de conventions à suivre dans un échange d'informations, que ce soit pour acheminer les données jusqu'au destinataire, ou pour que le destinataire comprennent comment il doit les utiliser. Pour classer ces protocoles, l'ISO (International Organization for Standardization) a défini un modèle de référence appelé modèle OSI (Open System Interconnection) comportant sept niveaux, ou couches, ayant des fonctionnalités différentes.

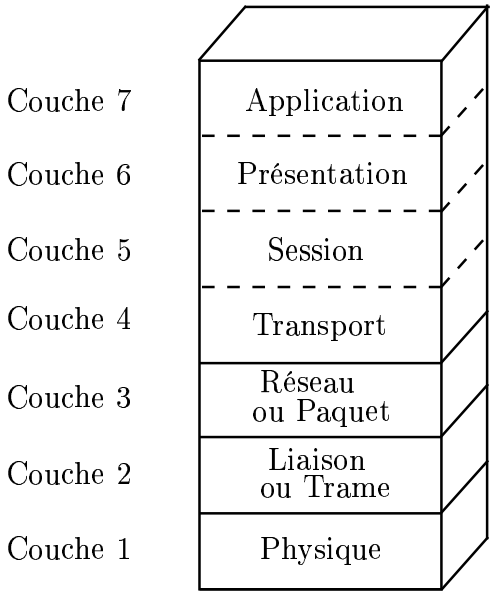


FIG. 2.1 – Modèle de référence

Chacune des couches du modèle OSI a son utilité propre. La couche 1 qui représente le niveau physique, gère les connexions matérielles. La couche 2 ou niveau trame, quant à elle, rend le canal fiable et gère les erreurs de transmission. La couche 3 ou niveau paquet, permet la communication point-à-point. La couche 4 ou niveau message, gère la remise correcte des informations. Viennent ensuite les couches 5, 6 et 7, les niveaux session, présentation et application respectivement, qui constituent les couches hautes. Elles permettent d'abstraire la communication et les données et de rendre le réseau transparent pour l'utilisateur.

Dans la suite, nous considérons un réseau comme la superposition de couches ayant des relations de client-serveur, c'est-à-dire qu'une couche envoie des requêtes qui sont exécutées par la couche immédiatement en dessous. Nous allons plus précisément considérer une architecture IP-sur-optique qui représente la superposition d'une couche de niveau 3 sur une couche de niveau 1 (voir [75]). Dans la suite, nous allons expliquer plus en détails ces deux couches en commençant par le niveau physique.

2.1.2 Niveau physique

Le niveau physique est le plus bas dans le modèle OSI. Il permet de gérer la transmission des signaux électriques. Il définit la façon dont les données vont être transmises sur le support physique. Les informations sont tout d'abord transformées en une suite de bits. On parle de codage de l'information ou de numérisation de l'information. On dispose ensuite de plusieurs techniques permettant la transmission des données. On distingue par exemple la transmission en bande passante, la modulation d'amplitude, de phase ou de fréquence. Parmi les médias permettant de faire transiter les informations, la fibre optique permet de transporter les signaux sous forme optique et non électrique. Les avantages de l'optique sont nombreux, notamment parce que les signaux sont mieux préservés et que les vitesses sont très importantes. Dans la suite, nous examinons les techniques basées sur la technologie optique.

2.1.2.1 La fibre optique

La fibre optique est un guide d'onde qui exploite les propriétés réfractrices de la lumière. En effet, lorsqu'un faisceau lumineux heurte obliquement la surface qui sépare deux milieux plus ou moins transparents, il se divise en deux. Une partie est réfléchie tandis que l'autre est réfractée, c'est-à-dire transmise dans le second milieu en changeant de direction. C'est ce principe qui est utilisé pour guider la lumière dans la fibre. La fibre optique est un tuyau dans lequel la lumière peut se propager. On peut utiliser des ondes de longueurs d'onde différentes pour envoyer simultanément plusieurs signaux.

La fibre optique présente, en terme de transmission, une faible atténuation, une très grande bande passante, un faible poids, une très petite taille et une grande souplesse. Au niveau électromagnétique, la fibre n'est pas sensible aux parasites et n'en crée pas elle-même. La fibre optique est donc un support privilégié pour les transmissions à haut débit. Un autre avantage de la fibre optique est l'aspect sécurité : il est très difficile

de brancher une écoute sur câble optique et une telle opération se traduit par une chute significative du signal dont la cause est facilement localisable. C'est pour tous ces avantages et malgré un prix encore élevé, que la fibre optique est utilisée dans le domaine des télécommunications pour la réalisation des réseaux à haut débit. Elle est utilisée également dans l'audiovisuel, la médecine, l'éclairage ou la signalisation routière.

2.1.2.2 Les brasseurs optiques

Les nœuds d'un réseau sont des centres d'aiguillage où les différentes longueurs d'onde transportées sur une fibre sont orientées vers leurs fibres de destination respectives. Le brassage consiste à mélanger des signaux provenant de fibres différentes. Il peut extraire ou insérer un ou plusieurs signaux avant de les envoyer vers une autre fibre optique. Le nombre de longueurs d'onde transmises sur une même fibre ainsi que la taille des réseaux augmentent considérablement. Il faut donc avoir des nœuds de brassage optique de plus en plus performants, pouvant gérer un très grand nombre de longueurs d'onde.

Il existe plusieurs sortes de brasseurs, comme par exemple les brasseurs optiques (OXC, Optical Cross-Connect), qui sont divisés en deux types : les brasseurs opaques qui utilisent des matrices de commutation électronique et les brasseurs transparents permettant d'aiguiller le faisceau lumineux issu de la fibre à l'aide de miroirs réfléchissants. Ces brasseurs permettent de transférer les signaux optiques indépendamment de leur débit et de leur format.

2.1.2.3 Le multiplexage

Le rôle principal d'un réseau d'accès est de collecter les signaux émis par les abonnés aux divers services de télécommunications et d'assurer leur acheminement vers un centre local où s'effectue leur aiguillage vers le réseau de service adéquat. Le réseau d'accès effectue le regroupement des différents débits d'abonnés, généralement faibles, vers des artères à plus haut débit assurant le raccordement aux réseaux de services. Le regroupement s'effectue dès que possible avec comme objectif de partager au moindre coût les supports physiques de transmission. Le multiplexage permet la transmission simultanée, sur un même support physique, d'informations indépendantes qu'on peut ensuite séparer par démultiplexage. On dit alors que chaque information occupe une voie sur le canal de transmission. Il existe plusieurs types de multiplexage, comme le PDH (Plesiochronous Digital Hierarchy), le SDH (Synchronous Digital Hierarchy), le WDM (Wavelength Division Multiplexing) ou le DWDM (Dense WDM).

2.1.2.4 Optical Transport Networks

Jusqu'à présent, l'architecture des réseaux était basée sur des anneaux SDH connectés par des systèmes DWDM point-à-point. Une topologie en anneaux permet la sécurisation du réseau, mais un système DWDM permet la gestion de capacités plus grandes. Cette architecture est peu à peu remplacée par des OTN (Optical Transport Networks) qui sont des réseaux de transmission haute capacité capables de gérer efficacement les énormes capacités de la transmission sur fibre optique. Leur rôle est de faire transiter des paquets sur des liaisons à 2,5, 10 et 40 Gbit/s. Ils sont composés de brasseurs OXC possédant un plan de commande ainsi qu'un plan de transfert gérés de façon opaque ou transparente selon le cas. Les plans de commande et de transfert seront décrits à la section 2.2.1.

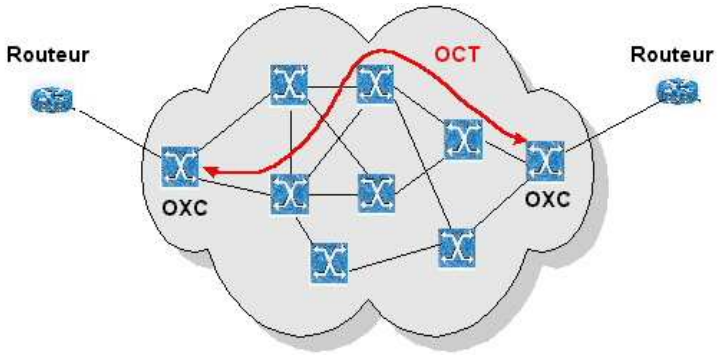


FIG. 2.2 – OTN

Les circuits optiques établis sur un OTN sont appelés OCT (Optical Channel Trail). La gestion d'un OTN demande un plan de commande distribué qui permet un maillage basé sur le principe de protection/restoration.

Le niveau physique est indispensable à un réseau. Il concerne toute la partie équipement du réseau. Nous allons à présent nous intéresser à la troisième couche du modèle OSI : le niveau paquet.

2.1.3 Niveau paquet

Le niveau paquet est le dernier niveau composant les couches basses du modèle OSI. Ce niveau permet d'acheminer correctement les paquets d'information jusqu'à l'utilisateur final. Il prend en charge l'adressage et le routage. Le protocole le plus répandu est le protocole IP que nous présentons maintenant.

2.1.3.1 Le protocole IP

Les clients IP sont des réseaux à commutation par paquets, c'est-à-dire des réseaux dans lesquels l'information est transmise par blocs accompagnés d'une certaine forme d'identification. Le protocole Internet (IP, Internet Protocol) [32, 51, 71] gère la transmission de ces paquets appelés datagrammes, à travers un ensemble de réseaux interconnectés, d'une source vers une destination, la source et la destination étant des machines hôtes identifiées par une adresse de longueur fixe (adresse IP). De nombreux autres protocoles (comme par exemple TCP, Transmission Control Protocol) viennent compléter les en-têtes des paquets pour garantir l'acheminent des paquets et la détection des erreurs [72].

En plus des données traditionnelles, Internet doit maintenant transporter voix et données multimédia. Les ressources nécessaires pour ces nouveaux services, en terme de débit et de bande passante, ont entraîné une transformation de l'infrastructure de l'Internet. Les réseaux actuels doivent pouvoir supporter des transmissions à hauts débits.

2.1.3.2 Le Routage

Parmi les fonctionnalités principales du niveau paquet, on distingue le routage qui consiste à déterminer un chemin permettant de relier deux machines distantes. Déterminer un chemin est une tâche complexe réalisée dans les grands réseaux par des protocoles dédiés dont le rôle est de découvrir la topologie du réseau et d'en tirer la meilleure route. Les protocoles de routage se différencient par les critères de choix des routes et la précision de la topologie découverte. On distingue par exemple, le protocole RIP (Routing Information Protocol) permettant à chaque routeur de communiquer aux autres routeurs la métrique, c'est à dire la distance qui les sépare du réseau IP (en nombre de sauts). Le protocole OSPF (Open Shortest Path First) est plus performant que RIP et l'a donc progressivement remplacé. Il s'agit d'un protocole de routage IP qui, contrairement à RIP, envoie aux routeurs adjacents le nombre de sauts qui les séparent des réseaux IP, chaque routeur transmet alors à tous les routeurs du réseau, l'état de chacun de ses liens. IS-IS (Intermediate System to Intermediate System) est un protocole à état de liens. Ainsi, les routeurs IS-IS maintiennent une vue topologique commune. La base de données topologique est construite individuellement et ensuite partagée entre tous les routeurs. Les paquets sont transmis par le plus court chemin. Le protocole BGP (Border Gateway Protocol) est utilisé pour véhiculer de très grands volumes de données entre des sous-réseaux de grande taille.

La majorité des protocoles de routage traditionnels se basent sur des algorithmes permettant d'obtenir le passage le plus rapide dans le réseau, mais ne prennent pas en compte d'autres mesures, comme les délais ou les congestions qui peuvent sérieusement diminuer les performances du réseau. La gestion du trafic est un objectif pour les administrateurs réseaux. Suite au développement considérable des réseaux de télécommunications et l'augmentation des flux de données, la consultation des tables de routage demande beaucoup de temps (les réseaux sont devenus très grands). Cela a motivé l'utilisation de labels et l'introduction du système MPLS.

2.1.3.3 MPLS

MPLS (MultiProtocol Label Switching) [4] est un système normalisé par l'IETF (Internet Engineering Task Force) pour gérer la signalisation et le routage dans la couche IP. Il a un rôle important dans la commutation et le passage des paquets à travers les réseaux de nouvelle génération. Ainsi, il améliore les services proposés par TCP/IP. MPLS est bien adapté à l'Internet sur réseau optique. Il améliore le rapport performance/prix des équipements de routage, ainsi que l'efficacité du routage en prenant en charge la gestion des contraintes de qualité de service. MPLS spécifie également les mécanismes permettant d'administrer les flux de trafic hétérogènes (entre des matériels différents, des machines différentes ou des applications différentes).

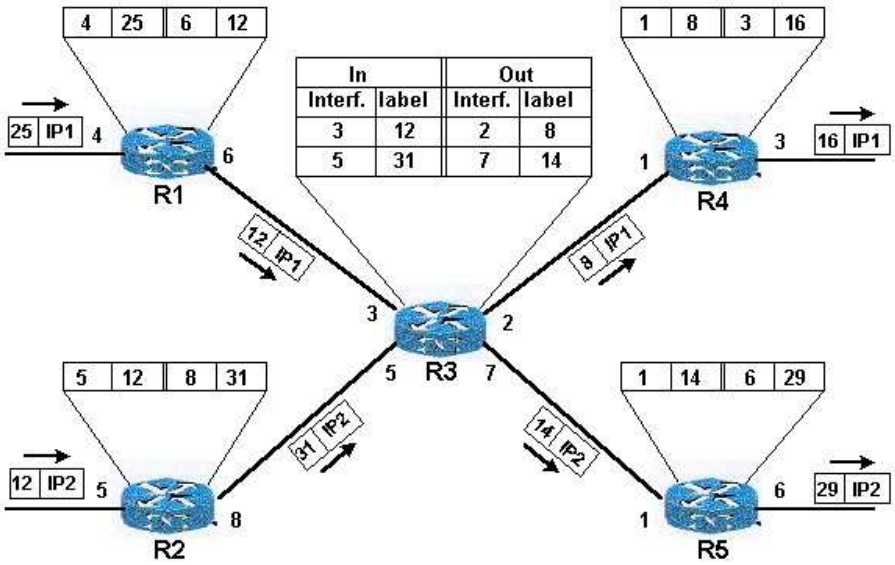


FIG. 2.3 – Commutation de labels

MPLS est basé sur la technique de commutation de labels (Label Swapping). La figure 2.3 illustre ce mécanisme. Par exemple, un paquet IP arrivant sur le routeur $R3$ par l'interface 3, possédant le label 12 va être envoyé sur l'interface 2 du routeur $R3$ et le paquet sortant du routeur aura non plus un label 12, mais un label 8. Cette commutation se fait en utilisant une table beaucoup plus petite que la table de routage globale. Les routeurs permutant ces labels, sans avoir besoin de consulter l'adresse IP ou la table de routage, sont appelés LSR (Label Switch Routers).

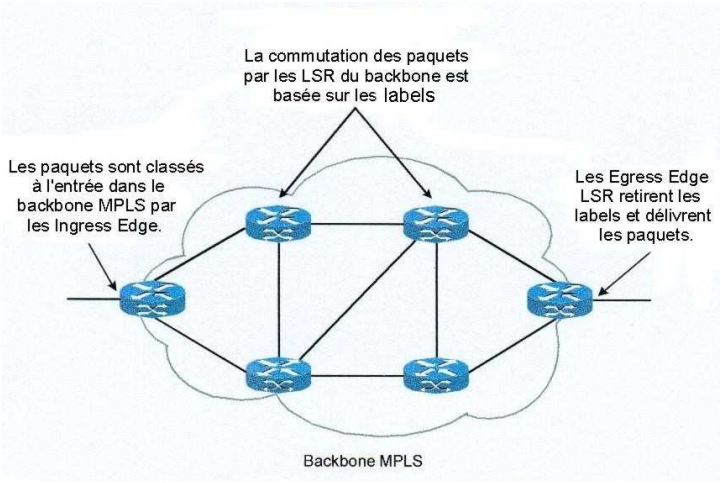


FIG. 2.4 – LSR

Les LSR se basent sur les labels pour commuter les paquets à l'intérieur du réseau MPLS (voir figure 2.4). Il existe des routeurs particuliers, appelés Edge LSR qui sont situés à la périphérie du réseau. Ils possèdent à la fois des interfaces IP traditionnelles et des interfaces connectées au réseau MPLS. Ils sont divisés en deux sous-classes : les Ingress LSR ou routeurs d'entrée, chargés d'imposer les labels aux paquets IP qui les traversent pour entrer dans le réseau MPLS et les Egress LSR ou routeurs de sortie, chargés de retirer les labels.

Les Ingress LSR classent les paquets, à leur entrée dans le réseau MPLS, dans différentes classes appelées FEC (Forwarding Equivalent Classes). Les FEC sont définies en fonction des préfixes IP appris par l'IGP (Interior Gateway Protocol ou protocole de passerelle intérieure) tournant sur le réseau MPLS, ainsi que des informations de qualité de service ou de TE (Traffic Engineering). Les paquets appartenant à une même FEC suivront le même chemin et seront gérés par la même méthode de *forwarding*. Il ne peut pas y avoir de reclassification des paquets à l'intérieur du réseau MPLS. L'ensemble des LSR utilisés par une FEC est appelé un LSP (Label Switched Path) ou chemin à commutation de labels. C'est une séquence de labels. Il existe un unique

LSP pour chaque FEC et les LSP sont unidirectionnels. Les LSR voisins sont connus grâce à l'IGP.

Pour propager les informations sur les labels à tous les LSR, il existe des protocoles de distribution de labels comme TDP/LDP (Tag/Label Distribution Protocol). Il existe plusieurs variantes des méthodes de propagation de labels. Soit les LSR propagent systématiquement tous leurs labels à leurs voisins, soit un LSR doit émettre une demande pour obtenir un label pour un sous-réseau IP demandé.

À partir des informations données par TDP/LDP, les LSR construisent deux tables :
- la TIB (Tag Information Base) contient les labels des LSR voisins. Ces labels ont été affectés à un sous-réseau par chaque LSR voisin.
- la TFIB (Tag Forwarding Information Base) est une sous-table de la TIB. Elle est construite à partir de la TIB et de la table de routage IP, et elle est utilisée pour commuter les paquets. Le LSR choisit l'entrée de la table TIB qui correspond au réseau IP et sélectionne comme label de sortie, le label annoncé par le voisin déterminé par l'IGP.

Le routeur, lorsqu'il reçoit un paquet, se base sur la TFIB pour faire suivre le paquet. Le routeur doit utiliser la technique de commutation CEF (Cisco Express Forwarding) pour pouvoir utiliser la TFIB.

À l'origine, MPLS a été développé pour la rapidité de la commutation de paquets, mais il a très vite permis la mise en œuvre de solutions de plus haut niveau comme les VPN (Virtual Private Networks) ou le TE (Traffic Engineering).

2.1.3.4 Virtual Private Networks

Un VPN (réseau privé virtuel) est un ensemble de sites placés sous la même autorité administrative, ou groupés suivant un intérêt particulier. Une solution VPN permet, par exemple, d'interconnecter les réseaux d'une même entreprise située sur plusieurs sites géographiques. MPLS-VPN fournit une méthode de raccordement de sites appartenant à un ou plusieurs VPN. Les tunnels LSP de MPLS sont nécessaires pour l'encapsulation du trafic VPN.

Dans un environnement MPLS-VPN, nous distinguons plusieurs types de routeurs (voir figure 2.5) :
- P (Provider) : situés à l'intérieur du réseau MPLS, ils acheminent les données par commutation de labels.
- PE (Provider Edge) : situés à la frontière du réseau MPLS, ils possèdent des interfaces reliées à des routeurs clients.

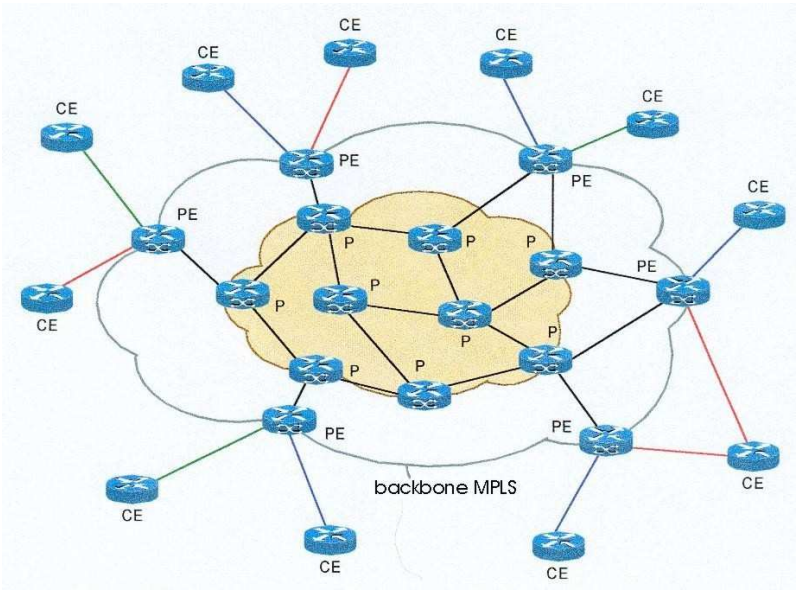


FIG. 2.5 – Différents types de routeurs dans un environnement MPLS-VPN

– CE (Customer Edge) : ce sont des routeurs IP traditionnels n'ayant aucune connaissance de MPLS.

Le but ici, est d'assurer la transmission de paquets IP entre les routeurs CE du même VPN ou de VPN différents.

Les entreprises utilisent de plus en plus l'adressage IP privé, ce qui leur permet d'utiliser les mêmes plages d'adresses IP. MPLS-VPN permet de gérer les recouvrements des plans d'adressage et peut différencier deux adresses identiques appartenant à deux VPN différents. Il permet l'isolation du trafic entre des sites clients n'appartenant pas aux mêmes VPN. Les PE possèdent plusieurs interfaces correspondant à des VPN particuliers. Une VRF (VPN Routing and Forwarding) est affectée à chacune des interfaces et permet aux PE de gérer plusieurs tables de routages (une par interface). Une VRF est constituée d'une table de routage, d'une FIB (Forwarding Information Base) et d'une table CEF, indépendantes des autres VRF et de la table de routage globale. Les VRF sont souvent désignées sur les routeurs par des couleurs qui sont définies localement.

Lorsqu'un routeur PE reçoit un paquet IP sur une interface cliente, il consulte la table de routage de la VRF rattachée à cette interface et n'a pas besoin d'accéder à sa table de routage globale. Cette possibilité d'utiliser plusieurs tables de routage indépendantes permet de gérer un plan d'adressage par site, même en cas de recouvrement d'adresses entre VPN différents.

Pour construire leurs tables VRF, les PE doivent s'échanger les routes correspondant

aux différents VPN. Ils utilisent pour cela le protocole de distribution de label MP-BGP (MultiProtocol BGP). Les CE échangent leurs routes IP avec leur PE au moyen de protocoles de routages classiques (par exemple eBGP, RIPv2 ou OSPF).

Le principe des VRF permet de concevoir facilement des routeurs virtuels, qui consultent leurs différentes tables de routage en fonction de l'interface d'entrée des paquets. Ces tables sont remplies avec les routes du VPN associé et le réseau MPLS assure la transmission des paquets entre les routeurs PE. Le contexte MPLS-VPN permet également un accès Internet, situé par définition à l'extérieur des différents VPN, aux différents clients, en leur permettant d'utiliser le meilleur chemin vers l'extérieur.

2.1.3.5 TE (Traffic Engineering)

La plupart des gros réseaux IP disposent de liens de secours en cas de panne, mais ces liens ne font pas partie des chemins optimaux définis par l'IGP. Le TE permet un meilleur emploi de ces liaisons.

L'IGP gère le routage intérieur dans un sous-réseau. C'est un ensemble de protocoles. Il contient entre autres OSPF et IS-IS. Les IGP classiques sont basés sur le principe du plus court chemin et ne gèrent pas les problèmes de congestion ou de capacité. Le TE permet aux administrateurs réseau d'établir des tunnels LSP à travers le réseau MPLS. Ces tunnels vont emprunter des liens qui ne sont généralement pas utilisés, ce qui permet un meilleur emploi des liaisons. Un tunnel est une séquence de routeurs à emprunter et il peut être défini de manière explicite ou dynamiquement selon les cas. Le TE permet également de réoptimiser les chemins empruntés par les tunnels LSP.

Il existe des priorités et un droit de préemption. Chaque interface MPLS-TE est configurée avec un débit maximum alloué au TE (c'est-à-dire une partie de la bande passante réservée au TE). L'interface répond à une demande d'établissement d'un tunnel par une autorisation ou non. S'il reste une largeur de bande passante suffisante, la permission est accordée et le tunnel est établi. Si ce n'est pas le cas, des tunnels de priorité inférieure sont fermés s'ils permettent d'avoir une capacité suffisante.

Le protocole de routage interne IGP doit être un protocole à état de liens comme OSPF ou IS-IS. En effet, pour déterminer le chemin à emprunter par un tunnel, les routeurs doivent avoir la connaissance complète de la topologie du réseau. MPLS utilise un routage CBR (Constraint-Based Routing) permettant de calculer des chemins en respectant des contraintes entre les nœuds. Ces contraintes peuvent être sur les ressources (critère de bande passante) ou bien administratives (obligation ou interdiction

d'utiliser certains liens dans certains chemins). Le CBR est utilisé pour le TE, mais aussi pour la gestion de la qualité de service (QoS).

En quelques années, le monde des réseaux IP s'est radicalement transformé, en partie grâce à MPLS. Une évolution profonde des infrastructures permet enfin d'atteindre l'Internet haut débit de qualité pour tous et des réseaux privés d'entreprise à qualité controlée.

2.2 Réseaux multicouches

Comme nous l'avons vu plus haut, un réseau est constitué d'un grand nombre de couches interconnectées et ayant chacune des fonctionnalités qui leur sont propres. Nous avons vu plus particulièrement la couche paquet et la couche physique. Dans la suite, nous considérerons un réseau IP-sur-optique comportant une *couche transport* de l'information des clients (ou couche optique) et une couche, appelée *couche cliente* (ou réseau IP), gérant la commande du réseau lui-même. Dans cette section, nous allons décrire les relations fortes existant entre ces couches. Nous allons tout d'abord définir le plan de transfert et le plan de commande, et présenter les interfaces de signalisation et de routage. Puis nous introduirons les différents modèles de service, ainsi que les trois modèles existants d'interconnexion entre les couches. Nous présenterons enfin le plan de commande GMPLS qui est nécessaire à la mise en place de deux de ces modèles d'interconnexion. La plupart des concepts de ce chapitre ont été présentés par J-L. Le Roux et G. Calvignac dans [77].

2.2.1 Plan de transfert, plan de commande

Un réseau est composé de deux couches : la couche cliente et la couche transport. La première est constituée de routeurs IP (dans le cas d'un client IP) qui sont connectés au réseau optique global (OTN) formé par l'interconnexion de plusieurs sous-réseaux de transport. Ces sous-réseaux sont composés de brasseurs optiques (OXC) interconnectés par des liaisons optiques. Les routeurs directement connectés aux réseaux de transport optique sont appelés routeurs de bord (Edge Routers ou Border Routers). Les interfaces logiques permettent la communication entre les deux couches et entre les sous-réseaux optiques.

Il existe deux types d'interfaces logiques (voir figure 2.6) :
- l'interface NNI (Network-to-Network Interface) est une interface entre deux sous-réseaux optiques,

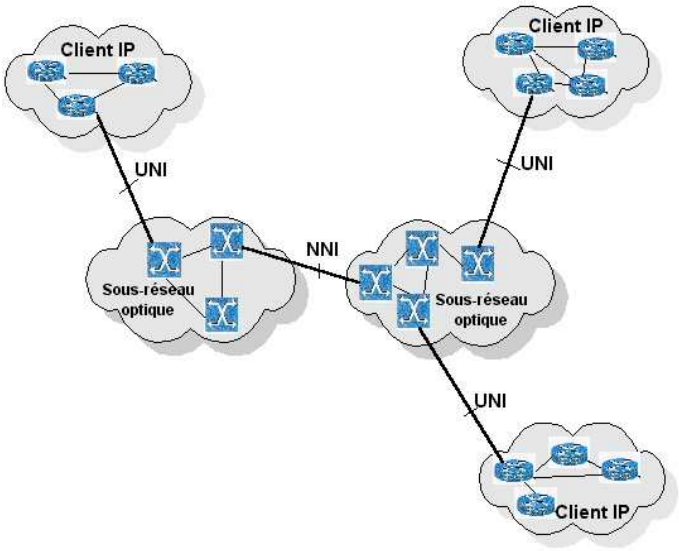


FIG. 2.6 – UNI et NNI

– l'interface UNI (User-Network-Interface) est une interface entre un sous-réseau optique et un sous-réseau IP.

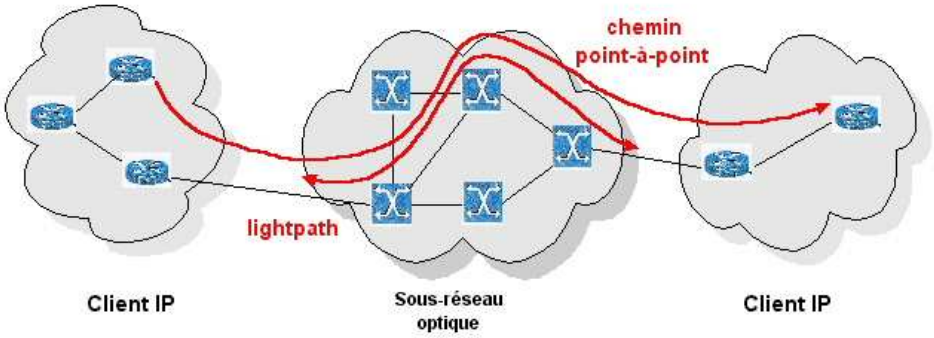


FIG. 2.7 – Lightpath

Les *"lightpaths"* ou circuits optiques sont établis entre un port d'entrée et un port de sortie dans le réseau optique, en réponse à une demande d'un routeur (voir figure 2.7). Ils sont bidirectionnels et sont utilisés dans les chemins point-à-point entre routeurs.

Le *plan de transfert* permet de véhiculer les données. Il n'y a pas d'interaction directe entre les plans de transfert des deux couches. Le *plan de commande* est chargé de la gestion des ressources. Il est utilisé pour la découverte de la topologie, le calcul des routes (routage), l'établissement des circuits optiques (signalisation) et la protection

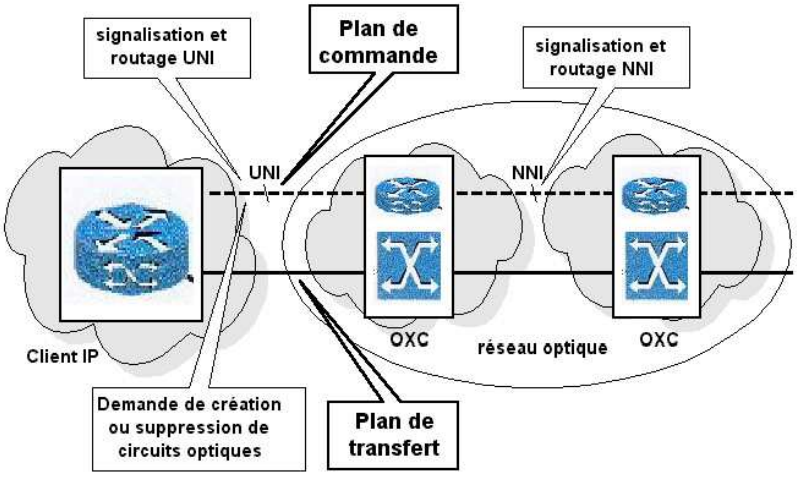


FIG. 2.8 – Plan de transfert et plan de commande

du réseau. Le plan de commande permet de véhiculer les informations de signalisation et de routage à travers les interfaces UNI et NNI (voir figure 2.8). Il transmet les requêtes des routeurs IP concernant la création ou la suppression des circuits optiques aux brasseurs OXC.

2.2.2 Modèles de services

Au niveau de l'organisation du plan de commande, deux modèles de service sont définis : le modèle diversifié et le modèle unifié.

Le modèle de service diversifié (Domain Services)

C'est le modèle existant actuellement. Les plans de commande sont cloisonnés. La signalisation et l'espace d'adressage sont séparés dans la couche transport et dans la couche cliente. La gestion des deux couches est totalement séparée. Le réseau de transport optique fournit des connexions optiques point-à-point au réseau client dans une relation client-serveur. L'interface UNI est nécessaire pour transmettre les messages.

Le modèle de service unifié (Unified Model)

C'est le modèle du futur. Il permet de voir le réseau IP et le réseau de transport optique comme un seul et même réseau. Les services peuvent être demandés de façon continue. Les réseaux de transport optique et leurs réseaux clients doivent avoir le même plan de commande. Les routeurs de la couche cliente et les brasseurs de la

couche optique sont similaires du point de vue du plan de commande. Les interfaces UNI et NNI fonctionnent de la même façon et sont vues comme une interface unique.

L'architecture d'un réseau client sur un réseau optique est définie par l'organisation du plan de commande. Les plans de commande de chacune des deux couches sont associés de façon plus ou moins serrée selon le modèle de service considéré. Cette association est définie à l'aide de modèles d'interconnexion.

2.2.3 Modèles d'interconnexion

L'IETF a défini trois modèles d'interconnexion appelés Overlay, Augmented et Peer [15, 19]. Chaque modèle détermine le volume d'informations sur la topologie et le routage échangé entre le plan de commande de la couche cliente et celui de la couche transport optique. Il définie également le niveau de contrôle des routeurs IP sur le choix des chemins dans le réseau optique.

Les modèles d'interconnexion s'appuient sur les différents modèles de service. La relation entre ces modèles est donnée par la figure suivante :

Modèles de Service	Diversifié		Unifié
Modèles d'Interconnexion	Overlay	Augmented	Peer

FIG. 2.9 – Interaction entre modèles

L'architecture des réseaux actuels est basée principalement sur le modèle de service diversifié et le modèle d'interconnexion Overlay. Les réseaux de télécommunications du futur utiliseront vraisemblablement le modèle de service unifié et le modèle d'interconnexion Peer. Nous allons présenter brièvement chacun de ces modèles d'interconnexion [75].

2.2.3.1 Modèle Overlay

Le modèle Overlay est le modèle couramment utilisé aujourd'hui. Le routage, la distribution de la topologie et la signalisation sont gérés indépendamment sur les deux couches. Les plans de commande peuvent être différents. Le réseau client n'a aucune information concernant le réseau optique et inversement. Le réseau de transport optique fournit un service au réseau client à travers l'interface UNI. Les routeurs ne peuvent

pas connaître les adresses IP accessibles à travers le réseau optique, avant que les circuits optiques (lightpaths) n'aient été établis. Ils doivent donc essayer d'installer les "lightpaths" sans savoir si le routeur IP destination est accessible. Des adjacences de routage doivent être montées sur les circuits optiques établis, afin d'échanger les informations de routage IP entre clients IP. La gestion de ces adjacences de routage (une par circuit optique) est très lourde.

Dans le modèle Overlay, la couche cliente et la couche transport sont complètement séparées. La sécurisation en termes de tolérance aux pannes de chacune des deux couches se fait de façon séparée, ce qui pose de gros problèmes, on ignore si deux liens partagent le même risque.

2.2.3.2 Modèle Augmented

Le modèle Augmented (également appelé modèle interdomaine) est un modèle intermédiaire entre le modèle Overlay et le modèle Peer. Comme pour le modèle Overlay, le routage, la distribution de la topologie et la signalisation sont indépendants, et ce modèle est également très facile à déployer. Néanmoins, le modèle Augmented nécessite un modèle de service unifié. Les deux couches doivent utiliser deux instances du même plan de commande. La topologie du réseau optique n'est pas visible pour le réseau client. Les clients IP ne peuvent donc pas établir des chemins dynamiquement à travers la couche optique. Néanmoins, des informations d'accessibilité sont données aux clients avant que les circuits optiques ne soient établis. Il n'est pas nécessaire de monter des adjacences de routage entre routeurs après l'établissement des circuits optiques.

2.2.3.3 Modèle Peer

Comme le modèle Augmented, le modèle Peer nécessite un modèle de service unifié. Le plan de commande doit être le même sur chacune des deux couches. Le routage et la signalisation sont gérés de la même façon. Les deux couches ont le même schéma d'adressage. Les éléments optiques deviennent comme des entités IP adressables. Tous les nœuds (routeurs ou brasseurs optiques) ont la même connaissance globale du domaine (informations sur la topologie et l'état des liens). Le réseau client a une vision complète du réseau optique, ce qui permet aux routeurs IP d'intégrer les caractéristiques des liens physiques dans le calcul des chemins à l'aide d'un protocole de routage tel que l'extension du protocole OSPF à la couche optique. Ce modèle peut être vu comme un réseau sur une seule couche ayant plusieurs technologies gérées par les mêmes mécanismes. Le modèle Peer permet une optimisation des ressources en calculant les chemins simultanément sur les deux couches. Il permet également la mise en place de

mécanismes de protection multicouches. Il simplifie la gestion du plan de commande. Ce modèle est encore à l'étude, bien que des tests concluants aient été faits, et n'a pas encore de véritable standardisation. Il en est de même pour le plan de commande GMPLS qui pourra être utilisé à la fois sur le réseau client et sur le réseau optique. Dans la suite, nous présentons ce nouveau plan de commande.

2.2.4 GMPLS

A la fin des années 90, le plan de commande MPLS-TE a permis de gérer les connexions dans la couche cliente. Les experts ont voulu étendre le MPLS-TE a un plan de commande permettant de gérer également les connexions dans la couche transport.

Le système GMPLS (Generalized MultiProtocol Label Switching) est un plan de commande générique qui peut être utilisé par les deux couches du réseau. Il supporte les trois modèles d'interconnexion que nous venons de voir. GMPLS généralise le concept de LSR, déja connu avec MPLS sur le réseau client, aux brasseurs OXC. De même, la notion de LSP est également étendue aux OCT de la couche transport qui sont des connexions point-à-point entre un nœud d'entrée et un nœud de sortie.

GMPLS comporte plusieurs fonctionalités principales comme :
- la gestion des canaux de contrôle et des liens de transport,
- la connaissance de la topologie : routage généralisé,
- le calcul des chemins,
- l'établissement et le maintien de circuits : signalisation,
- la sécurisation des circuits.

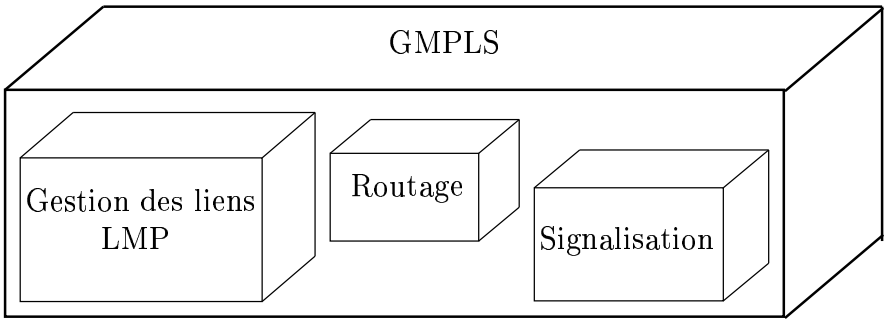


FIG. 2.10 – GMPLS

GMPLS est divisé en trois parties (voir figure 2.10) : le routage, la signalisation et la gestion des liens.

Le routage et la découverte de la topologie

Ils sont gérés par les protocoles ISIS-G ou OSPF-G qui sont des extensions des protocoles ISIS-TE et OSPF-TE existant dans le plan de commande MPLS-TE. Ils peuvent supporter des liens de transport non IP. La notion de TE est également généralisée aux liens de transport.

La signalisation

Les protocoles de signalisation sont utilisés pour établir et maintenir les circuits dans les réseaux de transport. Ils sont basés sur les protocoles de signalisation de MPLS. La notion de label est généralisée ainsi que leur distribution.

La gestion des liens

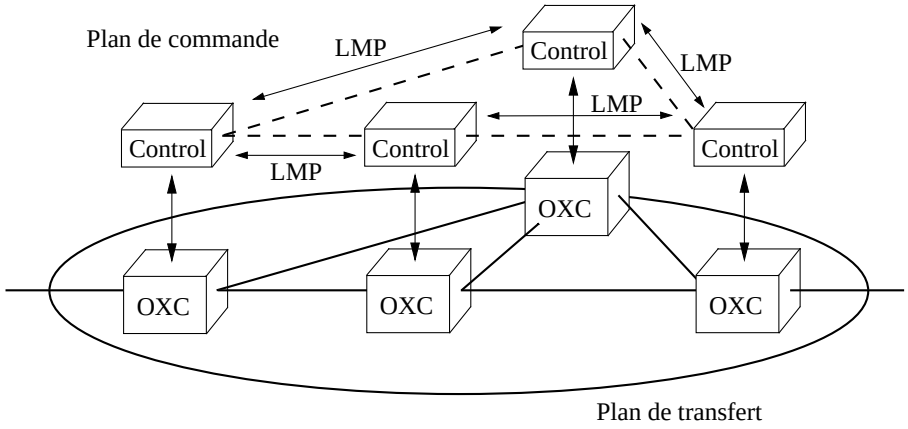


FIG. 2.11 – LMP

Cette partie est toute nouvelle, la gestion des liens est effectuée par le protocole LMP (Link Management Protocol). Contrairement aux réseaux MPLS, dans les réseaux GMPLS, les plans de commande et de transfert d'une couche sont séparés (voir figure 2.11). Les messages de commande comme le routage et la signalisation ne peuvent pas être transportés par les canaux de transfert. Ils doivent avoir des canaux spécifiques. LMP est chargé de l'établissement et de la gestion de ces canaux appelés *canaux de contrôle*. LMP vérifie également l'état des liens de transport. Il doit pouvoir détecter les pannes dans ces liens.

Le système GMPLS semble vouloir s'imposer dans les réseaux de télécommunications du futur. La réutilisation des protocoles de MPLS rend ce système relativement facile à développer. Le GMPLS facilite l'administration des réseaux et améliore les possibilités d'optimisation au niveau du routage, de la topologie, de la tolérance aux pannes. En

effet, il permet une communication entre les deux couches du réseau et peut supporter les trois modèles d'interconnexion. Certains constructeurs ont d'ores et déjà implémenté ce système et d'autres ont lancé son développement.

2.3 Conclusion

Dans ce chapitre, nous avons présenté brièvement les couches physique et paquet du modèle OSI en nous basant plus particulièrement sur la technologie optique pour la première et sur le protocole IP pour le niveau paquet. Nous avons ensuite considéré un réseau comme la superposition de deux couches : la couche transport (ou réseau optique) et la couche cliente (ou réseau IP). Nous avons ensuite décrit les principales interactions entre ces couches et présenté le plan de commande GMPLS commun à ces deux couches. Dans les chapitres suivants, nous allons présenter certains problèmes d'optimisation liés à ces réseaux multicouches. Ils sont liés à des problèmes de fiabilité en termes de tolérance aux pannes et de dimensionnement. Les deux premiers problèmes consistent à fixer la couche cliente et à optimiser la couche transport en termes de topologie pour le premier problème et de dimensionnement pour le deuxième. Pour le troisième problème, c'est la couche transport qui est fixée et la couche cliente doit être sécurisée. Le dernier problème consiste en la sécurisation simultanée des deux couches.

Chapitre 3

Sécurisation multicouche du réseau IP

L'introduction du plan de commande GMPLS et du modèle d'interconnexion Peer nous oblige à reconsidérer les problèmes de fiabilité et de routage. Dans ce chapitre, nous considérons un problème de sécurisation de la couche cliente. Nous présentons tout d'abord le problème et le modélisons en termes de graphe. Puis nous donnons une formulation sous forme d'un programme linéaire en nombres entiers. Nous présentons également de nouvelles contraintes valides pour le problème. Nous donnons des conditions nécessaires et suffisantes pour que certaines d'entre elles définissent des facettes du polytope associé au problème. Nous décrivons également des opérations de réduction de graphes qui sont très utiles lors de la séparation de ces contraintes. Ce travail a fait l'objet de l'article [14].

3.1 Présentation du problème MSIPND

Une des questions qui peut être posée lors de la conception de réseaux de télécommunications multicouches, concerne la sécurisation de la couche cliente. En effet, considérons un modèle Overlay, où les plans de commande et de transfert des deux couches sont séparés. On suppose que la topologie et le routage de la couche transport sont fixés et satisfont aux conditions de fiabilité requises. Supposons que l'on dispose d'un ensemble de brasseurs optiques reliés par des fibres optiques représentant la couche transport. On dispose également d'un ensemble de routeurs IP représentant la couche cliente. A chaque routeur (LSR) est associé un brasseur de la couche optique appelé brasseur de bord. Le routeur et le brasseur associé sont reliés par une interface UNI. De plus, on suppose qu'il existe, entre chaque paire de routeurs de la couche cliente, une

liaison virtuelle à laquelle est associé un coût fixe. Chacune de ces liaisons correspond à un chemin de la couche transport (OCT).

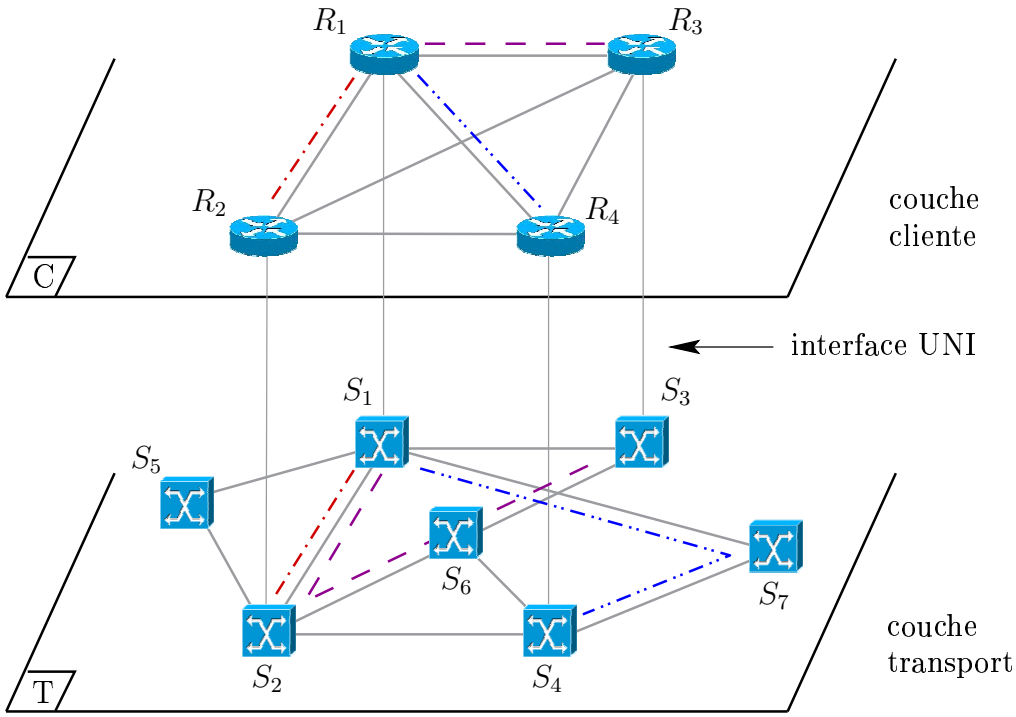


FIG. 3.1 – Exemple de réseau multicouche

La figure 3.1 représente un réseau à deux couches. La couche cliente contient 4 LSR : R_1, R_2, R_3 et R_4. Quant à la couche tranport, elle contient 7 OXC numérotés de S_1 à S_7. Les OXC $S_1, \ldots, S_4$ sont des brasseurs de bord. En effet, le LSR R_1 est relié à l'OXC S_1 via une interface UNI et il en est de même pour les routeurs R_2, R_3 et R_4 et leurs brasseurs associés S_2, S_3 et S_4. On remarque qu'il existe bien une liaison entre chaque paire de routeurs de la couche cliente. La liaison entre R_1 et R_2 dans la couche cliente correspond au chemin allant de S_1 à S_2 dans la couche transport. De même la liaison entre R_1 et R_3 est associée au chemin allant de S_1 à S_3 en passant par S_2 et S_6 et celle entre R_1 à R_3 est associée au chemin allant de S_1 à S_4 en passant par S_7. On remarque que toutes ces liaisons de la couche cliente ayant R_1 comme extrémité ont bien des chemins associés dans la couche transport entre S_1 qui est le point d'entrée de R_1 dans la couche transport (via l'interface UNI) au sommet associé à la deuxième extrémité de l'arête. Les brasseurs S_6 et S_7 qui ne sont pas des brasseurs de bord sont utilisés dans les chemins, même s'ils ne sont pas reliés à un routeur de la couche cliente.

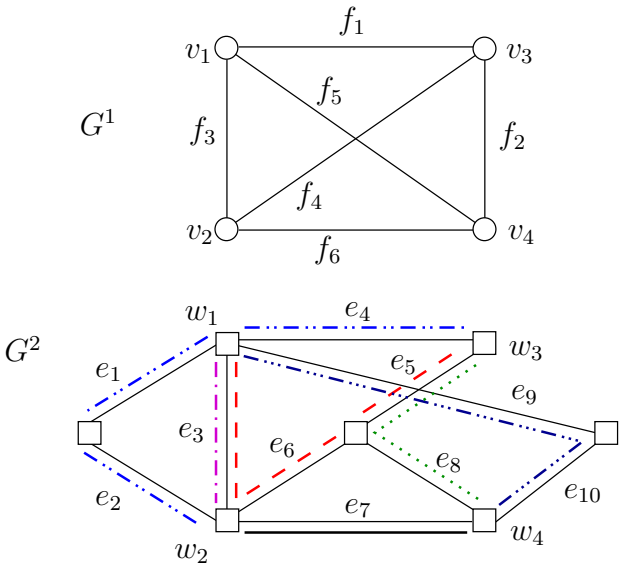
Le routage de la couche transport étant connu, on peut déterminer, pour chaque liaison e de cette couche, l'ensemble des liaisons de la couche cliente qui seront affectées

par la panne de la liaison optique e. Sur la figure 3.1, on remarque que les chemins associés aux liaisons $R_1 - R_2$ et $R_1 - R_3$ utilisent la même liaison $S_1 - S_2$ dans la couche transport. En cas de panne de cette liaison $S_1 - S_2$, les deux liaisons $R_1 - R_2$ et $R_1 - R_3$ seront donc inutilisables.

Le *problème de sécurisation multicouche du réseau IP* (MSIPND Problem, *Multilayer Survivable IP Network Design Problem*) consiste à déterminer les liaisons devant être installées sur la couche cliente, pour que la construction soit de coût minimum et que le réseau puisse supporter, sans être coupé, la panne d'une liaison de la couche transport.

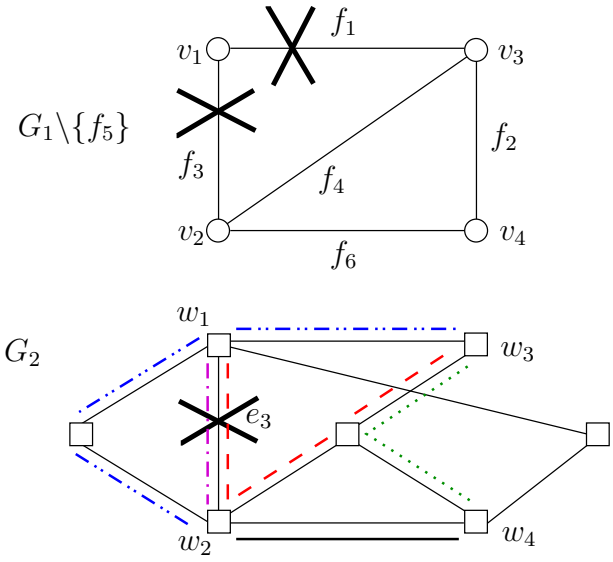
Ce problème peut être modélisé en termes de graphe. On associe à la couche cliente un graphe $G^1 = (V^1, E^1)$ où les sommets de V^1 correspondent aux routeurs et les arêtes de E^1 aux liaisons possibles entre les routeurs. On suppose que le graphe G^1 est complet. Chaque arête f de G^1 est munie d'un coût fixe $c(f) > 0$. On associe à la couche transport un graphe $G^2 = (V^2, E^2)$ où les sommets de V^2 correspondent aux brasseurs optiques et les arêtes de E^2 aux fibres optiques. A chaque sommet $v_i \in V^1$ est associé un sommet $w_i \in V^2$. Chaque arête f de E^1 est associée à un chemin du graphe G^2. Soit P_f l'ensemble des arêtes du chemin de G^2 associé à l'arête $f \in E^1$. Pour chaque arête $e \in E^2$, soit $F_e = \{f \in E^1 \mid e \in P_f\}$. F_e représente l'ensemble des liaisons de la couche cliente coupées lors d'une panne de l'arête e. Chaque arête $f \in F_e$ correspond à un chemin de la couche transport qui emprunte la liaison e. Donc si e tombe en panne, tous les chemins de la couche transport utilisant cette liaison sont coupés. Ceci implique que les liaisons de F_e dans la couche cliente sont aussi hors service. On suppose que le routage est fixé et donc P_f est connu pour chaque $f \in E^1$. On a donc F_e connu pour chaque $e \in E^2$. Le problème MSIPND consiste à trouver un sous-graphe H de G^1 tel que pour toute arête $e \in E^2$, le sous-graphe obtenu à partir de H auquel on a enlevé les arêtes de F_e soit connexe et que le coût total soit minimum.

La figure 3.2 présente les graphes G^1 et G^2 correspondant au réseau multicouche de la figure 3.1. La couche cliente est représentée par le graphe G^1 qui est complet. Les sommets de G^1 sont $v_1, v_2, \ldots, v_4$. Le graphe G^2 représente la couche transport. Chaque sommet $v_i \in V^1$ est associé au sommet $w_i \in V^2$. Les autres sommets ne sont pas nommés. Les arêtes de E^1 sont numérotées $f_1, f_2, \ldots, f_6$ et celles de E^2 $e_1, e_2, \ldots, e_{10}$. Chaque arête de E^1 correspond à un chemin dans le graphe G^2 associé à la couche transport. On a, par exemple, $P_{f_1} = \{e_3, e_5, e_6\}$, $P_{f_3} = \{e_3\}$ et $P_{f_5} = \{e_9, e_{10}\}$. En effet, l'arête f_1 représentant la liaison entre R_1 et R_3 sur la figure 3.1 est associée au chemin (e_3, e_6, e_5). De même, l'arête f_3 correspond au chemin (e_3) et l'arête f_5 correspond au chemin (e_9, e_{10}). Les arêtes f_2, f_4 et f_6 sont également associées à des chemins du graphe G^2.



Arête f	Chemin	P_f
f_1	– – –	e_3, e_5, e_6
f_2	·····	e_5, e_8
f_3	–·–·–	e_3
f_4	–··–··	e_1, e_2, e_4
f_5	–···–···	e_9, e_{10}
f_6	———	e_7

FIG. 3.2 – Exemple du problème



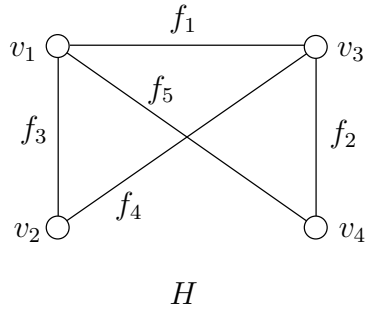
Arête	Chemin
f_1	– – – – –
f_2	·····
f_3	–·–·–·–
f_4	–··–··–··
f_6	———

FIG. 3.3 – Exemple de panne possible

La figure 3.3 illustre la notion de sécurisation. Dans cet exemple, on considère que l'arête f_5 ne fait pas partie de la solution. Si l'arête e_3 du graphe G^2, reliant w_1 et w_2, tombe en panne, les chemins utilisant cette arête sont brisés et donc les arêtes f_1 et f_3 de la couche cliente sont inutilisables. Le sommet $v_1 \in V^1$ est alors isolé.

Le routage étant fixé, on connait les chemins P_f pour tout $f \in E^1$. On peut donc

en déduire les ensembles F_e pour tout $e \in E^2$. On remarque par exemple que $F_{e_3} = \{f_1, f_3\}$. En effet, e_3 appartient aux deux chemins associés à f_1 et f_3. Si l'arête e_3 du graphe G^2 représentant la couche transport, tombe en panne, les liaisons de F_{e_3} ne sont plus utilisables. Si l'on supprime les arêtes de F_{e_3} du graphe $G^1 \backslash \{f_5\}$ de la figure 3.3, le sous-graphe obtenu n'est alors plus connexe et la solution n'est pas réalisable.



H

Arête e	F_e	Arête e	F_e
e_1	f_4	e_6	f_1
e_2	f_4	e_7	f_6
e_3	f_1, f_3	e_8	f_2
e_4	f_4	e_9	f_5
e_5	f_1, f_2	e_{10}	f_5

FIG. 3.4 – Solution réalisable

La figure 3.4 présente les ensembles F_e pour tout $e \in E^2$ et donne un sous-graphe H de G^1 représentant une solution réalisable du problème. Quelque soit l'arête e du graphe G^2 qui tombe en panne, le graphe $G^1 \backslash F_e$ reste connexe.

3.2 Formulation

Nous allons donner ici une formulation mathématique du problème de sécurisation multicouche du réseau IP. Si F est une solution du problème MSIPND alors pour tout $e \in E^2$, $F \backslash F_e$ induit un sous-graphe connexe de G^1. En conséquence, chaque coupe de $G^1 \backslash F_e$ doit contenir au moins une arête de F. De même, si pour tout $e \in E^2$, chacune des coupes de $G^1 \backslash F_e$ contient au moins une arête de F, alors le sous-graphe $F \backslash F_e$ de $G^1 \backslash F_e$ est connexe et donc F est une solution du problème.

Associons maintenant à chaque $F \subseteq E^1$, le vecteur d'incidence de F, $x^F \in \mathbb{R}^{|E^1|}$ tel que $x^F(f) = 1$ si $f \in F$ et 0 sinon. Soit $W \subset V^1$, $W \neq \emptyset$ et $W \neq V^1$, un sous-ensemble de sommets de V^1. En considérant la coupe $\delta_{G^1 \backslash F_e}(W)$ induite par W dans $G^1 \backslash F_e$ pour un certain $e \in E^2$ (voir figure 3.5), alors x^F doit vérifier la contrainte

$$x^F(\delta_{G^1 \backslash F_e}(W)) \geq 1.$$

Les contraintes de ce type sont appelées *contraintes de coupe*.

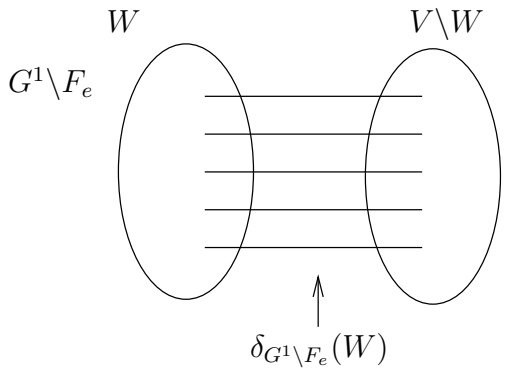


FIG. 3.5 – Coupe

On peut également voir que tout vecteur en 0-1 vérifiant les contraintes de coupe pour tout $W \subset V^1$, $\emptyset \neq W \neq V^1$, et tout $e \in E^2$, induit une solution du problème.

Par conséquent, le problème MSIPND est équivalent au programme linéaire en nombres entiers suivant

$$\text{Minimiser} \sum_{f \in E^1} c(f)x(f)$$

$$x(\delta_{G^1 \setminus F_e}(W)) \geq 1 \qquad \forall\, W \subset V^1,\ \emptyset \neq W \neq V^1,\ \forall\, e \in E^2, \tag{3.1}$$

$$0 \leq x(f) \leq 1 \qquad \forall\, f \in E^1, \tag{3.2}$$

$$x(f) \in \{0, 1\} \qquad \forall\, f \in E^1. \tag{3.3}$$

Les contraintes (3.2) sont appelées les *contraintes triviales*.

On remarque que si $|F_e| = 1$, pour tout $e \in E^2$ et $\bigcup_{e \in E^2} F_e = E^1$, le problème MSIPND n'est rien d'autre que le problème bien connu du sous graphe 2-arête connexe (voir la section 1.2.1). Ce dernier a été largement étudié ces dix dernières années [9, 41, 53, 55, 63]. Ce problème étant NP-difficile, le problème MSIPND l'est également.

3.3 Polyèdre associé et contraintes valides

3.3.1 Polyèdre associé

Nous considérons dans la suite un graphe $G = (V, E)$ et une famille de sous-ensembles d'arêtes de E que l'on notera $\mathcal{F} = \{F_1, \ldots, F_t\} \subseteq 2^E$ avec $t \geq 2$. Nous notons par

$G_i = (V, E_i)$ le sous-graphe de G obtenu en supprimant les arêtes de F_i, $i \in \{1, \ldots, t\}$. On a alors $E_i = E \backslash F_i$. Soit MSIPND$(G, \mathcal{F})$ l'enveloppe convexe des solutions entières du système

$$x(\delta_{G_i}(W)) \geq 1 \qquad \forall W \subseteq V, \quad \emptyset \neq W \neq V, \quad i = 1, \ldots, t, \tag{3.4}$$

$$0 \leq x(f) \leq 1 \qquad \forall f \in E. \tag{3.5}$$

On peut remarquer que si $G = G^1$ et $\mathcal{F} = \{F_e, e \in E^2\}$, MSIPND$(G, \mathcal{F})$ n'est rien d'autre que le polytope associé au problème MSIPND. On appelera MSIPND$(G, \mathcal{F})$ le *polytope des réseaux multicouches fiables* (*Multilayer Survivable IP Network Design Polytope*). On supposera qu'une solution du problème MSIPND est un ensemble d'arêtes $T \subseteq E$ tel que $G[T \backslash F_i]$ est connexe pour $i = 1, \ldots, t$. On notera par $\mathcal{S}(G, \mathcal{F})$ l'ensemble des solutions du problème MSIPND associé au graphe G et à la famille $\mathcal{F}$. On supposera également que $E \in \mathcal{S}(G, \mathcal{F})$.

Une arête e de G est dite *essentielle* s'il existe $i \in \{1, \ldots, t\}$ tel que e forme une coupe dans G_i. On peut noter que e est essentielle si et seulement si e appartient à toute solution $T \in \mathcal{S}(G, \mathcal{F})$. Soit E^* l'ensemble des arêtes essentielles de G. Le théorème suivant donne la dimension du polytope.

Théorème 3.1 *dim(MSIPND$(G, \mathcal{F})$)=$|E| - |E^*|$.*

Preuve. Si $e \in E^*$, comme $e \in T$ pour tout $T \in \mathcal{S}(G, \mathcal{F})$, la contrainte $x(e) = 1$ est une équation du système linéaire décrivant MSIPND$(G, \mathcal{F})$. Donc dim(MSIPND$(G, \mathcal{F})$) $\leq |E| - |E^*|$.
De plus, si $e \in E \backslash E^*$, $T_e = E \backslash \{e\} \in \mathcal{S}(G, \mathcal{F})$. En considérant les ensembles $T_e, e \in E \backslash E^*$ et E, on obtient $|E| - |E^*| + 1$ solutions dont les vecteurs d'incidence x^E et x^{T_e}, $e \in E \backslash E^*$ sont affinement indépendants. Donc dim(MSIPND$(G, \mathcal{F})$) $\geq |E| - |E^*|$, et par conséquent, dim(MSIPND$(G, \mathcal{F})$) $= |E| - |E^*|$. □

Comme conséquence du théorème 3.1, nous avons le corollaire suivant.

Corollaire 3.2 *MSIPND$(G, \mathcal{F})$ est de pleine dimension si et seulement si G_i est 2-arête connexe pour $i = 1, \ldots, t$.*

Dans la suite, on suppose que MSIPND$(G, \mathcal{F})$ est de pleine dimension. On supposera également que $G = (V, E)$ est un graphe complet. Cette dernière supposition n'est pas restrictive puisque le problème dans un graphe incomplet peut être réduit en un problème sur un graphe complet en donnant un coût suffisamment grand aux arêtes inexistantes. On supposera également que toutes les arêtes de G appartiennent au moins à un ensemble $F_i \in \mathcal{F}$.

Nous introduisons maintenant deux définitions qui sont utiles pour la suite.

Définition 3.3 *Un sous-graphe $H = (W, F)$ de $G = (V, E)$ est dit $\mathcal{F}$-connexe par rapport à $\mathcal{F} = \{F_1, \ldots, F_t\}$ si pour tout $i \in \{1, \ldots, t\}$, le graphe $H \backslash F_i$ est connexe.*

En particulier, un sous-graphe formé de 3 sommets reliés par 3 arêtes telles que chaque ensemble $F_i \in \mathcal{F}$ ne contienne pas plus d'une de ces 3 arêtes, est un sous-graphe $\mathcal{F}$-connexe.

Définition 3.4 *Un sous-graphe $H = (W, F)$ de $G = (V, E)$ est dit 2-$\mathcal{F}$-connexe par rapport à $\mathcal{F} = \{F_1, \ldots, F_t\}$ si pour tout $e \in F$, le graphe $H \backslash e$ est $\mathcal{F}$-connexe.*

Dans la suite, nous introduisons trois classes d'inégalités valides pour le problème MSIPND. Nous donnons des conditions nécessaires et des conditions suffisantes pour que ces inégalités définissent des facettes.

3.3.2 Contraintes de partition

La famille de contraintes valides que nous allons présenter dans la suite est induite par une partition de l'ensemble des nœuds du graphe.

Proposition 3.5 *Soient $V_1, \ldots, V_p$, $p \geq 2$, une partition de V et $F_i \in \mathcal{F}$. Alors*

$$x(\delta_{G_i}(V_1, \ldots, V_p)) \geq p - 1 \tag{3.6}$$

est valide pour MSIPND$(G, \mathcal{F})$.

Preuve. Comme G_i doit être connexe, tout graphe obtenu à partir de G_i en contractant les arêtes doit aussi être connexe, et doit donc contenir un arbre couvrant. Si le nombre de sommets du graphe contracté est p, alors ce graphe doit contenir au moins $p - 1$ arêtes. □

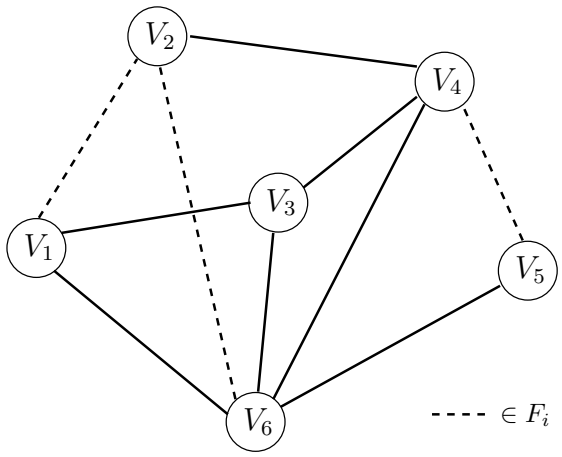


FIG. 3.6 – Partition

Les inégalités (3.6) sont appelées *contraintes de partition.*

Par exemple, la partition de la figure 3.6, divise l'ensemble des sommets en six ensembles. Les arêtes de l'ensemble F_i apparaissent en pointillés. Pour que le graphe G_i reste connexe, il faut que l'on ait au moins 5 arêtes en traits pleins (*i.e.* n'appartenant pas à l'ensemble F_i) entre les différents éléments de la partition.

Si $G = (V, E)$ est un graphe et $V_1, \ldots, V_p$, $p \geq 2$, est une partition de V, on notera par $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ le sous-graphe de G obtenu en contractant les sommets de V_j, $j = 1, \ldots, p$. Par extension, on notera $\widetilde{G}_i = (\widetilde{V}, \widetilde{E}_i)$ le graphe obtenu à partir de G_i, $i = 1, \ldots, t$, en contractant les ensembles $V_1, \ldots, V_p$. De même on notera par $\widetilde{F}_i$ la restriction de F_i dans $\widetilde{E}$.

Théorème 3.6 *L'inégalité (3.6) definit une facette de MSIPND$(G, \mathcal{F})$ seulement si*

1) $G_i(V_j)$ est 2-arête connexe, pour $j = 1, \ldots, p$,

2) $G(V_j)$ est $\mathcal{F}$-connexe pour $j = 1, \ldots, p$, si $\widetilde{E} \cap F_i = \emptyset$,

3) pour tout $j \in \{1, \ldots, t\} \backslash \{i\}$ tel que $F_j \cap \widetilde{E}_i \neq \emptyset$, il existe un ensemble d'arêtes $\widetilde{T} \subseteq \widetilde{E}_i$ tel que

3.1) $F_j \cap \widetilde{T} \neq \emptyset$,

3.2) $\widetilde{G}_i[\widetilde{T}]$ est un arbre,

3.3) toute coupe $\delta_{\widetilde{G}}(W)$ intersecte F_i, si $W \subset \widetilde{V}$ et $\delta_{\widetilde{G}_i}(W) = F_j \cap \widetilde{T}$,

4) il existe un ensemble d'arêtes $\widetilde{T} \subset \widetilde{E}_i$ induisant un arbre dans $\widetilde{G}_i$, tel que

$$T_0 = \widetilde{T} \cup F_i \cup (\cup_{j=1}^{p} E_G(V_j))$$

est une solution de $\mathcal{S}(G, \mathcal{F})$.

Preuve. 1) Supposons par exemple que $G_i(V_1)$ n'est pas 2-arête connexe. Il existe alors une partition V_1^1, V_1^2 de V_1 telle que $|\delta_G(V_1^1, V_1^2) \backslash F_i| \leq 1$. Considérons la partition $V_1', \ldots, V_{p+1}'$ telle que

$$\begin{aligned} V_1' &= V_1^1, \\ V_2' &= V_1^2, \\ V_j' &= V_{j-1} \text{ pour } j = 3, \ldots, p+1. \end{aligned}$$

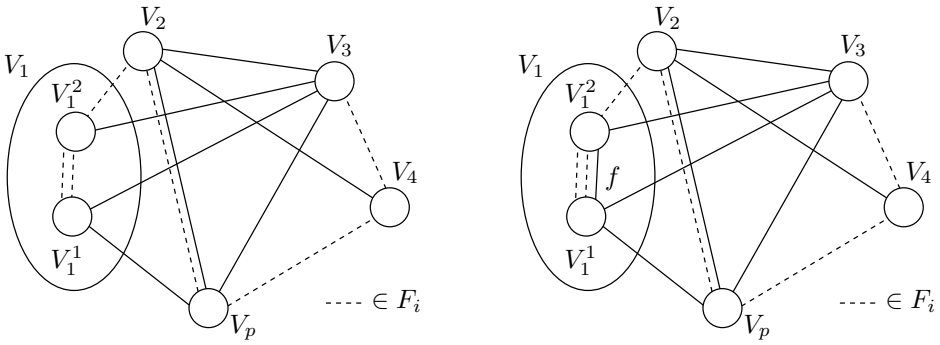


FIG. 3.7 –

Si $\delta_G(V_1^1, V_1^2) \backslash F_i = \emptyset$ (voir figure 3.7(a)), alors l'inégalité (3.6) est dominée par la contrainte correspondant à la partition $V_1', \ldots, V_{p+1}'$. Si ceci n'est pas le cas, et donc $\delta_G(V_1^1, V_1^2) \backslash F_i = \{f\}$ pour une arête f (voir figure 3.7(b)), alors l'inegalité (3.6) peut être écrite comme la somme de $-x(f) \geq -1$ et de la contrainte de partition associée à la partition $V_1', \ldots, V_{p+1}'$. Il s'ensuit que dans les deux cas, l'inégalité (3.6) ne définit pas une facette.

2) Supposons que $G(V_1)$ n'est pas $\mathcal{F}$-connexe. Alors il existe $k \in \{1, \ldots, t\}$ tel que $G_k(V_1)$ n'est pas connexe. En conséquence, il existe une partition V_1^1, V_1^2 de V_1 avec $\delta_G(V_1^1, V_1^2) \backslash F_k = \emptyset$. Si $\widetilde{E} \cap F_i = \emptyset$, alors toute solution de $\mathcal{S}(G, \mathcal{F})$ doit contenir au moins p arêtes de $\widetilde{E}_i$. Ceci implique que l'inégalité (3.6) n'est pas serrée par le vecteur d'incidence de toute solution de $\mathcal{S}(G, \mathcal{F})$. Il en résulte que (3.6) ne peut pas définir une facette.

3) Supposons le contraire. Soit $j \in \{1, \ldots, p\} \backslash \{i\}$ avec $F_j \cap \widetilde{E}_i \neq \emptyset$ tel que pour tout $\widetilde{T} \subseteq \widetilde{E}_i$, au moins une des assertions 3.1) 3.2) et 3.3) n'est pas satisfaite. On va montrer que (3.6) ne peut pas définir une facette. Comme $F_j \cap \widetilde{E}_i \neq \emptyset$, soit $f \in F_j \cap \widetilde{E}_i$. Si l'inégalité (3.6) définit une facette, comme (3.6) est différente de l'inégalité $x(f) \geq 0$, il

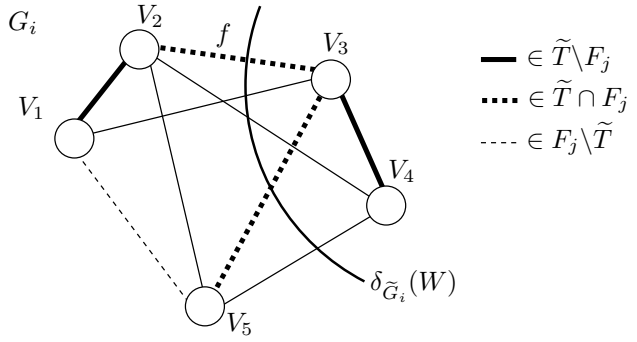


FIG. 3.8 –

doit exister une solution, notée $T \in \mathcal{S}(G,\mathcal{F})$, contenant f et dont le vecteur d'incidence satisfait l'inégalité (3.6) à l'égalité. Comme $T \in \mathcal{S}(G,\mathcal{F})$, $G[T\backslash F_i]$ est connexe, et donc $\widetilde{T} = T \cap \widetilde{E}_i$ induit un sous-graphe connexe de $\widetilde{G}_i$. De plus, comme $|\widetilde{T}| = |T \cap \widetilde{E}_i| = p-1$, $\widetilde{T}$ doit être un arbre dans $\widetilde{G}_i$ (voir figure 3.8). Notons que $f \in \widetilde{T}$ et $f \in F_j \cap \widetilde{E}_i$, donc $\widetilde{T} \cap F_j \neq \emptyset$. En conséquence, $\widetilde{T}$ satisfait 3.1) et 3.2). D'après notre hypothèse, $\widetilde{T}$ ne peut donc pas satisfaire 3.3). Soit $W \subseteq \widetilde{V}$ un ensemble de sommets induisant l'unique coupe $\delta_{\widetilde{G}_i}(W)$ dans $\widetilde{G}_i$ telle que $\delta_{\widetilde{G}_i}(W) = \widetilde{T} \cap F_j$. (Cette coupe peut être détectée par un simple étiquetage des sommets.) Pour l'exemple de la figure 3.8, on peut remarquer que la coupe $\delta_{\widetilde{G}_i}(W)$ contient deux arêtes de $\widetilde{T}$. Comme $\widetilde{T}$ ne satisfait pas 3.3), $\delta_{\widetilde{G}}(W) \cap F_i = \emptyset$. Puisque $\delta_{\widetilde{G}_i}(W) = \widetilde{T} \cap F_j$, on a $\delta_{\widetilde{G}}(W) \cap T \subseteq F_j$, et donc le sous-graphe induit par $T\backslash F_j$ n'est pas connexe. Ceci contredit le fait que T est une solution de $\mathcal{S}(G,\mathcal{F})$.

4) Si l'assertion n'est pas vérifiée, alors toute solution de $\mathcal{S}(G,\mathcal{F})$ contient au moins p arêtes de $\widetilde{E}_i$. Donc la face définie par l'inégalité (3.6) est vide, et en conséquence, l'inégalité ne peut définir de facette. □

Théorème 3.7 *L'inégalité (3.6) définit une facette de MSIPND$(G,\mathcal{F})$ si*

1) les conditions 1), 2), 3) du théorème 3.6 sont satisfaites,

2) $G(V_i)$ est 2-$\mathcal{F}$-connexe pour $i = 1,\ldots,p$,

3) il existe $\widetilde{T} \subseteq \widetilde{E}_i$ satisfaisant la condition 4) du théorème 3.6, tel que pour tout $W \subset \widetilde{V}$ et F_j, $j \neq i$, si $\delta_{\widetilde{G}}(W) \cap (\widetilde{F}_i\backslash\widetilde{F}_j) = \emptyset$ (resp. $|\delta_{\widetilde{G}}(W) \cap (\widetilde{F}_i\backslash\widetilde{F}_j)| = 1$), alors $|\delta_{\widetilde{G}}(W) \cap (\widetilde{T}\backslash\widetilde{F}_j)| \geq 2$ (resp. $|\delta_{\widetilde{G}}(W) \cap (\widetilde{T}\backslash\widetilde{F}_j)| \geq 1$),

4) $\widetilde{G}_i$ is 2-sommet connexe.

Preuve. Notons par $ax \geq \alpha$ l'inégalité (3.6), et soit $bx \geq \beta$ une inégalité définissant une facette de MSIPND$(G,\mathcal{F})$ tels que $\{x \in \text{MSIPND}(G,\mathcal{F}) \mid ax = \alpha\} \subseteq \{x \in$

MSIPND$(G,\mathcal{F}) \mid \ bx = \beta\}$. Pour que $ax \geq \alpha$ définisse une facette, il suffit de montrer qu'il existe $\rho > 0$ tel que $b = \rho a$. Pour cela, nous montrons tout d'abord qu'il existe $\rho \in I\!R$ tel que

$$b(e) = \rho \quad \text{pour tout } e \in \widetilde{E}_i. \tag{3.7}$$

Par la condition c) il existe un ensemble d'arêtes $\widetilde{T} \subseteq \widetilde{E}_i$ induisant un arbre dans $\widetilde{G}_i$ tel que $T_0 = \widetilde{T} \cup F_i \cup (\cup_{j=1,\ldots,p} E_G(V_j))$ appartient à $\mathcal{S}(G,\mathcal{F})$. Soit $\widetilde{T}_0 = \widetilde{T} \cup \widetilde{F}_i$. $\widetilde{T}_0$ est solution de $\mathcal{S}(\widetilde{G},\widetilde{\mathcal{F}})$. Comme $\widetilde{T}$ est un arbre couvrant dans $\widetilde{G}_i$, pour tout arête $e \in \widetilde{E}_i \backslash \widetilde{T}$,

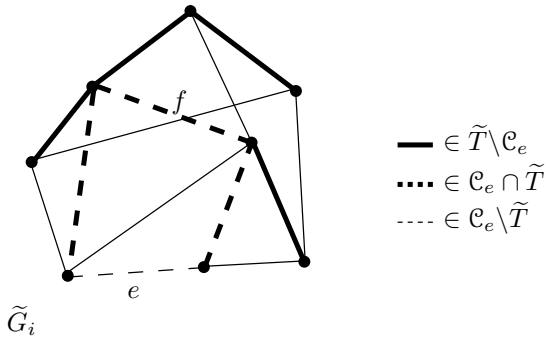


FIG. 3.9 –

il existe un unique cycle dans $\widetilde{G}_i$ formé par e et un chemin de $\widetilde{T}$. On notera par $\mathcal{C}_e$, V_e et P_e, ce cycle, son ensemble de sommets et le chemin de $\widetilde{T}$ dans $\mathcal{C}_e$, respectivement (voir figure 3.9).
Soit $e \in \widetilde{E}_i \backslash \widetilde{T}$. Soit $f \in \mathcal{C}_e \backslash \{e\}$. Soit $T_e = (T_0 \backslash \{f\}) \cup \{e\}$. On va montrer que $T_e \in \mathcal{S}(G,\mathcal{F})$. Comme $T_0 \in \mathcal{S}(G,\mathcal{F})$, il suffit de montrer que $\widetilde{T}_e = (\widetilde{T}_0 \backslash \{f\}) \cup \{e\}$ est une solution de $\mathcal{S}(\widetilde{G},\widetilde{\mathcal{F}})$. Notons tout d'abord que, comme $(\widetilde{T} \backslash \{f\}) \cup \{e\}$ est un arbre dans $\widetilde{G}_i$, alors $\widetilde{T}_e \backslash \widetilde{F}_i$ induit un sous-graphe connexe dans $\widetilde{G}_i$. Maintenant considérons les ensembles $\widetilde{F}_j$, $j \neq i$. Si $f \in \widetilde{F}_j$, comme $\widetilde{T}_0 \in \mathcal{S}(\widetilde{G},\widetilde{\mathcal{F}})$ et donc $\widetilde{G}[\widetilde{T}_0 \backslash \widetilde{F}_j]$ est connexe, il s'ensuit que $\widetilde{G}[\widetilde{T}_e \backslash \widetilde{F}_j]$ est connexe. Supposons que $f \notin \widetilde{F}_j$. Si $\widetilde{G}[\widetilde{T}_e \backslash \widetilde{F}_j]$ n'est pas connexe, alors il existe $W \subseteq \widetilde{V}$ tel que $\delta_{\widetilde{G}}(W) \cap (\widetilde{T}_e \backslash \widetilde{F}_j) = \emptyset$. Comme $\widetilde{G}[\widetilde{T}_0 \backslash \widetilde{F}_j]$ est connexe, on a $\delta_{\widetilde{G}}(W) \cap (\widetilde{T}_0 \backslash \widetilde{F}_j) = \{f\}$. Donc $\delta_{\widetilde{G}}(W) \cap (\widetilde{F}_i \backslash \widetilde{F}_j) = \emptyset$ et $|\delta_{\widetilde{G}}(W) \cap (\widetilde{T} \backslash \widetilde{F}_j)| < 2$. Mais ceci est une contradiction. Ainsi $\widetilde{G}[\widetilde{T}_e \backslash \widetilde{F}_j]$ est connexe. En conséquence, $\widetilde{T}_e \in \mathcal{S}(\widetilde{G},\widetilde{\mathcal{F}})$, et donc $T_e \in \mathcal{S}(G,\mathcal{F})$.

Comme $ax^{T_0} = ax^{T_e} = \alpha$, on a $bx^{T_0} = bx^{T_e} = \beta$ et en conséquence $b(e) = b(f)$. Comme f est une arête quelconque de $\mathcal{C}_e$ et e est quelconque dans $\widetilde{E}_i \backslash \widetilde{T}$, on a

$$b(e) = b(f) \quad \text{pour tout } f \in \mathcal{C}_e, \quad \text{pour tout } e \in \widetilde{E}_i \backslash \widetilde{T}. \tag{3.8}$$

Si $\mathcal{C}_e$ contient une corde $h = uv$ n'appartenant pas à F_i, alors $h \in \widetilde{E}_i \backslash \widetilde{T}$. Notons que

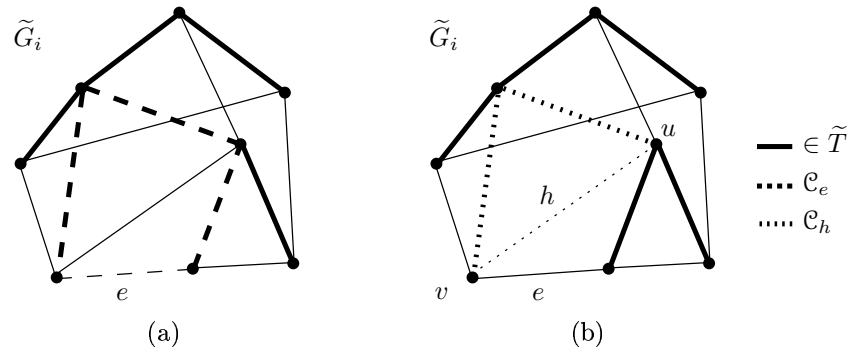


FIG. 3.10 –

$\mathcal{C}_h \backslash \{h\} \subset \mathcal{C}_e$. (La figure 3.10 reprend l'exemple de la figure 3.9. Sur la figure 3.10(a), on peut remarquer que le cycle $\mathcal{C}_e$ n'a qu'une seule corde. La figure 3.10(b) représente le cycle associé à cette corde notée h.) D'après (3.8) par rapport à e et $\mathcal{C}_e$ et h et $\mathcal{C}_h$ on a $b(e') = \rho$ pour un certain ρ pour toute arête $e' \in \mathcal{C}_e \cup \mathcal{C}_h$. Comme h est une corde quelconque de $\mathcal{C}_e$, on obtient

$$b(e') = \rho \quad \text{pour tout } e' \in E_{\widetilde{G}_i}(V_e). \tag{3.9}$$

Comme par d), $\widetilde{G}_i$ est 2-sommet connexe, $\widetilde{E}_i \backslash \widetilde{T} \neq \emptyset$. D'où, par le précédent développement, ainsi que (3.9), on peut supposer qu'il existe un sous-ensemble de sommets $U \subset \widetilde{V}$ tel que la restriction de $\widetilde{T}$ sur U, notée $\widetilde{T}_U$, est un arbre couvrant dans $\widetilde{G}_i(U)$, et

$$b(e) = \rho \quad \text{pour tout } e \in E_{\widetilde{G}_i}(U). \tag{3.10}$$

Si $U = \widetilde{V}$, alors (3.7) est satisfaite et la preuve est terminée. Supposons que ce ne soit pas le cas. Pour prouver (3.7), il suffit de montrer qu'il existe un sous-ensemble de sommets U' strictement contenu dans U et pour lequel (3.10) est satisfaite. Comme $\widetilde{T}$ est un arbre couvrant dans $\widetilde{G}_i$, il existe une arête notée $g = uw$ de $\widetilde{T}$ appartenant à la coupe $\delta_{\widetilde{G}_i}(U)$ avec $u \in U$ et $w \in \widetilde{V} \backslash U$. Comme $\widetilde{T}_U$ est un arbre couvrant dans $\widetilde{G}_i(U)$, il doit exister un sommet $u' \in U$ tel que $g' = uu'$ est une arête de $\widetilde{T}_U$. Comme $\widetilde{G}_i$ est 2-sommet connexe, il doit exister un chemin P entre u' et w n'empruntant pas u. Soit $\mathcal{C}$ le cycle formé par g, g' et P. On montre qu'il existe au moins une arête, notée e', dans $\mathcal{C} \backslash \widetilde{T}$ telle que le chemin $P_{e'}$ de $\mathcal{C}_{e'}$ contient à la fois les arêtes g et g'. En effet, si ce n'est pas le cas, alors l'ensemble d'arêtes $T^* = \{g, g'\} \cup \{\bigcup_{e \in \mathcal{C} \backslash \widetilde{T}} P_e\} \cup (\widetilde{T} \cap \mathcal{C})$ contient un cycle. Comme $T^* \subset \widetilde{T}$, on obtient une contradiction. En conséquence, le cycle $\mathcal{C}_{e'}$ contient g et g'. (La figure 3.11 illustre cette contradiction, en effet, sur cet exemple, on suppose que le chemin $P_{e'}$ ne contient ni g, ni g'. Alors, clairement, on

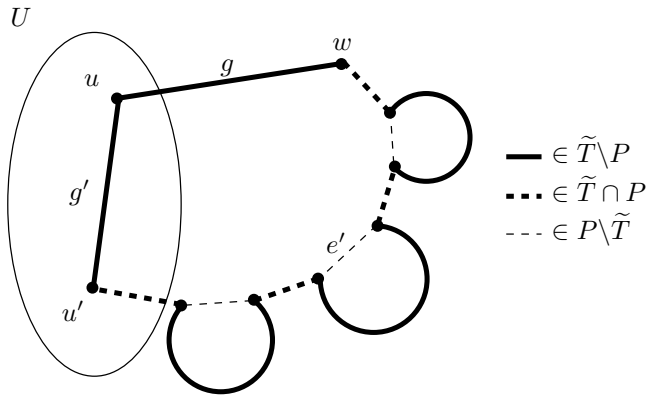


FIG. 3.11 –

peut voir que les arêtes en traits épais (pleins ou pointillés) forment un cycle, ce qui est impossible.) Par (3.10), on a $b(e) = \rho'$ pour $\rho' \in I\!R$ et pour toute arête $e \in E_{\widetilde{G}_i}(V_{e'})$. Comme g et g' appartiennent à $\mathcal{C}_{e'}$, d'après (3.8) on a $\rho = \rho'$, et donc $b(e) = \rho$ pour tout $e \in E_{\widetilde{G}_i}(U \cup V_{e'})$. Comme $w \in V_{e'}$, $U' = U \cup V_{e'}$ contient strictement U. De plus, on a $b(e) = \rho$ pour tout $e \in E_{\widetilde{G}_i}(U')$, ce qui complète la preuve de (3.7). On a donc montré que $b(e) = \rho$ pour tout $e \in \widetilde{E}_i$.

Si e est une arête de $\delta_G(V_1, \ldots, V_p) \cap F_i$, on peut montrer que $\widetilde{T}'_0 = \widetilde{T}_0 \backslash \{e\}$ est une solution de $\mathcal{S}(G, \mathcal{F})$. Comme $ax^{\widetilde{T}'_0} = ax^{\widetilde{T}_0} = \alpha$ et $bx^{\widetilde{T}'_0} = bx^{\widetilde{T}_0} = \beta$, alors $b(e) = 0$. De plus, si f est une arête de $E_G(V_i)$, comme par b) $G(V_i)$ est 2-$\mathcal{F}$-connexe, alors $\widetilde{T}_0 \backslash \{f\} \in \mathcal{S}(G, \mathcal{F})$, et donc de la même manière $b(f) = 0$.

Par conséquent,

$$b(e) = \begin{cases} \rho & \text{pour tout } e \in \delta_{G_i}(V_1, \ldots, V_p), \\ 0 & \text{pour tout } e \in \cup(\cup_{j=1,\ldots,p} E_G(V_j)). \end{cases}$$

Donc $b = \rho a$. Comme la face induite par $bx \geq \beta$ est une facette de MSIPND$(G, \mathcal{F})$, $b \geq 0$. En conséquence $\rho > 0$, ce qui termine la preuve du thérorème. □

Remarquons que les contraintes de partition ont uniquement des coefficients 0 et 1. Notre seconde classe de contraintes, décrite dans la section suivante, peut avoir des coefficients différents de 0 et de 1.

3.3.3 Contraintes de coupe-cycle

Dans cette section, nous introduisons une nouvelle classe d'inégalités valides pour MSIPND$(G, \mathcal{F})$. Ces inégalités sont induites par un sous-ensemble d'arêtes d'une coupe, et peuvent avoir des coefficients arbitraires aussi grands que l'on veut.

Soient $W \subset V$ et $T_1 = \{e_1, \ldots, e_s\}$, $s \geq 3$, un ensemble d'arêtes de la coupe $\delta_G(W)$. Soit $1 \leq q < s$ un entier. On suppose que pour tout $i = 1, \ldots, s$, il existe $j_i \in \{1, \ldots t\}$ tel que $F_{j_i} \cap T_1 = \{e_i, \ldots, e_{i+q-1}\}$ (les indices sont modulo s). Soit $T_2 = \delta_G(W) \backslash (T_1 \cup (\bigcap_{i=1,\ldots,s} F_{j_i}))$. Une telle configuration sera appelée une *configuration de coupe-cycle* (voir figure 3.12).

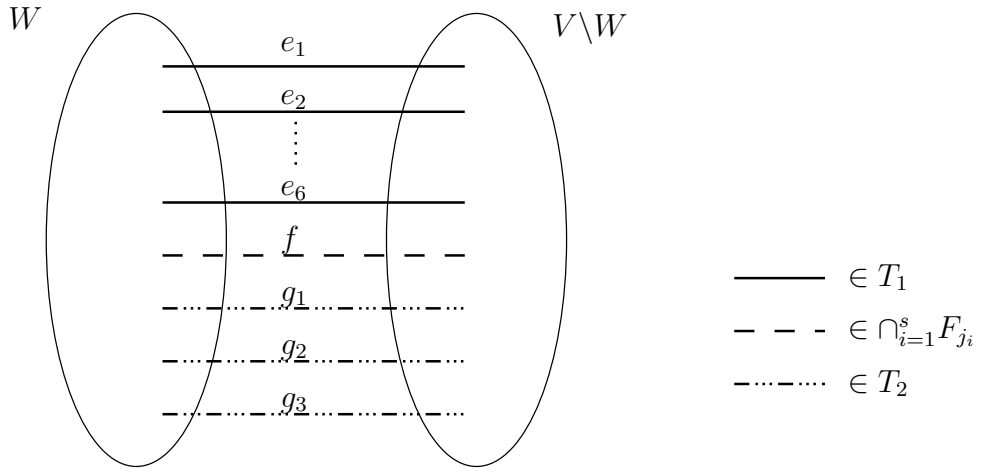


FIG. 3.12 – Une configuration de coupe-cycle

Considérons, par exemple, la configuration de coupe-cycle donnée dans la figure 3.12 où $\delta(W) = \{e_1, e_2, \ldots, e_6, f, g_1, g_2, g_3\}$. Soit $T_1 = \{e_1, e_2, \ldots, e_6\}$. On remarque que $T_1 \subset \delta(W)$. On suppose qu'il existe un sous-ensemble d'arêtes F_{j_i}, $i = 1, \ldots, 6$ tels que $\{e_i, e_{i+1}, f\} = F_{j_i} \cap \delta(W)$ pour $i = 1, \ldots, 6$ (les indices sont modulo 6). On a alors $F_{j_i} \cap T_1 = \{e_i, e_{i+1}\}$ pour $i = 1, \ldots, 6$. On remarque que $f \in \cap_{i=1}^{6} F_{j_i}$ et donc $f \notin T_2$. En fait, $T_2 = \{g_1, g_2, g_3\}$.

Théorème 3.8 *Soit (W, T_1, T_2) une configuration de coupe-cycle. Pour $e \in \delta_G(W)$, soient $r_e = |\{i \in \{1, \ldots, s\} \,|\, e \in \delta_G(W) \backslash F_{j_i}\}|$, et r le plus petit entier tel que $r(s-q) \geq \max_{e \in T_2}\{r_e\}$. Alors, l'inégalité*

$$x(T_1) + rx(T_2) \geq \left\lceil \frac{s}{s-q} \right\rceil \tag{3.11}$$

est valide pour MSIPND$(G, \mathcal{F})$.

Preuve. Les inégalités suivantes sont valides pour MSIPND$(G, \mathcal{F})$.

$$x(\delta_{G_{j_i}}(W)) \geq 1 \qquad \text{pour tout } i = 1, \ldots, s, \tag{3.12}$$

$$(r(s-q) - r_e)x(e) \geq 0 \qquad \text{pour tout } e \in T_2. \tag{3.13}$$

En sommant ces inégalités, on obtient

$$(s-q)x(T_1) + r(s-q)x(T_2) \geq s.$$

En divisant cette inégalité par $s - q$ et en arrondissant le membre de droite à l'entier supérieur, on obtient l'inégalité (3.11). □

Les contraintes (3.11) seront appelées *contraintes de coupe-cycle.*

Pour l'exemple de la figure 3.12, on remarque que $g_1, g_2, g_3 \notin \cup_{i=1}^{6} F_{j_i}$ et par conséquent, $r = 2$. La contrainte de coupe-cycle correspondant à cette configuration est donc donnée par :

$$x(T_1) + 2x(T_2) \geq 2.$$

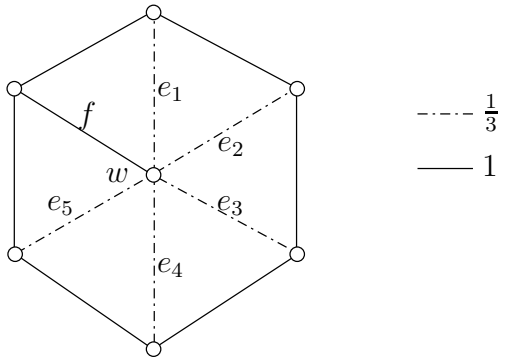


FIG. 3.13 –

De même, considérons le graphe $G = (V, E)$ donné par la figure 3.13. Soit $\mathcal{F} = \{F_1, \ldots, F_5\}$ tel que $F_i = \{f, e_i, e_{i+1}\}$ (les indices sont modulo 5). Soit $W = \{w\}$. Soit $T_1 = \{e_1, \ldots, e_5\}$. On remarque que $F_i \subset \delta(W)$ pour $i = 1, \ldots, 5$ et $T_2 = \emptyset$. D'après le théorème 3.8, l'inégalité $x(T_1) \geq 2$ est valide pour MSIPND$(G, \mathcal{F})$. On peut également voir que cette contrainte de coupe-cycle coupe le point $\bar{x}$ donné par $\bar{x}(e_i) = \frac{1}{3}$ pour $i = 1, \ldots, 5$, $\bar{x}(f) = 1$ et $\bar{x}(e) = 1$ sinon, qui est un point extrême du polytope défini par les contraintes de coupe et les contraintes de partition.

Dans ce qui suit, nous nous restreindrons au cas où $q = 1$ et $\delta_G(W) \backslash (T_1 \cup T_2) = \emptyset$. Nous donnons des conditions nécessaires et suffisantes pour que les inégalités (3.11)

définissent des facettes dans ce cas. En résolvant le MSIPND$(G, \mathcal{F})$ par un algorithme de coupes, nous avons remarqué que la plupart des contraintes de coupe-cycle violées sont de ce type.

Théorème 3.9 *Supposons que $q = 1$ et $\delta_G(W)\backslash(T_1 \cup T_2) = \emptyset$. Alors l'inégalité (3.11) définit une facette de MSIPND$(G, \mathcal{F})$ si et seulement si les conditions suivantes sont satisfaites.*

1) T_1 est maximal, c'est-à-dire que pour toute arête e de T_2, il existe $i \in \{1, \ldots, s\}$ tel que $e \in F_{j_i}$. (On peut noter que dans ce cas on a $r = 1$.)

2) $G(W)$ et $G(\overline{W})$ sont $\mathcal{F}$-connexes.

3) Si $G(W)$ (resp. $G(\overline{W})$) n'est pas 2-$\mathcal{F}$-connexe, alors pour toute arête $e \in E_G(W)$ telle que $G(W)\backslash e$ (resp. $G(\overline{W})\backslash e$) n'est pas $\mathcal{F}$-connexe, il existe deux arêtes g_1, $g_2 \in \delta_G(W)$ telles que

3.1) $g_1, g_2 \notin F_i$, pour tout $i \in I_e$,

3.2) $|F_j \cap \{g_1, g_2\}| \leq 1$ pour tout $j \in \{1, \ldots, t\}\backslash I_e$,

3.3) pour tout sous-ensemble W_1 de W (resp. W_1' de $\overline{W}$) tel que $\delta_G(W_1, W\backslash W_1) \subset F_i \cup \{e\}$ (resp. $\delta_G(W_1', \overline{W}\backslash W_1') \subset F_i \cup \{e\}$) pour un certain $i \in I_e$, on a $|\delta_G(W_1, \overline{W}) \cap \{g_1, g_2\}| = 1$ (resp. $|\delta_G(W_1', \overline{W}) \cap \{g_1, g_2\}| = 1$).

Ici I_e représente l'ensemble $\{i \in \{1, \ldots, t\} \mid G_i(W)\backslash e$ n'est pas connexe$\}$ (resp. $\{i \in \{1, \ldots, t\} \mid G_i(\overline{W})\backslash e$ n'est pas connexe$\}$.

Preuve. *Conditions nécessaires* : 1) Supposons qu'il existe une arête f de T_2 qui n'appartient à aucun ensemble F_{j_i}, $i = 1, \ldots, s$. Donc $r_f = s$. On a donc $\max_{e \in T_2}\{r_e\} = r_f = s$. Comme $q = 1$, on a r le plus petit entier tel que $r(s-1) \geq s$. Il s'ensuit que $r = 2$ et donc la contrainte de coupe-cycle correspondant à T_1 et T_2, dans ce cas, peut être écrite de la façon suivante.

$$x(T_1) + 2x(T_2) \geq 2. \tag{3.14}$$

Comme toutes les arêtes de E appartiennent à un ensemble F_i, il existe $l \in \{1, \ldots, t\}\backslash \{j_1, \ldots, j_s\}$ tel que $f \in F_l$. Considérons les ensembles $T_1' = T_1 \cup \{f\}$ et $T_2' = T_2\backslash\{f\}$. Considérons les inégalités valides

$$\begin{array}{ll} x(\delta_{G_l}(W)) \geq 1, & \\ x(e) \geq 0 & \text{pour tout } e \in (T_1 \cap F_l), \\ 2x(e) \geq 0 & \text{pour tout } e \in (T_2 \cap F_l)\backslash\{f\}, \\ x(f) \geq 0. & \end{array}$$

En sommant ces inégalités, les inégalités (3.12) et les inégalités (3.13) pour tout $e \in T_2 \backslash \{f\}$, on obtient

$$sx(T_1') + 2sx(T_2') \geq s + 1.$$

(On rappelle que $r = 2$.) En divisant par s et en arrondissant le membre de droite à l'entier supérieur, on obtient l'inégalité

$$x(T_1') + 2x(T_2') \geq 2, \tag{3.15}$$

qui est donc valide pour MSIPND$(G, \mathcal{F})$. Comme cette inégalité domine l'inégalité (3.14), la contrainte (3.15) ne peut pas définir de facette. Donc T_1 doit être maximal. Pour tout $e \in T_2$, il existe alors $i \in \{1, \ldots, s\}$ tel que $e \in F_{j_i}$. D'où $r_e \leq s - 1$. Comme $q = 1$, on a $r(s-1) \geq s - 1$. Il s'ensuit que $r = 1$.

2) Supposons que pour $i \in \{1, \ldots, t\}$, on a par exemple, $G_i(W)$ non connexe. Alors il existe une partition W_1, W_2 de W telle que $\delta_{G_i}(W_1, W_2) = \emptyset$ (voir figure 3.14).

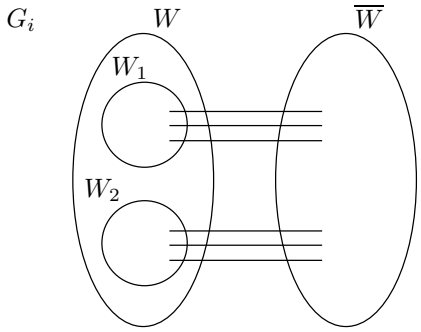


FIG. 3.14 –

Donc $\delta_{G_i}(W_1) = \delta_{G_i}(W_1, \overline{W})$ et $\delta_{G_i}(W_2) = \delta_{G_i}(W_2, \overline{W})$. Il en résulte que

$$\begin{aligned} x(\delta_{G_i}(W)) &= x(\delta_{G_i}(W_1, \overline{W})) + x(\delta_{G_i}(W_2, \overline{W})) \\ &= x(\delta_{G_i}(W_1)) + x(\delta_{G_i}(W_2)) \\ &\geq 2. \end{aligned}$$

Comme $\delta_{G_i}(W) \subseteq T_1 \cup T_2$ et $x(e) \geq 0$ pour tout $e \in E$, on a, en conséquence, $x(T_1) + x(T_2) \geq 2$. Mais cette inégalité domine (3.14), et donc (3.14) ne définit pas une facette.

3) Nous allons prouver l'assertion pour $G(W)$, la preuve pour $G(\overline{W})$ étant similaire. Supposons donc que $G(W)$ n'est pas 2-$\mathcal{F}$-connexe. Supposons que l'assertion n'est pas vérifiée pour une arête $e \in E_G(W)$. Nous allons montrer que toute solution T de $\mathcal{S}(G, \mathcal{F})$ telle que $|T \cap \delta_G(W)| = 2$ contient e. En effet, supposons le contraire, c'est-à-dire $e \notin T$.

Soient g_1, g_2 des arêtes de $\delta_G(W)$ dans T. Alors g_1, g_2 ne satisfont pas au moins une des conditions 3.1), 3.2), 3.3).

- Si g_1, g_2 ne satisfont pas 3.1), alors il existe $i \in I_e$ tel que, disons, $g_1 \in F_i$. Donc il

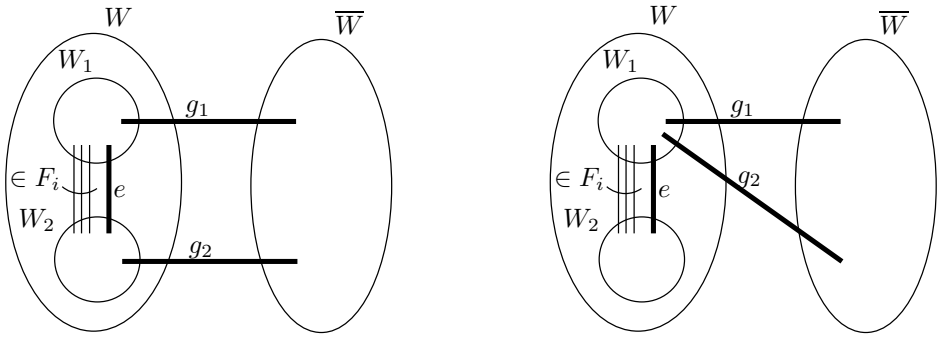


FIG. 3.15 –

existe $W_1 \subset W$ tel que $\delta_G(W_1, W\backslash W_1) \cap T \subseteq F_i \cup \{e\}$. S.p.d.g., on peut supposer que $g_1 \in \delta_G(W_1, \overline{W})$. Si $g_2 \in \delta_G(W\backslash W_1, \overline{W})$ (resp. $g_2 \in \delta_G(W_1, \overline{W})$), alors $\delta_{G_i}(W_1) \cap T = \emptyset$ (resp. $\delta_{G_i}(W\backslash W_1) \cap T = \emptyset$) (voir figure 3.15 qui présente ces deux cas de figures). Donc le graphe induit par $T\backslash F_i$ n'est pas connexe, une contradiction.

- S'il existe $j \in \{1, \ldots, t\}\backslash I_e$ tel que $g_1, g_2 \in F_j$ alors il est clair que $T\backslash F_j$ n'induit pas un graphe connexe, une contradiction.

- Maintenant, supposons que g_1, g_2 ne satisfont pas 3.3). Alors, on peut supposer qu'il existe une partition $(W_1, W\backslash W_1)$ de W et $i \in I_e$ tel que $\delta_G(W_1, W\backslash W_1) \subset F_i \cup \{e\}$ et $\delta_G(W_1, \overline{W})$ soit contient $\{g_1, g_2\}$, soit n'intersecte pas cet ensemble. Si $\{g_1, g_2\} \subset \delta_G(W_1, \overline{W})$ (resp. $\{g_1, g_2\} \cap \delta_G(W_1, \overline{W}) = \emptyset$), alors $\delta_{G_i}(W_1) \cap T = \emptyset$ (resp. $\delta_{G_i}(W\backslash W_1) \cap T = \emptyset$). Donc $T \notin \mathcal{S}(G, \mathcal{F})$, une contradiction.

Ainsi pour tout $T \in \mathcal{S}(G, \mathcal{F})$ tel que $|T \cap \delta_G(W)| = 2$, on a $e \in T$. Ceci implique que la face définie par (3.14) est contenue dans $\{x \in \text{ MSIPND}(G, \mathcal{F}) \mid x(e) = 1\}$. Comme (3.14) n'est pas une contrainte triviale, il s'ensuit que (3.14) ne définit pas une facette.

Conditions suffisantes : Supposons que $\delta_G(W)\backslash(T_1 \cup T_2) = \emptyset$ et $q = 1$. Supposons également que les conditions 1), 2) et 3) sont satisfaites. Par 1) l'inégalité (3.11) peut être écrite sous la forme

$$x(T_1) + x(T_2) \geq 2.$$

Notons l'inégalité (3.11) par $ax \geq \alpha$, et soit $bx \geq \beta$ une inégalité définissant une facette de MSIPND$(G, \mathcal{F})$ telle que $\{x \in \text{ MSIPND}(G, \mathcal{F}) \mid ax = \alpha\} \subseteq \{x \in \text{ MSIPND}(G, \mathcal{F}) \mid bx = \beta\}$. Pour montrer que $ax \geq \alpha$ définit une facette, nous allons montrer comme précédemment qu'il existe $\rho > 0$ tel que $b = \rho a$.

Comme $s \geq 3$, soient e_1, e_2 deux arêtes de T_1. Soit

$$\Gamma_0 = \{e_1, e_2\} \cup E_G(W) \cup E_G(\overline{W}).$$

Comme e_1 et e_2 appartiennent à T_1 et $q = 1$, e_1 et e_2 appartiennent à des ensembles F_i différents, et comme d'après la condition 2) les deux graphes $G(W)$ et $G(\overline{W})$ sont $\mathcal{F}$-connexes, il s'ensuit que $\Gamma_0 \in \mathcal{S}(G, \mathcal{F})$. Considérons l'ensemble d'arêtes $\Gamma_i = (\Gamma_0 \backslash \{e_1\}) \cup \{e_i\}$ pour $i = 3, \ldots, s$. Il est clair que, comme les arêtes e_i, $i = 1, \ldots, s$ appartiennent à des ensembles F_j différents, alors $\Gamma_i \in \mathcal{S}(G, \mathcal{F})$ pour $i = 3, \ldots, s$. De plus, on a $ax^{\Gamma_0} = ax^{\Gamma_i} = \alpha$. Donc $bx^{\Gamma_i} = bx^{\Gamma_0} = \beta$ et en conséquence $b(e_i) = b(e_1)$ pour $i = 3, \ldots, s$. Par symétrie, on a donc

$$b(e) = \rho \text{ pour } \rho \in I\!R, \text{ pour tout } e \in T_1. \tag{3.16}$$

Maintenant, considérons une arête $f \in T_2$. D'après la condition 1), il existe $i \in \{1, \ldots, s\}$ tel que $f \in F_{j_i}$. Considérons l'ensemble $\Gamma_f = (\Gamma_i \backslash \{e_i\}) \cup \{f\}$ (resp. $\Gamma_f = (\Gamma_0 \backslash \{e_i\}) \cup \{f\}$) si $i \in \{3, \ldots, s\}$ (resp. $i \in \{1, 2\}$) où Γ_0 et Γ_i sont les ensembles d'arêtes introduits précédemment. Il est clair que $ax^{\Gamma_f} = ax^{\Gamma_i} = ax^{\Gamma_0} = \alpha$. Donc $bx^{\Gamma_f} = bx^{\Gamma_i} = bx^{\Gamma_0} = \beta$, et donc $b(f) = b(e_i)$. Ceci ajouté à (3.16) donne

$$b(e) = \rho \text{ pour tout } e \in T_1 \cup T_2. \tag{3.17}$$

Considérons une arête $e \in E_G(W)$. Si $G(W) \backslash e$ est $\mathcal{F}$-connexe, alors $\Gamma_0 \backslash \{e\}$ est solution de $\mathcal{S}(G, \mathcal{F})$, et donc $b(e) = 0$. Sinon, d'après 3), il existe deux arêtes g_1, g_2 satisfaisant les conditions 3.1), 3.2), 3.3). Soit $\Gamma'_0 = (\Gamma_0 \backslash \{e_1, e_2, e\}) \cup \{g_1, g_2\}$. On va montrer que $\Gamma'_0 \in \mathcal{S}(G, \mathcal{F})$. Dans la suite, nous noterons H_i le sous-graphe de G induit par $\Gamma'_0 \backslash F_i$, pour $i \in \{1, \ldots, t\}$.
Soit F_i tel que $i \in I_e$ (où I_e est défini dans la condition 3)). Par 3.1), $g_1, g_2 \notin F_i$. Comme $G(\overline{W})$ est $\mathcal{F}$-connexe, si H_i n'est pas connexe, alors il doit exister $W_1 \subset W$, tel que $\delta_{H_i}(W_1) = \emptyset$. Comme $E_G(W) \backslash \{e\} \subset \Gamma'_0$, il s'ensuit que $\delta_G(W_1, W \backslash W_1) \subset F_i \cup \{e\}$ et $g_1, g_2 \in \delta_G(W \backslash W_1, \overline{W})$. Mais cela contredit 3.3).
Maintenant, considérons un ensemble F_i avec $i \in \{1, \ldots, t\} \backslash I_e$. Alors $G_i(W) \backslash e$ est connexe. Comme d'après 3.2), $\{g_1, g_2\} \backslash F_i \neq \emptyset$, et $G(\overline{W})$ est $\mathcal{F}$-connexe, on a H_i connexe. Donc $\Gamma'_0 \in \mathcal{S}(G, \mathcal{F})$.
On a aussi $ax^{\Gamma'_0} = \alpha$. Donc $bx^{\Gamma'_0} = \beta$. Comme $bx^{\Gamma_0} = \beta$ et d'après (3.17), $b(g_1) = b(g_2) = b(e_1) = b(e_2)$, on obtient $b(e) = 0$.
Donc $b(e) = 0$ pour tout $e \in E_G(W)$. D'une façon similaire on montre que $b(e) = 0$ pour tout $e \in E_G(\overline{W})$. Par conséquent, d'après (3.17), il en résulte que $b = \rho a$. De plus, on a $\rho > 0$, et donc la preuve est complète. □

Considérons le graphe de la figure 3.16 et les ensembles d'arêtes $F_i = \{e_{i+5}\}$ pour $i = 1, \ldots, 5$. Soit $\bar{x}$ la solution donnée par $\bar{x}(e_i) = \frac{1}{2}$ pour $i = 1, \ldots, 5$ et $\bar{x}(e_i) = 1$ pour $i = 6, \ldots, 10$. Il est clair que $\bar{x}$ satisfait les contraintes triviales et les contraintes de coupe par rapport à $F_1, \ldots, F_5$. La solution $\bar{x}$ satisfait également les contraintes de

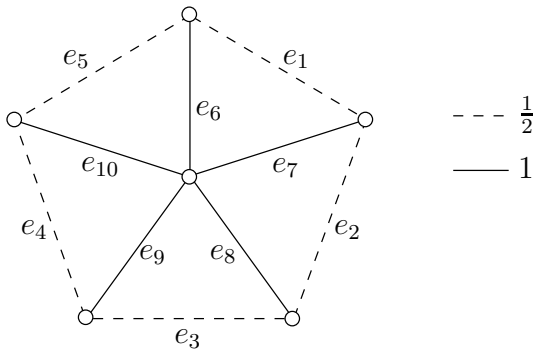


FIG. 3.16 – Un point extrême fractionnaire

partition et les contraintes de coupe-cycle. De plus, il n'est pas difficile de voir que $\bar{x}$ est un point extrême du polytope donné par ces inégalités. Néanmoins, $\bar{x}$ viole l'inégalité

$$x(e_1) + x(e_2) + x(e_3) + x(e_4) + x(e_5) \geq 3,$$

qui est valide pour le polytope associé. Dans ce qui suit, nous allons montrer que cette inégalité fait partie d'une classe plus générale d'inégalités valides pour MSIPND$(G, \mathcal{F})$.

3.3.4 Contraintes d'étoile-partition

Soient $G = (V, E)$ un graphe et $\mathcal{F} = \{F_1, \ldots, F_t\}$, $t \geq 2$, une famille de sous-ensembles d'arêtes de E. Soit $V_0, V_1, \ldots, V_p$ une partition de V avec p impair. Supposons que pour tout $i = 1, \ldots, p$, il existe $j_i \in \{1, \ldots, t\}$ tel que $F_{j_i} \cap \delta_G(V_i, V_0) \neq \emptyset$. Soit $\Lambda = \{e \in E \mid e \in \delta_G(V_k, V_l) \cap F_{j_k} \cap F_{j_l}$, pour $k, l \in \{1, \ldots, p\}\}$. Soit $F = \bigcup_{i=1}^{p} (F_{j_i} \cap \delta_G(V_i, V_0)) \cup \Lambda$. Une telle configuration sera appelée une *configuration d'étoile-partition* (voir figure 3.17).

Par exemple, considérons la configuration d'étoile-partition donnée par la figure 3.17. La partition correspondante a six éléments $V_0, V_1, \ldots, V_5$. A chaque V_i est associé un ensemble d'arêtes F_{j_i} tel que $F_{j_i} \cap \delta_G(V_i, V_0) \neq \emptyset$ pour $i = 1, \ldots, 5$. Ces ensembles sont représentés par différents types de lignes. En effet, $e_1 \in F_{j_1} \cap \delta_G(V_1, V_0)$, $e_6 \in F_{j_2} \cap \delta_G(V_2, V_0)$, $e_5 \in F_{j_3} \cap \delta_G(V_3, V_0)$, $e_3, e_4 \in F_{j_4} \cap \delta_G(V_4, V_0)$ et $e_2 \in F_{j_5} \cap \delta_G(V_5, V_0)$. Notons que l'arête e_7 entre V_1 et V_2 appartient à la fois à l'ensemble F_{j_1} et à l'ensemble F_{j_2}, et donc $e_7 \in \Lambda$. On peut voir de plus que $\Lambda = \{e_7\}$.

Considérons l'inégalité

$$x(\delta_G(V_0, \ldots, V_p) \backslash F) \geq \left\lceil \frac{p}{2} \right\rceil . \tag{3.18}$$

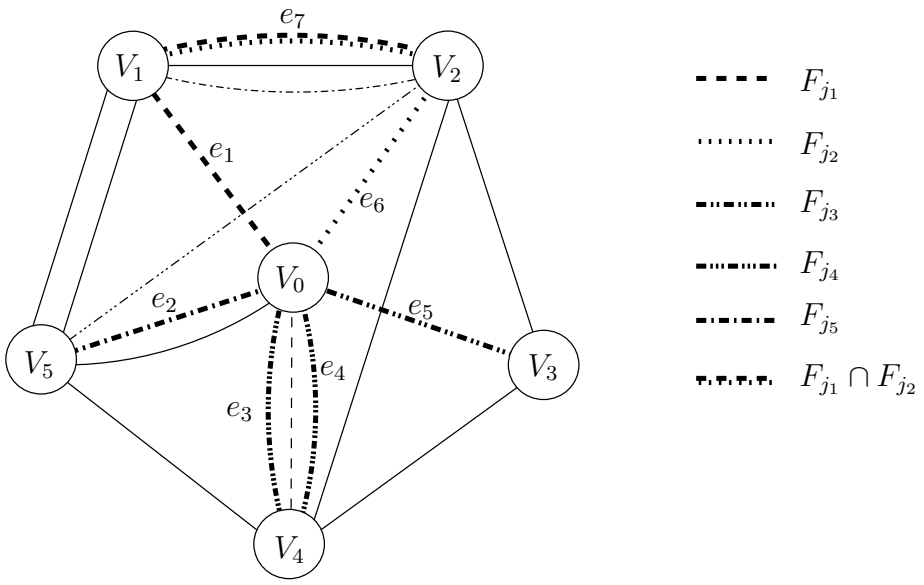


FIG. 3.17 – Une configuration d'étoile-partition

Théorème 3.10 *L'inégalité (3.18) est valide pour MSIPND$(G, \mathcal{F})$.*

Preuve. Il est clair que les inégalités suivantes sont valides pour MSIPND$(G, \mathcal{F})$.

$$\begin{aligned} x(\delta_{G\setminus F_{j_i}}(V_i)) &\geq 1 \quad \text{pour tout } i = 1, \ldots, p, \\ x(e) &\geq 0 \quad \text{pour tout } e \in \delta(V_0)\setminus F, \\ x(e) &\geq 0 \quad \text{pour tout } e \in (\delta_G(V_k, V_l) \cap F_{j_k})\setminus F_{j_l},\ k = 1, \ldots, p,\ l = 1, \ldots, p,\ k \neq l. \end{aligned}$$

En sommant ces inégalités, on obtient l'inégalité

$$2x(\delta(V_0, \ldots, V_p)\setminus F) \geq p.$$

En divisant par 2 et en arrondissant le membre de droite à l'entier supérieur, on obtient l'inégalité (3.18). □

Les inégalités (3.18) seront appelées *contraintes d'étoile-partition.*

Par exemple, la contrainte d'étoile-partition correspondant à la configuration de la figure 3.17 est donnée par

$$x(\delta(V_0, V_1, ..., V_5)\setminus F) \geq 3,$$

où $F = \{e_1, \ldots, e_6, e_7\}$.

Des inégalités similaires à (3.18) appelées contraintes de F-partition ont été introduites dans [63] pour le problème du sous-graphe 2-arête connexe (voir section 1.2.1.2).

On définit à présent quelques notations qui vont être utiles pour définir les conditions nécessaires et les conditions suffisantes pour que les contraintes (3.18) définissent des facettes. Soient $R_{j_i} = F_{j_i} \cap \delta_G(V_i, V_0)$ pour $i = 1, \ldots, p$ et $\Delta = F \cup (\bigcup_{i=0,\ldots,p} E_G(V_i))$. Par exemple, pour la configuration d'étoile-partition de la figure 3.17, on a $R_{j_1} = \{e_1\}$, $R_{j_2} = \{e_6\}$, $R_{j_3} = \{e_5\}$, $R_{j_4} = \{e_3, e_4\}$ et $R_{j_5} = \{e_2\}$.

Théorème 3.11 *L'inégalité (3.18) définit une facette de MSIPND$(G, \mathcal{F})$ seulement si les assertions suivantes sont vérifiées.*

1) Il existe un ensemble d'arêtes $\widetilde{T} \subseteq \delta_G(V_1, \ldots, V_p)$ tel que $\widetilde{T} \cup \Delta \in \mathcal{S}(G, \mathcal{F})$ et $|\widetilde{T}| = \frac{p+1}{2}$.

2) Pour toute arête $e \in \delta_G(V_i, V_i')$, $i, i' \in \{1, \ldots, p\}$, il n'existe pas $i_0 \in \{1, \ldots, t\}$ tel que $(R_{j_i} \cup R_{j_{i'}} \cup \{e\}) \subseteq F_{i_0}$.

Preuve. Supposons que (3.18) définit une facette de MSIPND$(G, \mathcal{F})$. Il est clair qu'il existe une solution $T \in \mathcal{S}(G, \mathcal{F})$ telle que $|T \cap (\delta_G(V_0, \ldots, V_p) \backslash F)| = \frac{p+1}{2}$. Sinon, la face définie par (3.18) serait vide, et donc ne pourrait pas être une facette. Soit $T' = T \cap (\delta_G(V_0, \ldots, V_p) \backslash F)$. Supposons que $T' \cap \delta_G(V_0) \neq \emptyset$. On montre que $|T' \cap \delta_G(V_0)| = 1$

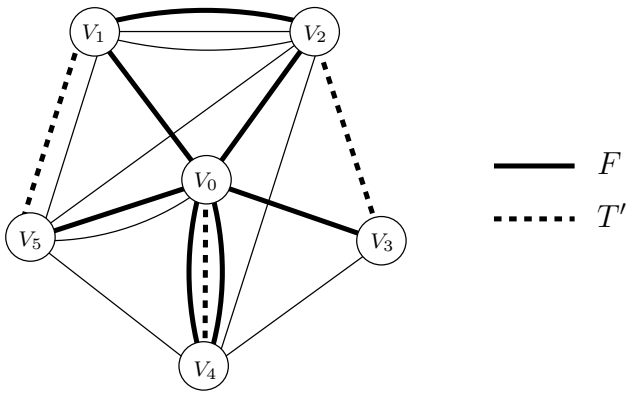


FIG. 3.18 –

(voir figure 3.18). En effet, supposons que ce n'est pas le cas et soient f_1, f_2 deux arêtes de $T' \cap \delta_G(V_0)$. Soient $i_1, i_2 \in \{1, \ldots, p\}$ tels que $f_1 \in \delta_G(V_{i_1}, V_0)$ et $f_2 \in \delta_G(V_{i_2}, V_0)$, et supposons, s.p.d.g., que $i_1 \neq i_2$. Comme $T \in \mathcal{S}(G, \mathcal{F})$, et donc $G[T \backslash F_{j_i}]$ est connexe pour $i = 1, \ldots, p$, on a $T \cap (\delta_G(V_i) \backslash F_{j_i}) \neq \emptyset$ pour tout $i \in \{1, \ldots, p\} \backslash \{i_1, i_2\}$. Comme $T \cap (\delta_G(V_i) \backslash F_{j_i}) = T' \cap (\delta_G(V_i) \backslash F_{j_i})$, on a donc $T' \cap (\delta_G(V_i) \backslash F_{j_i}) \neq \emptyset$ pour tout

$i \in \{1,\ldots,p\}\backslash\{i_1,i_2\}$. On a donc $|T'| \geq 2 + \frac{p-2}{2}$. Comme p est impair, cela implique que $|T'| \geq \frac{p+3}{2}$, une contradiction.
En conséquence, toute solution de $\mathcal{S}(G,\mathcal{F})$, dont le vecteur d'incidence satisfait (3.18) à l'égalité, contient au plus une arête de $\delta_G(V_0)\backslash F$. Si 1) n'est pas satisfaite, alors toute solution de $\mathcal{S}(G,\mathcal{F})$ dont le vecteur d'incidence satisfait (3.18) à l'égalité contient exactement une arête de $\delta_G(V_0)\backslash F$. Mais cela implique que la face définie par (3.18) est contenue dans l'hyperplan $\sum_{e\in\delta_G(V_0)\backslash F} x(e) = 1$. Mais, ceci contredit le fait que (3.18) définit une facette.
Maintenant considérons une arête $e \in \delta_G(V_i,V_{i'})$ pour $i,i' \in \{1,\ldots,p\}$. Comme (3.18) est différente des inégalités $x(e) \geq 0$, il doit exister une solution, notée T_1, de $\mathcal{S}(G,\mathcal{F})$ contenant e et dont le vecteur d'incidence satisfait (3.18) à l'égalité. Soit $\widetilde{T}_1 = T_1\backslash\Delta$. Comme $|\widetilde{T}_1| = \frac{p+1}{2}$, e peut être adjacent à au plus une arête de $\widetilde{T}_1\backslash\{e\}$ et $\delta_G(V_i,V_{i'}) \cap \widetilde{T}_1 = \{e\}$. Si $R_{j_i} \cup R_{j_{i'}} \cup \{e\} \subseteq F_{i_0}$ pour $i_0 \in \{1,\ldots,t\}$, alors $G[T_1\backslash F_{i_0}]$ n'est pas connexe, ce qui est impossible. □

Théorème 3.12 *L'inégalité (3.18) définit une facette de MSIPND$(G,\mathcal{F})$ si*

1) *la condition 1) du théorème 3.11 est satisfaite,*
2) *$G(V_i)$ est 2-$\mathcal{F}$-connexe pour $i = 0,1,\ldots,p$,*
3) *pour tout $i,i' \in \{1,\ldots,p\}$, $e \in R_{j_i}$ et $l \in \{1,\ldots,t\}$, $(R_{j_i} \cup R_{j_{i'}})\backslash(F_l \cup \{e\}) \neq \emptyset$,*
4) *pour tout $e \in \delta_G(V_i,V_{i'})$, $i,i' \in \{1,\ldots,p\}$, il n'existe pas $i_0 \in \{1,\ldots,t\}$ avec soit $F_{j_i} \cup \{e\} \subseteq F_{i_0}$, soit $F_{j_{i'}} \cup \{e\} \subseteq F_{i_0}$.*

Preuve. Notons l'inégalité (3.18) par $ax \geq \alpha$ et soit $bx \geq \beta$ une inégalité définissant une facette de MSIPND$(G,\mathcal{F})$ telle que $\{x \in$ MSIPND$(G,\mathcal{F}) \mid ax = \alpha\} \subseteq \{x \in$ MSIPND$(G,\mathcal{F}) \mid bx = \beta\}$. Comme nous l'avons fait précédemment, nous allons montrer que $b = \rho a$ pour $\rho > 0$.
Soit $\widetilde{G} = (\widetilde{V},\widetilde{E})$ le graphe obtenu en contractant les ensembles $V_0,\ldots,V_p$. Soient $p = 2k+1$, $k \geq 1$. D'après la condition 1) du théorème 3.11, il existe un ensemble d'arêtes $\widetilde{T} \subseteq \delta_G(V_1,\ldots,V_p)$ tel que $|\widetilde{T}| = k+1$ et $T_0 = \widetilde{T} \cup \Delta \in \mathcal{S}(G,\mathcal{F})$. Donc $\widetilde{T} \cap \delta_G(V_i) \neq \emptyset$ pour $i = 1,\ldots,2k+1$. Après d'éventuelles permutations de l'ensemble $V_1,\ldots,V_{2k+1}$, on peut supposer que $\widetilde{T} = \{e_1,e_2,\ldots,e_{k+1}\}$ tel que $e_i \in \delta_G(V_{2i-1},V_{2i})$ pour $i = 1,2,\ldots,k+1$ (où les indices sont pris modulo $2k+1$). On peut remarquer que $\widetilde{T} \cap \delta(V_1) = \{e_1,e_{k+1}\}$ et les arêtes de $\widetilde{T}\backslash\{e_1\}$ sont deux à deux non adjacentes (voir figure 3.19). Soit e une arête de $\delta_G(V_2)\backslash F_{j_2}$ entre V_2 et V_l, $l \neq 2$. Considérons l'ensemble d'arêtes $T_e = (T_0\backslash\{e_1\}) \cup \{e\}$. On va montrer que $T_e \in \mathcal{S}(G,\mathcal{F})$.
Comme par 2) $G(V_i)$ est 2-$\mathcal{F}$-connexe et donc $\mathcal{F}$-connexe, alors pour $i = 0,1,\ldots,2k+1$, il suffit de montrer que $\widetilde{G}[T_e\backslash F_j]$ est connexe pour $j = 1,\ldots,t$.

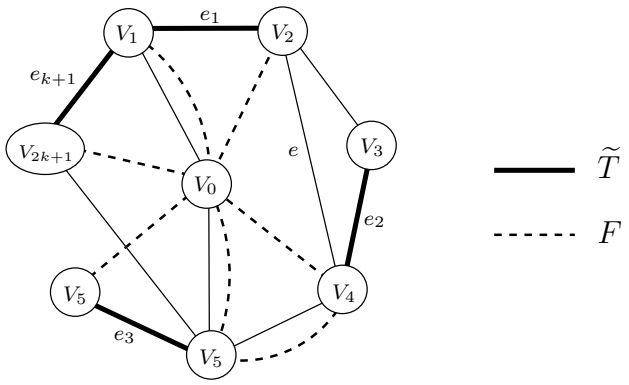


FIG. 3.19 –

Considérons un sommet V_i, $i \in \{1,\ldots,p\}$. Soit f une arête de $T_e\backslash\Delta$ incidente à V_i. Notons que $f \in (\widetilde{T}\backslash\{e_1\}) \cup \{e\}$. Soit i' tel que $f \in \delta_G(V_i, V_{i'})$. D'après 3), on a $(R_{j_i} \cup R_{j_{i'}})\backslash F_j \neq \emptyset$. Et, par 4) on a $(F_{j_i} \cup \{f\})\backslash F_j \neq \emptyset \neq (F_{j_{i'}} \cup \{f\})\backslash F_j$. Donc dans $\widetilde{G}[T_e\backslash F_j]$, V_i est relié à V_0 par une chemin consistant en une ou deux arêtes.
En conséquence, dans $\widetilde{G}[T_e\backslash F_j]$ tous les sommets V_i, $i = 1,\ldots,p$ sont reliés à V_0. Donc $\widetilde{G}[T_e\backslash F_j]$ est connexe, et T_e est solution de $\mathcal{S}(G,\mathcal{F})$.
Comme $ax^{T_0} = ax^{T_e}$, on a $bx^{T_0} = bx^{T_e}$ et donc $b(e_1) = b(e)$. Comme e est une arête de $\delta_G(V_2)\backslash F_{j_2}$ choisie arbitrairement, il s'ensuit qu'il existe ρ_2 tel que

$$b(e) = \rho_2 \quad \text{pour tout } e \in \delta_G(V_2)\backslash F_{j_2}. \tag{3.19}$$

Considérons un sommet V_i, $i \in \{1,\ldots,p\}\backslash\{2\}$, et soit $i' \in \{1,\ldots,p\}\backslash\{i\}$ tel que $\delta_G(V_i, V_{i'}) \cap (\widetilde{T}\backslash\{e_1\}) \neq \emptyset$. Notons que soit $i' = i+1$, soit $i' = i-1$. Comme G est complet, $\delta_G(V_{i'}, V_2) \neq \emptyset$. Soit g une arête de $\delta(V_{i'}, V_2)$, et considérons l'ensemble d'arêtes $T_g = (T_0\backslash\{e_1\}) \cup \{g\}$. Par le précédent développement, $T_g \in \mathcal{S}(G,\mathcal{F})$. Remarquons également que T_g a une structure similaire à celle de T_0, c'est-à-dire $|T_g\backslash\Delta| = k+1$. De la même façon que précédemment, on peut montrer qu'il existe ρ_i tel que

$$b(e) = \rho_i \quad \text{pour tout } e \in \delta_G(V_i)\backslash F_{j_i}. \tag{3.20}$$

De (3.19) et (3.20), on peut déduire que $\rho_i = \rho$ pour $i = 1,\ldots,p$ et $\rho \in \mathbb{R}$, et donc $b(e) = \rho$ pour tout $e \in \delta_G(V_0,\ldots,V_p)\backslash F$.
Si e est une arête de $E_G(V_i)$, pour $i \in \{0,\ldots,p\}$, comme $G(V_i)$ est 2-$\mathcal{F}$-connexe, $T_0' = T_0\backslash\{e\}$ est solution de $\mathcal{S}(G,\mathcal{F})$ et donc $b(e) = 0$.
Maintenant soit e une arête de R_{j_i} pour $i \in \{1,\ldots,p\}$. On montre que $\overline{T}_0 = T_0\backslash\{e\} \in \mathcal{S}(G,\mathcal{F})$. Soit $i' \in \{1,\ldots,p\}\backslash\{i\}$ tel que $\delta_G(V_i, V_{i'}) \cap \widetilde{T} \neq \emptyset$. Soit $j \in \{1,\ldots,t\}$. D'après c) on a $(R_{j_i} \cup R_{j_{i'}})\backslash F_j \neq \emptyset$. Si $e \in F_j$, alors $G(\overline{T}_0\backslash F_j) = G(T_0\backslash F_j)$, et donc $G(\overline{T}_0\backslash F_j)$ est connexe. Supposons que $e \notin F_j$. Par c) $(R_{j_i} \cup R_{j_{i'}})\backslash(F_j \cup \{e\}) \neq \emptyset$. Alors dans

$\widetilde{G}[\widetilde{T}_0 \backslash F_j]$ les deux sommets V_i et $V_{i'}$ sont reliés à V_0. Comme $T_0 \in \mathcal{S}(G, \mathcal{F})$ et donc tous les autres sommets sont reliés à V_0, on a $\widetilde{G}[\overline{T}_0 \backslash F_j]$ connexe. En conséquence, on a $b(e) = 0$. D'une façon similaire on montre que $b(e) = 0$ pour tout $e \in \Lambda$.
On a donc montré que

$$b(e) = \begin{cases} \rho & \text{pour tout } e \in \delta_G(V_1, \ldots, V_p) \backslash F, \\ 0 & \text{pour tout } e \in \Delta. \end{cases}$$

D'où $b = \rho a$. Comme $bx \geq \beta$ définit une facette, on a également $\rho > 0$. □

3.4 Opérations de réduction

Soient $G = (V, E)$ un graphe et $\mathcal{F} = \{F_1, \ldots, F_t\} \subseteq 2^E$ avec $t \geq 2$ une famille d'ensembles d'arêtes de E. Considérons le polytope $P(G, \mathcal{F})$ défini par les contraintes triviales et les contraintes de coupe :

$$x(e) \geq 0 \qquad \text{pour tout } e \in E, \tag{3.21}$$

$$x(e) \leq 1 \qquad \text{pour tout } e \in E, \tag{3.22}$$

$$x(\delta_{G_i}(W)) \geq 1 \qquad \text{pour tout } W \subseteq V, \quad \emptyset \neq W \neq V, \quad i = 1, \ldots, t. \tag{3.23}$$

Puisque le problème de séparation des inégalités de $P(G, \mathcal{F})$ est polynomial, le problème MSIPND peut se résoudre en temps polynomial dans la classe des graphes G pour lesquels MSIPND$(G, \mathcal{F})$=$P(G, \mathcal{F})$.

Dans cette section, nous décrivons quelques opérations de réduction de graphes. Ces opérations, qui préservent la propriété MSIPND$(G, \mathcal{F})$=$P(G, \mathcal{F})$, peuvent être très utiles dans la résolution du problème par une méthode de coupes. Elles utilisent des idées similaires à celles développées par Kerivin et al. dans [55] pour le problème du sous-graphe 2-arête connexe.

Etant donnée une solution x de $P(G, \mathcal{F})$, notons par
- $E_0(x)$ l'ensemble des arêtes $e \in E$ telles que $x(e) = 0$,
- $E_1(x)$ l'ensemble des arêtes $e \in E$ telles que $x(e) = 1$,
- $E_f(x)$ l'ensemble des arêtes $e \in E$ telles que $0 < x(e) < 1$,
- $V_i(x)$ l'ensemble des sommets $v \in V$ tels que $x(\delta_{G_i}(v)) = 1$, pour $i = 1, \ldots, t$,
- $\tau_i(x)$ l'ensemble des coupes dans G_i serrées pour x, pour $i = 1, \ldots, t$.

Soit x un point extrême de $P(G,\mathcal{F})$. Alors il existe des ensembles $\tau'_i(x) \subset \tau_i(x)$ pour $i = 1,\ldots,t$ tels que x soit l'unique solution du système

$$S(x) = \begin{cases} x(e) = 0 & \text{pour tout } e \in E_0(x), \\ x(e) = 1 & \text{pour tout } e \in E_1(x), \\ x(\delta_{G\setminus F_i}(v)) = 1 & \text{pour tout } v \in V_i(x), i = 1,\ldots,t, \\ x(\delta_{G\setminus F_i}(W)) = 1 & \text{pour tout } \delta_{G\setminus F_i}(W) \in \tau'_i(x), i = 1,\ldots,t, \end{cases}$$

où $|E_0(x)| + |E_1(x)| + \sum_{i=1,\ldots,t} |\tau'_i(x)| = |E|$.

3.4.1 Opérations de réduction et points extrêmes

Considérons une solution x de $\mathcal{P}(G,\mathcal{F})$. Nous donnons dans un premier temps, quelques lemmes techniques. Ces lemmes seront utilisés par la suite pour caractériser différentes opérations de réduction de graphes.

Lemme 3.13 *Soit $f \in E$ une arête telle que $x(f) = 0$. Soit $G' = (V', E')$ le graphe obtenu à partir de G en supprimant f. Soit $\mathcal{F}' = \{F'_1,\ldots,F'_t\}$ avec $F'_i = F_i\setminus\{f\}$ pour $i = 1,\ldots,t$. Soit x' la restriction de x dans E'. Alors x est un point extrême de $\mathcal{P}(G,\mathcal{F})$ si et seulement si x' est un point extrême de $\mathcal{P}(G',\mathcal{F}')$.*

Preuve. Triviale. □

Lemme 3.14 *Soit $W \subseteq V$ un ensemble de sommets de V avec $\delta_G(W) = \{f,g\}$ tel que $x(f) = x(g) = 1$. Supposons que f appartient à un seul ensemble de $\mathcal{F}$, noté F_{i_0}, et $F_{i_0} = \{f\}$. Soit $G' = (V', E')$ le graphe obtenu à partir de G en contractant f. Soit $\mathcal{F}' = \{F'_1,\ldots,F'_t\}$ avec $F'_i = F_i\setminus\{f\}$ pour $i = 1,\ldots,t$. Si x est un point extrême de $\mathcal{P}(G,\mathcal{F})$ alors x' est un point extrême de $\mathcal{P}(G',\mathcal{F}')$ où x' est la restriction de x sur E'.*

Preuve. Soit x un point extrême de $\mathcal{P}(G,\mathcal{F})$. On peut remarquer que comme $\delta_G(W) = \{f,g\}$, f,g ne peuvent pas appartenir au même ensemble F_i. En effet, on aurait dans ce cas $x(\delta_{G_i}(W)) = 0 < 1$, ce qui contredit le fait que x est une solution de $\mathcal{P}(G,\mathcal{F})$. On veut montrer que les coupes du système $S(x)$ sont aussi des coupes dans G' et, donc serrées pour x'. Pour cela, il suffit de montrer que pour toute coupe $\delta_{G_i}(U)$ de $S(x)$, on a $\delta_{G_i}(U) = \delta_{G'_j}(U')$ pour $U' \subset V'$ et $j \in \{1,...,t\}$, où G'_j est le graphe obtenu à partir de G' en supprimant F'_j. En effet, notons que, comme $x(f) = x(g) = 1$ et $S(x)$ est non singulière, $f,g \notin \delta_{G_i}(U)$. Si f appartient à $E_G(U)$ (resp. $f \in E_G(\overline{U})$), alors il est clair

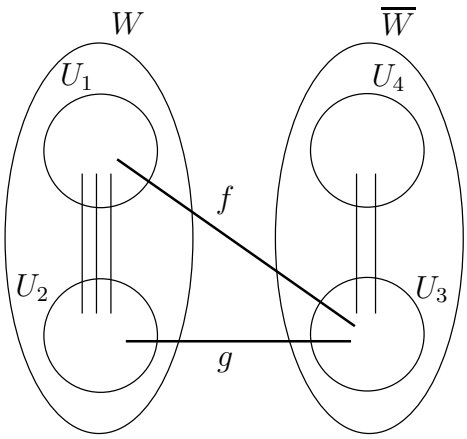


FIG. 3.20 –

que $\delta_{G_i}(U) = \delta_{G'_i}(U')$, où $U' = \overline{U}$ (resp. $U' = U$). On peut remarquer que ce cas ne se produit que si $i \neq i_0$.

Supposons que $i = i_0$, et que $f \in \delta_G(U)$. Soit $J \subset \{1, \ldots, t\}$ tel que $g \in F_j$ pour tout $j \in J$. Soit $U_1 = U \cap W$, $U_2 = W \setminus U$, $U_3 = \overline{W} \cap \overline{U}$ et $U_4 = \overline{W} \cap U$. Comme $g \notin \delta_{G_{i_0}}(U)$, on peut supposer, s.p.d.g., que $g \in \delta_G(U_2, U_3)$. De même on peut supposer que $f \in \delta_G(U_1, U_3)$ (voir figure 3.20). Soit $D = \delta_{G_{i_0}}(U)$. On peut noter que, comme $F_{i_0} = \{f\}$, $\delta_G(U) = D \cup \{f\}$. Si $\delta_G(U_3, U_4) \neq \emptyset$, alors soit $\delta_{G_{i_0}}(U_1)$, soit $\delta_{G_{i_0}}(U_4)$ est strictement contenue dans $\delta_{G_{i_0}}(U)$. Comme $\delta_{G_{i_0}}(U)$ est serrée pour x, cela implique qu'au moins l'une des deux coupes $\delta_{G_{i_0}}(U_1)$ et $\delta_{G_{i_0}}(U_4)$ est violée par x, une contradiction. Donc $\delta_G(U_3, U_4) = \emptyset$, et en conséquent $U_4 = \emptyset$. Il s'ensuit que $D = \delta_G(U_1, U_2)$, et $\delta_G(U_2) = D \cup \{g\}$. Comme $x(D) = x(\delta_{G_{i_0}}(U)) = 1$, il en résulte que toute arête de D, appartenant à un certain F_j, $j \in J$, doit avoir pour valeur zero. Sinon, on aurait $x(\delta_{G_j}(U_2)) < 1$, ce qui est impossible. Soit $U' = U_2$. On a $\delta_{G_{i_0}}(U) = \delta_{G'_j}(U')$.
Maintenant, soit $S'(x)$ le système obtenu à partir de $S(x)$ en supprimant l'équation $x(f) = 1$. x' est une solution de $S'(x)$. Comme $S(x)$ est non singulière, il est clair que $S'(x)$ est aussi non singulière. Comme les équations de $S'(x)$ correspondent à des contraintes de $\mathcal{P}(G', \mathcal{F}')$, on sait alors que x' est un point extrême de $\mathcal{P}(G', \mathcal{F}')$. □

Lemme 3.15 *Soient u, v deux sommets de V. Supposons qu'il existe deux arêtes $f, g \in [u, v]$ telles que*

1) $x(f) = x(g) = 1$, et

2) il n'existe pas $i \in \{1, \ldots, t\}$ tel que $f, g \in F_i$.

Soit $G' = (V', E')$ le graphe obtenu en contractant les arêtes de $[u, v]$, et $\mathcal{F}' = \{F'_1, \ldots, F'_t\}$ avec $F'_i = F_i \setminus [u, v]$. Alors, si x est un point extrême de $\mathcal{P}(G, \mathcal{F})$, x' est un point extrême de $\mathcal{P}(G', \mathcal{F}')$.

Preuve. Il est clair que $x' \in \mathcal{P}(G', \mathcal{F}')$. Soit $\delta_{G_i}(U)$ une coupe de $\tau_i'(x)$. On montre que $\delta_{G_i}(U) \cap [u,v] = \emptyset$. Supposons que ce n'est pas le cas, c'est-à-dire $[u,v] \backslash F_i \subset \delta_{G_i}(U)$. Comme d'après 2) f et g n'appartiennent pas au même ensemble F_i, on peut supposer s.p.d.g, que $f \notin F_i$. Comme $x(f) = 1$ et $x(g) > 0$, il s'ensuit que $g \in F_i$ et $x(e) = 0$ pour tout $e \in [u,v] \backslash (F_i \cup \{f\})$. Mais cela implique que $x(\delta_{G_i}(U)) = 1$ est redondante dans $S(x)$, ce qui contredit le fait que $S(x)$ est non singulière.
Donc, $\delta_G(U) \cap [u,v] = \emptyset$. En conséquence, $\delta_{G_i'}(U) = \delta_{G_i}(U)$ et donc $x'(\delta_{G_i'}(U)) = x(\delta_{G_i}(U)) = 1$. Soit $\widetilde{S}(x)$ le système obtenu à partir de $S(x)$ en supprimant les équations $x(f) = 1$, $x(g) = 1$ et $x(e) = 0$ avec $e \in [u,v]$. On sait alors que x' est une solution de $\widetilde{S}(x)$. On peut également remarquer que toutes les équations de $\widetilde{S}(x)$ correspondent à des contraintes de $\mathcal{P}(G', \mathcal{F}')$. Comme $S(x)$ est non singulier, $\widetilde{S}(x)$ est aussi non singulier, ce qui implique que x' est un point extrême de $\mathcal{P}(G', \mathcal{F}')$. □

Lemme 3.16 *Soit $W \subseteq V$ un ensemble de sommets avec $|W| \geq 2$ tel que $G(W)$ est $\mathcal{F}$-connexe et $x(e) = 1$ pour tout $e \in E_G(W)$. Soit $G' = (V', E')$ le graphe obtenu en contractant W. Soit $\mathcal{F}' = \{F_1', \ldots, F_t'\}$ avec $F_i' = F_i \backslash E_G(W)$. Soit x' la restriction de x sur E'. Si x est un point extrême de $\mathcal{P}(G, \mathcal{F})$, alors x' est un point extrême de $\mathcal{P}(G', \mathcal{F}')$.*

Preuve. Comme $x' \in \mathcal{P}(G', \mathcal{F}')$, comme nous l'avons fait précédemment, on va montrer que x' est l'unique solution d'un sous-système de $S(x)$ contenant des contraintes de $\mathcal{P}(G', \mathcal{F}')$.
Soit $\delta_{G_i}(U) \in \tau_i'(x)$ pour $U \subseteq V$ et $i \in \{1, \ldots, t\}$. On montre que soit $W \subseteq U$ soit $W \subseteq V \backslash U$. En effet, supposons au contraire que $U \cap W \neq \emptyset \neq (V \backslash U) \cap W$. Soient $u \in U \cap W$ et $u' \in (V \backslash U) \cap W$. Comme $G(W)$ est $\mathcal{F}$-connexe, on sait que $G[E_G(W) \backslash F_i]$ est connexe. En conséquence, u et v doivent être reliés par un chemin de $E_G(W) \backslash F_i$. Donc, comme $x(e) = 1$ pour tout $e \in E_G(W)$, $\delta_{G_i}(U) \cap E_1(x) \neq \emptyset$. Mais cela implique que $x(\delta_{G_i}(U)) = 1$ est redondante dans $S(x)$, une contradiction. Donc, toutes les coupes serrées de $S(x)$ correspondent à des contraintes de coupe de $\mathcal{P}(G', \mathcal{F}')$. Soit $\bar{S}(x)$ le système obtenu à partir de $S(x)$ en supprimant les équations $x(e) = 1$, $e \in E_G(W)$. x' est une solution de $\bar{S}(x)$. Comme $\bar{S}(x)$ est non singulier, ceci implique que x' est un point extrême de $\mathcal{P}(G', \mathcal{F}')$. □

Soient $\theta_1, \ldots, \theta_4$ les opérations introduites par les lemmes 3.13 - 3.16, respectivement, *i.e.*

θ_1 : supprimer une arête $e_0 \in E_0(x)$.

θ_2 : contracter une arête f telle que f appartient à une coupe contenant exactement 2 arêtes et f appartient à un seul ensemble de $\mathcal{F}$ et cet ensemble contient uniquement f.

θ_3 : contracter un ensemble d'arêtes parallèles $[u, v]$ tel qu'il existe deux arêtes $f, g \in [u, v] \cap E_1(x)$ n'appartenant pas au même ensemble $F_i \in \mathcal{F}$.

θ_4 : contracter un sous-ensemble de sommets $W \subseteq V$ tel que $|W| \geq 2$, $G(W)$ est $\mathcal{F}$-connexe et $E_G(W) \cap E_1(x) = E_G(W)$.

Les figures 3.21-3.23 présentent ces opérations de réductions. Sur la figure 3.21, nous avons un ensemble de sommets W définissant une coupe ayant exactement deux arêtes notées f et g. On suppose que l'arête $f \in F_{i_0}$ pour un unique $i_0 \in \{1, \ldots, t\}$ et que $F_{i_0} = \{f\}$. On peut alors appliquer l'opération θ_2 et contracter l'arête f. L'arête f disparait et les sommets u et v se rejoignent en un seul sommet.

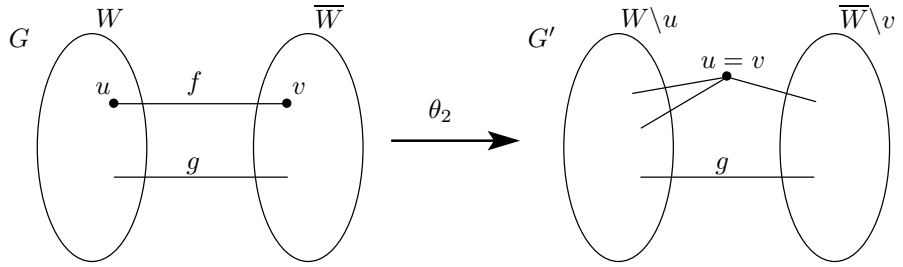


FIG. 3.21 – Opération θ_2

Sur la figure 3.22, on peut remarquer que l'on a quatre arêtes parallèles entre les sommets u et v. Parmi ces arêtes, on sait que les arêtes f et g prennent 1 pour valeur dans la solution fractionnaire considérée et on sait également que ces arêtes n'appartiennent pas au même ensemble F_i. On peut alors appliquer l'opération θ_3 et contracter toutes les arêtes entre u et v et, comme pour l'opération θ_2, les sommets u et v ne forment plus qu'un.

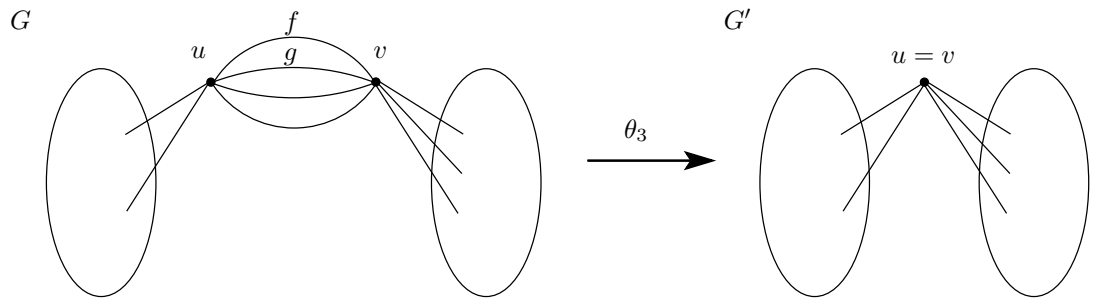


FIG. 3.22 – Opération θ_3

Le graphe $G = (V, E)$ de la figure 3.23 contient un sous-ensemble de sommets $W \subseteq V$ tel que $G(W)$ est $\mathcal{F}$-connexe et dont toutes les arêtes du sous-graphe $G(W)$ prennent 1

pour valeur dans le point fractionnaire considéré. Alors, on peut appliquer l'opération θ_4 et contracter l'ensemble de sommets W qui n'est alors plus qu'un point dans le graphe réduit G'.

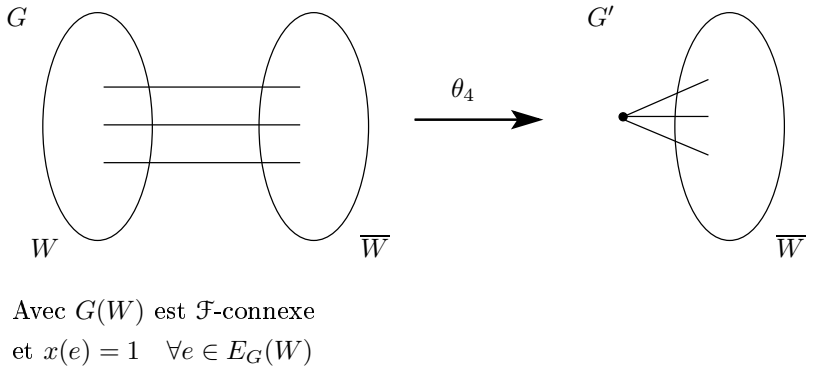


FIG. 3.23 – Opération θ_4

Les opérations $\theta_1, \ldots, \theta_4$ ont des conséquences algorithmiques intéressantes. En effet, ces opérations peuvent être implémentées en temps polynomial et dans n'importe quel ordre. De plus, les lemmes précédents montrent que si x est un point extrême de $\mathcal{P}(G, \mathcal{F})$, en appliquant de façon répétitive les opérations $\theta_1, \ldots, \theta_4$, on obtient un graphe $G' = (V', E')$, un ensemble $\mathcal{F}'$ et une solution $x' \in I\!R^{E'}$ qui est un point extrême de $\mathcal{P}(G', \mathcal{F}')$. Nous allons maintenant montrer, dans la section suivante, que ces opérations préservent les contraintes violées, parmi celles introduites dans le chapitre précédent.

3.4.2 Opérations de réduction et contraintes valides

Les opérations θ_1, θ_2, θ_3 et θ_4 peuvent être utiles dans une phase de préprocessing pour la résolution de notre problème. En effet, ces opérations n'affectent pas les contraintes de coupe, de partition, de coupe-cycle ou d'étoile-partition violées par x. En d'autres termes, toutes les contraintes de coupe, de partition, de coupe-cycle ou d'étoile-partition violées dans le graphe de départ par x, sont violées par x' dans le graphe réduit. Comme nous allons le montrer dans cette section, la séparation des inégalités (3.6), (3.11) et (3.18) par rapport à x dans G, peut être réduite à la séparation de ces inégalités dans G' par rapport à x'.

Dans ce qui suit, nous allons montrer qu'il existe une inégalité du type (3.6) (resp. (3.11)) (resp. (3.18)) violée par x dans G par rapport à $\mathcal{F}$ si et seulement si il existe une inégalité de même type violée par x' dans G' par rapport à $\mathcal{F}'$.

Soit une partition $V'_1, \ldots, V'_p$, $p \geq 2$ de V', on pose $V_1, \ldots, V_p$ la partition de V où V_i pour $i = 1, \ldots, p$, est obtenue à partir de V'_i en éclatant les sommets de V'_i qui ont été trouvés par contraction.

Lemme 3.17 *1) Si x viole une inégalité, valide pour MSIPND($G, \mathcal{F}$), de type (3.6) (resp. (3.11)) (resp. (3.18)), alors x' viole une inégalité de type (3.6) (resp. (3.11)) (resp. (3.18)) valide pour MSIPND($G', \mathcal{F}'$).*

2) 2.1) Si $a'x \geq \alpha'$ est une inégalité valide pour MSIPND($G', \mathcal{F}'$) de type (3.6) correspondant à $V'_1, \ldots, V'_p$ et un ensemble F'_i, $i = 1, \ldots, t$, alors l'inégalité $ax \geq \alpha$ où

$$a(e) = \begin{cases} a'(e) & si \quad e \in E', \\ 1 & si \quad e \in \delta_G(V_1, \ldots, V_p) \backslash (F_i \cup E'), \\ 0 & sinon, \end{cases}$$

et $\alpha = \alpha'$, est une inégalité de type (3.6) valide pourMSIPND($G, \mathcal{F}$).

2.2) Si $a'x \geq \alpha'$ est une inégalité valide pour MSIPND($G', \mathcal{F}'$) de type (3.11) associée aux ensembles d'arêtes T'_1 et T'_2. Soient $T_1 = T'_1$ et $T_2 = T'_2 \cup (\delta_G(W) \backslash (T'_1 \cup T'_2))$. Soit r défini comme dans le thérorème 3.8. Soit $ax \geq \alpha$ l'inégalité donnée par $\alpha = \alpha'$ et

$$a(e) = \begin{cases} r & si \quad e \in T_2, \\ 1 & si \quad e \in T_1, \\ 0 & sinon. \end{cases}$$

Alors $ax \geq \alpha$ est une inégalité de type (3.11) valide pour MSIPND($G, \mathcal{F}$).

2.3) Si $a'x \geq \alpha'$ est une inégalité valide pour MSIPND($G', \mathcal{F}'$) de type (3.18) associée à la partition $V'_0, \ldots, V'_p$ de V' et les ensembles $F'_{j_1}, \ldots, F'_{j_p}$, valide pour MSIPND($G', \mathcal{F}'$). Alors l'inégalité $ax \geq \alpha$ donnée par $\alpha = \alpha'$ et

$$a(e) = \begin{cases} a'(e) & si \quad e \in E', \\ 1 & si \quad e \in (\delta_G(V_0, V_i) \backslash (F_{j_i} \cup E') \quad pour \quad i \in \{1, \ldots, p\}, \\ 0 & sinon, \end{cases}$$

est une inégalité de type (3.18) valide pour MSIPND($G, \mathcal{F}$).

De plus si $a'x \geq \alpha'$ est violée par x', alors $ax \geq \alpha$ est violée par x.

Preuve. 1) Nous allons montrer l'assertion pour les contraintes de partition. La preuve est similaire pour les autres types de contraintes. Pour cela, on peut supposer que G' est obtenu par une application de θ_1, θ_2, θ_3 et θ_4. Soient $V_1, \ldots, V_p$

une partition de V et $i \in \{1, \ldots, t\}$. Supposons que $x(\delta_{G_i}(V_1, \ldots, V_p)) - p + 1 < 0$.
- Si G' est obtenu à l'aide de l'opération θ_1, il est clair que x' viole la contrainte de partition dans G' correspondant à $\delta_{G'_i}(V_1, \ldots, V_p)$.
- Supposons que G' est obtenu à l'aide de l'opération θ_2 associée à l'arête f. On montre que $f \notin \delta_{G_i}(V_1, \ldots, V_p)$. En effet, supposons par exemple que $f = uv \in \delta_G(V_j, V_{j'}) \backslash F_i$, où $j, j' \in \{1, \ldots, p\}$. Comme $x(f) = 1$ et $x(e) \geq 0$ pour tout $e \in \delta_G(V_j, V_{j'})$, on sait que $x(\delta_G(V_j, V_{j'})) \geq 0$. Soit $\widetilde{V}_1, \ldots, \widetilde{V}_{p-1}$ la partition de V obtenue à partir de $V_1, \ldots, V_p$ en fusionnant V_j et $V_{j'}$, en un même ensemble, noté $\widetilde{V}_1$. Soit $\widetilde{V}'_1 = (\widetilde{V}_1 \backslash \{u, v\}) \cup \{w\}$, où w est le sommet provenant de la contraction de l'arête f. On peut noter que $\widetilde{V}'_1, \widetilde{V}_2, \ldots, \widetilde{V}_{p-1}$ est une partition de V'. De plus, on a

$$\begin{aligned} x'(\delta_{G'_i}(\widetilde{V}'_1, \widetilde{V}_2, \ldots, \widetilde{V}_{p-1})) &= x(\delta_{G_i}(\widetilde{V}_1, \ldots, \widetilde{V}_{p-1})) \\ &= x(\delta_{G_i}(V_1, \ldots, V_p)) - x(\delta_G(V_j, V_{j'}) \backslash F_i) \\ &= x(\delta_{G_i}(V_1, \ldots, V_p)) - x(f) - x(\delta_G(V_j, V_{j'}) \backslash (F_i \cup \{f\})) \\ &< p - 2. \end{aligned}$$

Donc x' viole la contrainte de partition induite par $\widetilde{V}'_1, \widetilde{V}_2, \ldots, \widetilde{V}_{p-1}$ associée à F'_i.
- Si G' est obtenu à partir de G par θ_3 ou θ_4, on peut construire de la même façon une partition de G' telle que la contrainte associée à cette partition et à F'_i soit violée par x'.

2) On peut remarquer que toute arête e de $E \backslash E'$ avec $x(e) = 1$ a ses deux extrémités dans le même élément de la partition.

□

Le lemme 3.17 montre que chercher une inégalité de type (3.6), (3.11) ou (3.18), violée par x, revient à chercher une inégalité de même type violée par x' dans G'. On peut noter que cette procédure peut être appliquée quelque soit la solution de $\mathcal{P}(G, \mathcal{F})$ et, donc, permet de séparer des solutions fractionnaires qui ne sont pas nécessairement des points extrêmes de $\mathcal{P}(G, \mathcal{F})$. En conséquence, pour plus d'efficacité, nos procédures de séparation seront appliquées au graphe réduit G'.

Au vu de ces résultats et afin d'améliorer la performance de nos procédures de séparation, nous avons implémenté les opérations θ_1, θ_2, θ_3 et θ_4. Si une inégalité de coupe, de partition, de coupe-cycle ou d'étoile-partition est violée par x' dans G', alors cette inégalité peut être "liftée" en une inégalité de coupe, de partition, de coupe-cycle ou d'étoile-partition dans G qui coupe x.

3.4.3 Exemples de réductions de graphes

Dans cette section, nous allons présenter différents exemples de réductions sur des graphes de toutes tailles et pour différents points fractionnaires. Dans la suite, et pour toutes les figures représentant des points fractionnaires, on fera apparaître les arêtes à 1 avec des traits pleins, les arêtes à 0.5 avec des pointillés longs, et les arêtes fractionnaires qui ne sont pas à 0.5 avec des pointillés sérrés. Les exemples suivants ont été obtenus en utilisant un programme qui sera décrit au chapitre 4.

Opérations successives

Nous montrons tout d'abord les différentes opérations de réduction sur un point fractionnaire d'une instance à 28 sommets et 46 arêtes. La figure 3.24 présente le point fractionnaire considéré. Seule l'opération θ_1 qui consiste à ne pas faire apparaître les arêtes à 0 a été appliquée.

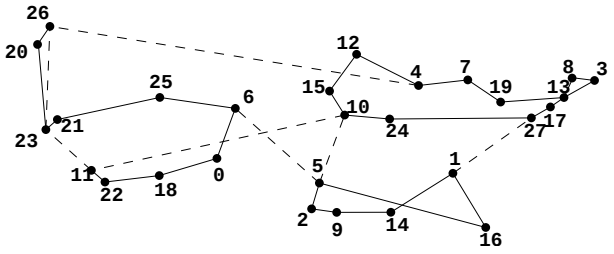


FIG. 3.24 – Point fractionnaire avant réduction

Pour obtenir le graphe réduit de la figure 3.25, nous avons appliqué 19 opérations θ_2. Ensuite, 2 opérations θ_3 ont été appliquées pour obtenir le graphe de la figure 3.26(a). On peut ainsi contracter les arêtes parallèles entre les sommets 9 et 16 et entre les sommets 21 et 26.

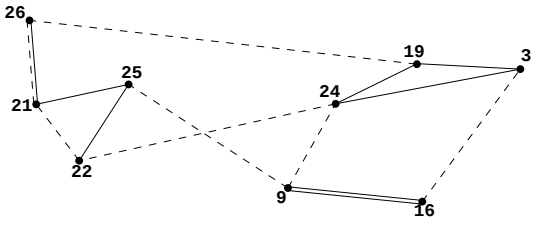


FIG. 3.25 – Point fractionnaire avec réduction à l'aide de l'opération θ_2

Une opération θ_4 permet d'obtenir le graphe réduit de la figure 3.26(b). En effet, le sous-graphe induit par les sommets 3, 19 et 24 est constitué de 3 arêtes à 1. Ces arêtes

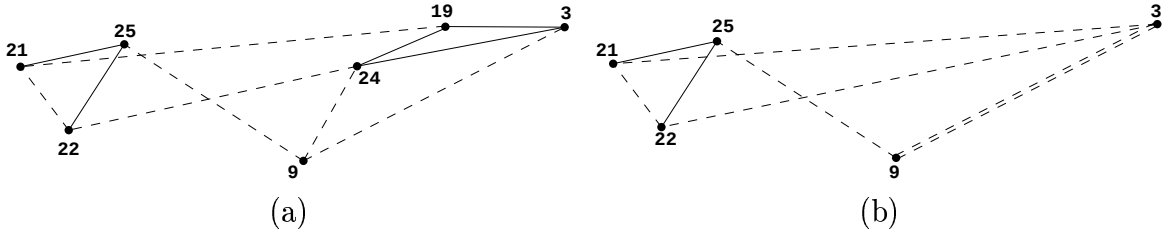


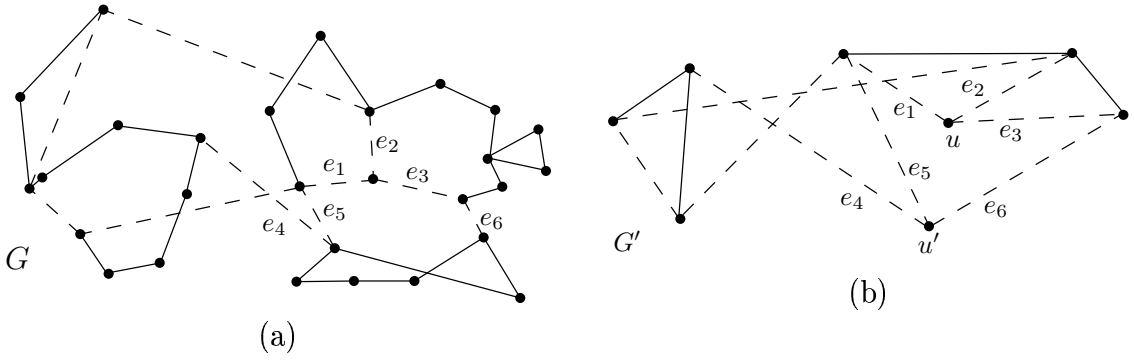
FIG. 3.26 – Point fractionnaire avec réduction à l'aide (a) de l'opération θ_3 puis (b) de l'opération θ_4

n'appartenant pas au même $F_i \in \mathcal{F}$, ce sous-graphe est $\mathcal{F}$-connexe, et peut donc être contracté en un sommet unique nommé 3.

Pour cette instance et le point fractionnaire considéré, les opérations de réduction permettent de passer d'un graphe à 28 sommets à un graphe à 5 sommets.

Détection de contraintes

L'exemple que nous donnons ici permet d'illustrer l'ajout de contraintes de coupe-cycle et d'étoile-partition. En effet, nous présentons des contraintes sur les graphes réduits qui coupent le point fractionnaire associé, et nous montrons que ces contraintes, une fois liftées, permettent de couper le point fractionnaire du graphe de départ.



La figure 3.27(a) montre un point fractionnaire obtenu en résolvant un problème MSIPND sur un graphe à 28 sommets et $|\mathcal{F}| = 46$. En appliquant les opérations $\theta_1, \ldots, \theta_4$, on obtient le graphe réduit G' de la figure 3.27(b) ainsi que la solution fractionnaire correspondante. Les sommets u et u' de G' induisent les contraintes de coupe-cycle violées

$$x(e_1) + x(e_2) + x(e_3) \geq 2,$$
$$x(e_4) + x(e_5) + x(e_6) \geq 2,$$

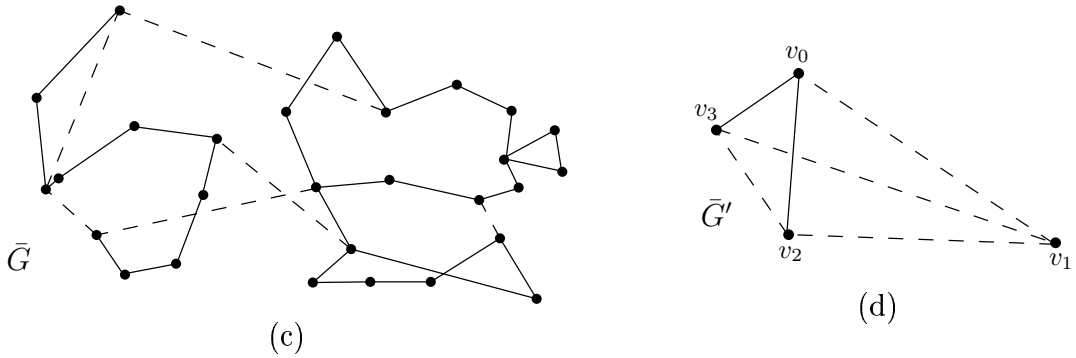


FIG. 3.27 –

où les arêtes e_1, e_2, e_3 (resp. e_4, e_5, e_6) n'appartiennent pas au même F'_i de $\mathcal{F}'$. Remarquons que la contrainte de coupe-cycle induite par u' n'apparait pas comme une coupe sur un sommet dans G. En liftant ces inégalités et en les ajoutant à la relaxation linéaire, on obtient la solution fractionnaire de la figure 3.27(c). On note par $\bar{G}$ le graphe support de cette solution. En utilisant les opérations $\theta_1, \ldots, \theta_4$, on obtient le graphe $\bar{G}'$ de la figure 3.27(d). Soit F l'ensemble des arêtes incidentes à v_0 dans $\bar{G}'$. Chaque arête de F appartient à un ensemble $F'_i \in \mathcal{F}'$. En considérant les sommets de $\bar{G}'$ comme les éléments $V_0, \ldots, V_3$ de la partition de $\bar{G}'$ où $V_i = \{v_i\}$, pour $i = 0, \ldots, 3$, on obtient la contrainte d'étoile-partition suivante

$$x(\delta_{\bar{G}'}(V_0, \ldots, V_3) \backslash F) \geq 2.$$

Cette inégalité est violée par la solution de $\bar{G}'$. En liftant cette inégalité, on obtient une contrainte d'étoile-partition qui est violée par le point fractionnaire de $\bar{G}$.

Autres exemples

Nous donnons maintenant d'autres exemples de réduction de graphes contenant entre 18 et 50 sommets. Pour toutes les figures, nous donnerons à gauche le graphe de départ et à droite le graphe réduit associé.

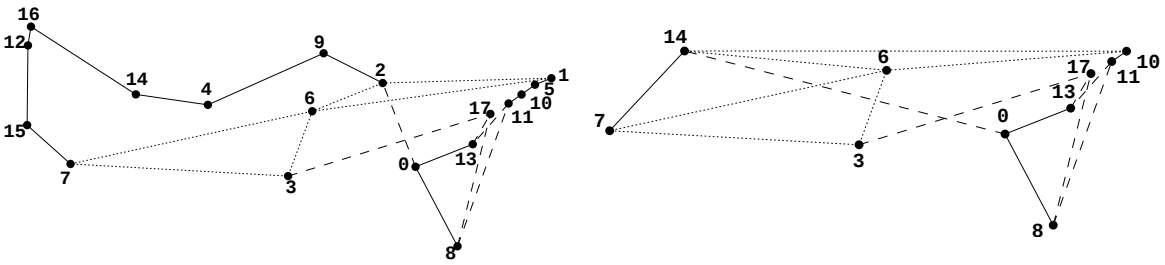


FIG. 3.28 – Instance à 18 sommets et 31 arêtes

Pour le graphe de la figure 3.28, la réduction a permis de transformer le graphe de 18 sommets en un graphe plus petit ne comportant que 10 sommets. Les sommets 1 et 5 ont été compressés au sommet 10 et les sommets 2, 4, 9, 12, 15 et 16 ont été compressés au sommet 14. On peut repérer très facilement deux contraintes de coupe-cycle associées aux sommets 3 et 17. De même, on peut vérifier que les deux cycles impairs utilisant le sommet 6 permettent de définir deux contraintes d'étoile-partition.

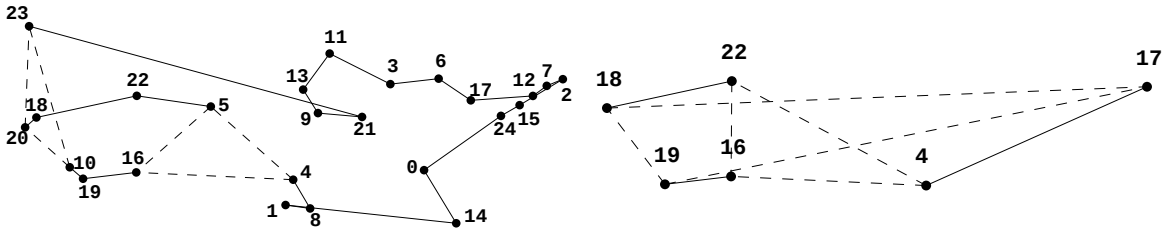


FIG. 3.29 – Instance à 25 sommets et 40 arêtes

La réduction de la figure 3.29 a permis de passer d'un graphe à 25 sommets à un graphe à 6 sommets. On peut facilement trouver deux contraintes d'étoile-partition en considérant les cycles impairs composés des sommets 17, 18 et 19 et des sommets 4, 16 et 22.

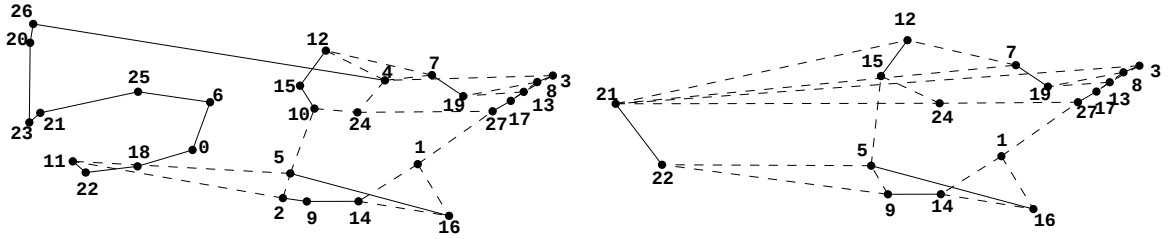


FIG. 3.30 – Instance à 28 sommets et 46 arêtes

Pour le graphe de la figure 3.30, le graphe à 28 sommets est réduit à un graphe à 17 sommets. On peut par exemple, faire remarquer les deux contraintes de coupe-cycle associées aux sommets 1 et 24, mais aussi la contrainte d'étoile-partition liée au cycle contenant les sommets 5, 9 et 22.

L'instance de la figure 3.31 est réduite à un graphe de seulement 7 sommets dans lequel on identifie très facilement la contrainte d'étoile-partition liée aux sommets 8, 16 et 28.

Sur la figure 3.32, le graphe à 35 sommets est réduit à un graphe de 14 sommets. Cette instance contient beaucoup d'arêtes fractionnaires qui ne sont pas à 0.5. On remarque très facilement la contrainte de coupe-cycle associée au sommet 6, ainsi que deux contraintes d'étoile-partition déterminées à l'aide des cycles correspondant aux sommets 20, 31 et 34 et aux sommets 24, 30 et 33.

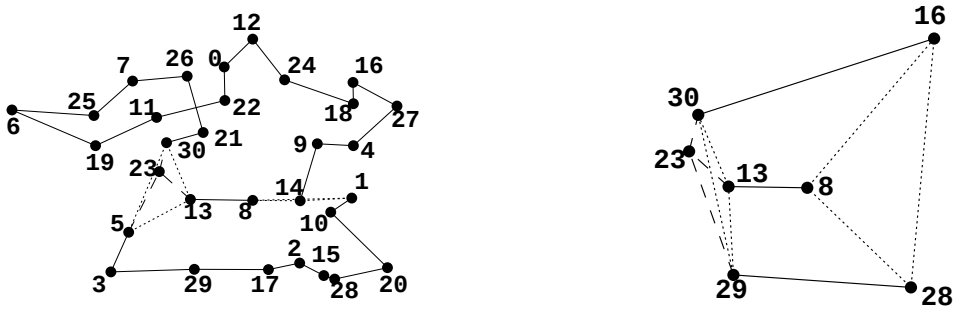


FIG. 3.31 – Instance à 31 sommets et 46 arêtes

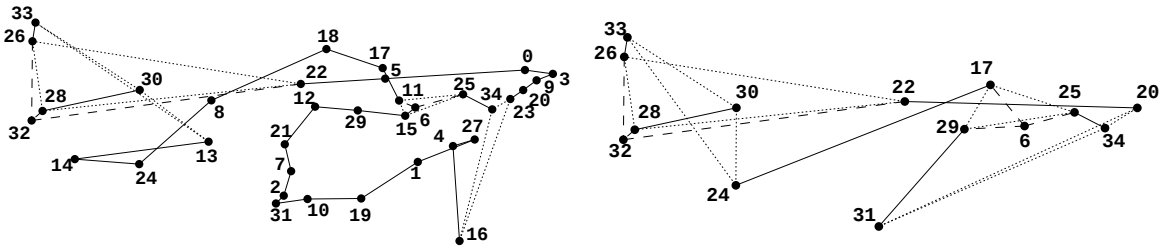


FIG. 3.32 – Instance à 35 sommets et 63 arêtes

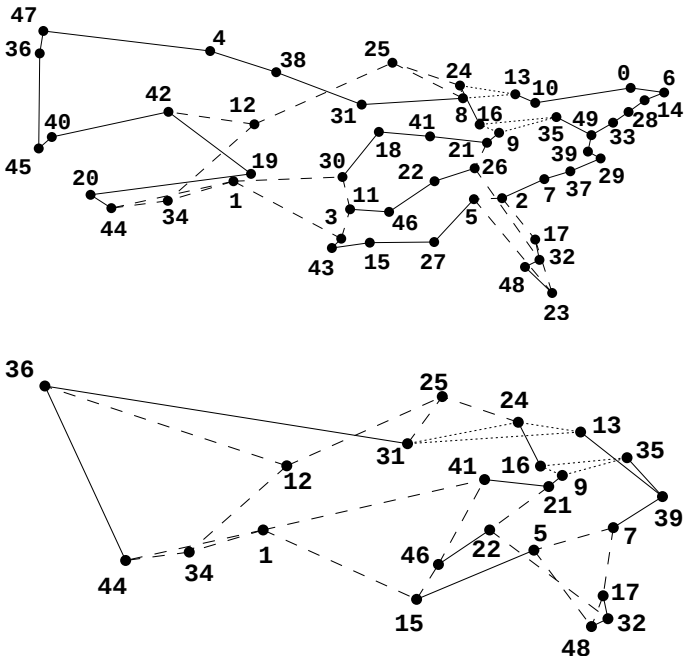


FIG. 3.33 – Instance à 50 sommets et 95 arêtes

Les opérations de réduction semblent être également utiles pour des graphes de grande taille comme celui de la figure 3.33. Ici il s'agit d'un graphe de 50 sommets réduit en un graphe de 23 sommets. Sur ce dernier graphe, on identifie entre autres, deux contraintes

de coupe-cycle associées aux sommets 12 et 25, et trois contraintes d'étoile-partition associées aux cycles $(1, 34, 44)$, $(9, 16, 35)$ et $(13, 24, 31)$.

Nous pouvons noter que les contraintes d'étoile-partition que nous avons identifiées sur ces exemples sont toujours liées à un cycle impair ayant trois arêtes. De même, les contraintes de coupe-cycle sont toujours associées à un sommet dans le graphe réduit et non à un ensemble de sommets. Ce sont, en effet, les cas les plus fréquents dans la pratique.

3.5 Conclusion

Dans ce chapitre, nous avons présenté le problème de sécurisation multicouche du réseau IP. Nous avons donné une formulation sous la forme d'un programme linéaire en nombres entiers de ce problème et présenté de nouvelles contraintes valides. Nous avons également donné des conditions nécessaires et suffisantes pour que celles-ci définissent des facettes du polytope associé au problème. Nous avons également introduit des opérations de réduction de graphes qui vont être très utiles lors de la séparation des différentes classes de contraintes. Les résultats présentés dans ce chapitre nous ont permis de développer un algorithme de coupes et branchements que nous présentons dans le chapitre suivant.

Chapitre 4

Algorithme de coupes et branchements pour le problème MSIPND

Dans ce chapitre, nous présentons un algorithme de coupes et branchements pour résoudre le problème de sécurisation multicouche du réseau IP. Cet algorithme est basé sur les résultats polyédraux du chapitre précédent. Ce travail a pour but de montrer l'efficacité des contraintes définies lors de l'étude polyédrale et d'obtenir des solutions exactes pour des problèmes réels.

4.1 Algorithme de coupes et branchements

4.1.1 Aperçu général de l'algorithme

On suppose que l'on a deux graphes $G^1 = (V^1, E^1)$ et $G^2 = (V^2, E^2)$, représentant respectivement la couche IP et la coupe optique du réseau. On suppose également donné une famille $\mathcal{F} = \{F_e \subseteq E^1, e \in E^2\}$ d'ensembles d'arêtes de E^1 associés à chaque arête de G^2, où F_e est l'ensemble d'arêtes de E^1 affectées par la panne de l'arête e de E^2. Pour respecter les notations du chapitre précédent, on note les éléments de $\mathcal{F}$ par $F_1, \ldots, F_t$, où $t = |E^2|$ et $G^1 = (V^1, E^1)$ par $G = (V, E)$. Un vecteur poids associé aux arêtes $c \in \mathbb{R}^{E^1}$ est également connu.

Pour commencer l'optimisation, nous considérons le programme linéaire suivant, donné par les contraintes de coupe associées aux sommets du graphe G, ainsi que les contraintes triviales. Ce programme linéaire, consistant en $|E|$ variables, $|V| \times |\mathcal{F}|$

contraintes de coupe associées aux sommets du graphe et $2|E|$ contraintes triviales, est le suivant :

$$\mathrm{PL}_{debut} \begin{cases} \text{Min} \sum_{f \in E} c(f)x(f) & \\ x(\delta_{G_i}(v)) \geq 1 & \text{pour tout } v \in V, \text{ pour tout } i = 1, \ldots, t, \\ 0 \leq x(f) \leq 1 & \text{pour tout } f \in E. \end{cases}$$

La solution optimale $y \in I\!R^E$ de la relaxation du problème MSIPND est réalisable pour le problème si y est un vecteur entier qui satisfait toutes les coupes. En général, la solution ainsi obtenue n'est pas réalisable pour le problème MSIPND. Dans ce cas, à chaque itération de l'algorithme, nous essayons de trouver des inégalités valides pour le problème MSIPND et violées par la solution optimale du programme linéaire courant. Les inégalités trouvées sont ajoutées au programme linéaire courant, puis nous résolvons le nouveau programme linéaire ainsi obtenu. Ce processus est repris jusqu'à ce qu'aucune inégalité violée ne soit détectée. L'algorithme procède alors à un branchement qui est effectué à l'aide du logiciel ABACUS. L'algorithme 4.1 illustre les principales phases de notre algorithme de coupes et branchements.

Algorithme 4.1 Algorithme de coupes et branchements

Données : un graphe $G = (V, E)$, une famille $\mathcal{F}$ d'ensembles d'arêtes et un vecteur poids $c \in I\!R^E$.
Sortie : solution optimale de Min$\{cx \mid x \in$MSIPND$(G, \mathcal{F})\}$.

1 : PL $\leftarrow$ PL_{debut}
2 : Résoudre le programme linéaire PL.
Soit y la solution optimale de PL.
3 : **Si** y est réalisable pour MSIPND(G,$\mathcal{F}$) **alors**
y est une solution optimale. STOP
4 : **Si** des contraintes (coupe, coupe-cycle, étoile-partition, partition) violées par y sont trouvées **alors**
Les ajouter à LP.
Aller en 2.
5 : **Sinon**
Brancher sur une variable fractionnaire.
6 : Prendre la meilleure solution de tous les sous-problèmes.

La séparation des classes d'inégalités présentées précédemment s'effectue dans l'ordre suivant :

1. constraintes de coupe
2. contraintes de coupe-cycle
3. contraintes d'étoile-partition
4. contraintes de partition

On peut remarquer que toutes les inégalités sont globales (*i.e.* valides dans tout l'arbre de branchement) et plusieures contraintes peuvent être ajoutées à chaque itération. De plus, on passe à la classe d'inégalités suivante seulement si l'on ne trouve plus d'inégalités violées. Notre stratégie consiste à essayer de détecter des contraintes violées à chaque nœud de l'arbre de branchement afin d'obtenir la meilleure borne inférieure possible et donc de limiter le nombre de nœuds générés dans l'arbre. Les inégalités sont ajoutées par ensembles d'au plus 200 inégalités à la fois.

Pour conserver les contraintes générées, nous avons créé des structures de données appelées pools dont la taille augmente dynamiquement. Toutes les contraintes générées sont mémorisées dans un pool et sont dynamiques (*i.e.* elles sont enlevées du programme linéaire courant quand elles ne sont pas actives). Nous séparons tout d'abord les inégalités du pool. Si toutes les contraintes du pool sont satisfaites par la solution du programme linéaire courant, on sépare les classes d'inégalités dans l'ordre donné ci-dessus.

Dans ce qui suit, nous décrivons les procédures de séparation utilisées dans notre algorithme. Nous utilisons des algorithmes exacts ainsi que des heuristiques selon la classe d'inégalités. Toutes nos séparations sont appliquées sur le graphe G' avec des poids $(\overline{y}'(e),\ e \in E')$ associés aux arêtes où $\overline{y}'$ est la restriction sur E' de la solution courante du programme linéaire $\overline{y}$. Mais auparavant, nous présentons notre test de faisabilité.

4.1.2 Test de faisabilité

Les contraintes de base du problème (c'est-à-dire les contraintes de coupe) sont en nombre exponentiel. Elles ne peuvent donc pas être toutes énumérées et ne sont pas toutes présentes dans le programme linéaire de départ. Par conséquent, une solution optimale du programme linéaire de départ, même si elle est entière, n'est pas obligatoirement une solution réalisable du problème MSIPND, il faut également que cette

solution vérifie toutes les contraintes de coupe. Nous avons donc ajouté un test de faisabilité, qui vérifie si une solution est réalisable ou non. Ce test se ramène à vérifier que toutes les contraintes de coupe sont satisfaites. Il est basé, comme pour la séparation des contraintes de coupe (voir section suivante) sur une implémentation de Gusfield [46, 47] de l'algorithme de Gomory-Hu [39] utilisant l'algorithme de Goldberg et Tarjan [38] pour la recherche d'un flot maximum dans un graphe.

4.1.3 Séparation des contraintes de coupe

Le problème qui consiste à déterminer une ou plusieurs contraintes de coupe violées par $\bar{y}'$ peut être résolu en appliquant l'algorithme de Gomory-Hu [39] au graphe G' en considérant le coût $\bar{y}'$ associé aux arêtes de G'. Cet algorithme produit l'arbre de Gomory-Hu ayant la propriété que pour toute paire de sommets $s, t \in V'$, la coupe minimum séparant s et t dans G' n'est rien d'autre que la coupe minimum séparant s et t dans l'arbre. Gusfield [46, 47] a donné une version de cet algorithme consistant à résoudre $|V'| - 1$ problèmes de flot maximum dans le graphe G' en respectant un certain ordre.

D'après le théorème du flot maximum - coupe minimum, dû à Ford et Fulkerson [33], nous pouvons résoudre le problème de recherche d'une coupe minimum en temps polynomial. Dans tous nos problèmes de flot maximum (et donc de coupe minimum), nous utilisons l'algorithme de Goldberg et Tarjan [38] qui est un des plus performants algorithmes pour ce problème. Cet algorithme a une complexité en $\mathcal{O}(m'n'log\frac{n'^2}{m'})$ avec m' et n' le nombre d'arêtes et de sommets de G' respectivement.

L'algorithme nécessite, pour chaque $F_i \in \mathcal{F}$, au plus $|V'| = n'$ calculs d'une coupe minimum. Donc la séparation des contraintes de coupe pour $F_i \in \mathcal{F}$ a une complexité en $\mathcal{O}(m'n'^2log\frac{n'^2}{m'})$. Notre algorithme, permettant de séparer ces contraintes pour tout $F_i \in \mathcal{F}$, a donc une complexité en $\mathcal{O}(m'n'^2tlog\frac{n'^2}{m'})$ où $t = |\mathcal{F}|$.

4.1.4 Séparation des contraintes de partition

Pour la séparation des contraintes de partition, nous utilisons une séparation exacte, mais nous utilisons également une séparation heuristique afin d'améliorer les temps de calcul. Lors de la séparation, nous utilisons tout d'abord notre heuristique. Si cette heuristique ne trouve plus de contrainte violée, alors nous utilisons la séparation exacte.

4.1.4.1 Séparation heuristique

Cette heuristique gloutonne consiste en deux phases. Dans un premier temps, on contracte toutes les arêtes de E' ayant une valeur 1. S'il existe une contrainte de partition violée avec des arêtes de valeur 1 entre les éléments, alors la partition obtenue en contractant ces arêtes est aussi violée. Dans un second temps, on teste si la partition triviale donnée par le graphe résultant correspond à une contrainte violée. Si ce n'est pas le cas, on commence à contracter les arêtes ayant des grandes valeurs, jusqu'à ce que, soit le graphe ait p sommets et un poids strictement inférieur à $p-1$, soit le graphe consiste en deux sommets. Dans le deuxième cas, si une contrainte de partition est violée, c'est une contrainte de coupe violée.

4.1.4.2 Séparation exacte

Les contraintes de partition (3.6) ont été montrées valides pour différents problèmes d'optimisation combinatoire. Un premier algorithme de séparation pour ces inégalités à été développé par Cunningham [20] et nécessite la recherche de $|E'|$ coupes minimums. Barahona [8] a réduit ce temps de calcul à $|V'|$ recherches de coupes minimum. Ces deux algorithmes, celui de Cunningham et celui de Barahona donnent la coupe la plus violée. Pour plus d'efficacité, nous avons implémenté l'algorithme de Barahona, qui a une complexité en $O(m'n'^3 t log\frac{n'^2}{m'})$. Baïou, Barahona et Mahjoub [7] ont également donné un algorithme polynomial pour résoudre ce problème de séparation basé sur la minimisation d'une fonction sous-modulaire symétrique.

4.1.5 Séparation des contraintes de coupe-cycle

Nous avons étudié la séparation des contraintes de coupe-cycle (3.11) uniquement dans le cas où $q = 1$. En effet, nous avons remarqué que les contraintes de coupe-cycle violées trouvées lors de notre résolution sont le plus souvent de ce type. D'après le théorème 3.9 1) une contrainte de coupe-cycle avec $q = 1$, induite par une coupe $\delta_G(W)$ et qui définit une facette pour MSIPND$(G, \mathcal{F})$, est de la forme

$$x(T_1) + x(T_2) \geq 2, \tag{4.1}$$

où $T_1 \cap T_2 = \emptyset$, $T_1 \cup T_2 \subset \delta_G(W)$ et T_1, T_2 satisfont le théorème 3.8. Pour séparer ces contraintes, nous avons développé une heuristique basée sur l'abre de Gomory-Hu qui fonctionne comme suit.

Dans un premier temps, on contracte toutes les arêtes de valeur 1. Ensuite on calcule l'arbre de Gomory-Hu sur le graphe résultant, que l'on note $\widetilde{G}'$. Chaque coupe donnée par l'arbre de Gomory-Hu, ayant une valeur strictement inférieure à 2 donne une inégalité de type (4.1) violée par $\overline{y}'$. Si $\delta_{\widetilde{G}'}(W) = \{f_1, \ldots, f_m\}$, alors on détermine les ensembles T_1 et T_2 de telle sorte que T_1 soit maximal, en utilisant l'algorithme 4.2.

Algorithme 4.2

$T_1 \leftarrow \emptyset$; $T_2 \leftarrow \emptyset$; $\bar{\mathcal{F}} \leftarrow \emptyset$;
pour $i = 1$ à m **faire**
  si $f_i \in F_{j_0}$ pour $j_0 \in \{1, \ldots, t\}$ et $f_i \notin F_j$ pour tout $F_j \in \bar{\mathcal{F}}$ **alors**
    $T_1 \leftarrow T_1 \cup \{f_i\}$;
    $\bar{\mathcal{F}} \leftarrow \bar{\mathcal{F}} \cup \{F_{j_0}\}$;
  sinon
    $T_2 \leftarrow T_2 \cup \{f_i\}$;
pour tout $f_i \in T_2$ **faire**
  si $f_i \in F_j$ pour tout $F_j \in \bar{\mathcal{F}}$ **alors**
    $T_2 \leftarrow T_2 \backslash \{f_i\}$;

Comme l'algorithme de Gomory-Hu tourne avec une compléxité trop importante pour l'utiliser trop fréquemment dans une séparation ($O(|V'|^4)$), nous considérons tout d'abord les coupes sur les sommets $\delta_{\widetilde{G}'}(v)$, $v \in V'$. Le calcul de l'arbre de Gomory-Hu sur le graphe $\widetilde{G}'$ est considéré uniquement lorsqu'aucune coupe de ce type, ayant une valeur strictement inférieure à 2, n'a été trouvée.

4.1.6 Séparation des contraintes d'étoile-partition

Notre algorithme de séparation des contraintes d'étoile-partition (3.18) consiste tout d'abord à déterminer un cycle impair ayant des valeurs fractionnaires dans le graphe support, satisfaisant un certain nombre de conditions. En effet, comme on peut le remarquer, le graphe induit par $\delta(V_1, \ldots, V_p) \backslash F$ où $V_0, \ldots, V_p$ induit une contrainte d'étoile-partition associée à F, peut représenter un cycle impair. C'est par exemple le cas pour l'exemple d'étoile-partition de la figure 3.27 d). Cette observation nous a permis de donner la séparation heuristique suivante. On cherche tout d'abord les cycles impairs dans G' qui sont formés d'arêtes ayant des valeurs fractionnaires dans $\overline{y}'$. Puis, pour chaque cycle impair détecté $(v_1, \ldots, v_p)$, on essaye de trouver des ensembles F_{j_i}, $j_i \in \{1, \ldots, t\}$, $i = 1, \ldots, p$ parmi les arêtes de $\delta_G(v_i, V' \backslash \{v_1, \ldots, v_p\})$ tels que la contrainte d'étoile-partition induite par $V' \backslash \{v_1, \ldots, v_p\}, \{v_1\}, \ldots, \{v_p\}$, et F_{j_i}, $i = 1, \ldots, p$ soit violée par $\overline{y}'$. Cette heuristique peut être implémentée avec une complexité $O(|V'|^2)$.

4.2 Résultats expérimentaux

4.2.1 Contexte informatique

Avant de présenter nos différents résultats expérimentaux, nous donnons un bref descriptif des logiciels et du matériel informatique que nous avons utilisés.

L'algorithme de coupes et branchements décrit précédemment a été implementé en utilisant le logiciel ABACUS (A Branch-And-CUt System), version 2.4 alpha [1, 30, 81]. Pour la résolution des programmes linéaires, ABACUS fait appel au logiciel CPLEX, version 8.1 [2], qui est une implémentation très rapide de l'algorithme du simplexe.

Ce travail expérimental, implémenté en langage C++, a été réalisé sur un Pentium IV cadencé à 2,4 Ghz avec 1024 Mo de mémoire vive. Nous avons fixé un temps maximum d'exécution à 5 heures. Dans la section suivante, nous allons présenter brièvement les instances sur lesquelles nous avons effectué nos tests.

4.2.2 Description des instances traitées

Les résultats qui sont présentés ici ont été obtenus à partir d'instances provenant d'applications réelles et d'instances aléatoires. Pour toutes les instances, le graphe G^1, représentant le réseau IP, est considéré complet.

Les instances aléatoires proviennent de la TSP Library [76]. Le graphe G^1 est obtenu en considérant un sous-ensemble de sommets d'instances de le TSP Library. Le graphe G^1 est complet. Le coût d'installation d'une arête est égal à la distance euclidienne entre ses deux extrémités. Les ensembles d'arêtes F_e sont générés aléatoirement. Ces instances sont générées avec 10 à 45 sommets et $|\mathcal{F}| = 10, 15, 20, 30$. Nous avons testé cinq instances pour chaque taille et chaque $|\mathcal{F}|$. Nous considérons les moyennes des résultats obtenus pour ces 5 instances.

Les instances réelles sont extraites de réseaux opérationnels fournis par France Télécom. Ces instances ont entre 18 et 120 sommets et les ensembles $\mathcal{F}$ associés contiennent de 31 à 243 ensembles d'arêtes. En fait, France Télécom nous a fourni la topologie du réseau optique et le routage entre chaque paire de nœuds du réseau. A chaque arête f du réseau IP, on associe le chemin de routage dans le réseau optique reliant les brasseurs correspondant aux routeurs IP extrémités de f. En utilisant ces chemins, nous avons généré $\mathcal{F} = \{F_e \subseteq E^1,\ e \in E^2\}$ où F_e est l'ensemble des arêtes f de E^1 telles que

e appartient au chemin associé à f. Pour ces instances, nous avons utilisé différentes fonctions coût.

Nos résultats expérimentaux sont reportés dans les tables des sections suivantes. Les différentes colonnes de ces tables représentent :

$\|V^1\|$	:	le nombre de sommets de G^1,
$\|\mathcal{F}\|$	:	le nombre d'ensembles F_e,
NC	:	le nombre de contraintes de coupe générées,
NCC	:	le nombre de contraintes de coupe-cycle générées,
NSP	:	le nombre de contraintes d'étoile-partition générées,
NP	:	le nombre de contraintes de partition générées,
NT	:	le nombre de nœuds de l'arbre de branchement générés,
o/p	:	le nombre de problèmes résolus à l'optimum sur le nombre d'instances testées (uniquement pour les instances aléatoires),
Copt	:	la valeur de la solution optimale (uniquement pour les instances réelles),
Gap	:	l'erreur relative entre la meilleur borne supérieure (la valeur optimale si le problème à été résolu à l'optimum) et la borne inférieure obtenue par la phase de coupes (avant branchement),
TT	:	le temps CPU total en h :m :s.

4.2.3 Instances aléatoires

Notre première série d'expérimentations concerne les instances aléatoires. La table 4.1 rapporte la moyenne des résultats obtenus pour ces instances. On remarque que pour moins de 25 sommets, tous les problèmes ont été résolus à l'optimum. De plus, pour $|\mathcal{F}| = 10$ toutes les instances ont également été résolues dans la limite de temps de calcul fixée. Pour les instances de 30 sommets (resp. 35 sommets) avec $|\mathcal{F}| \leq 20$ (resp. $|\mathcal{F}| \leq 15$) nous avons pu également obtenir la solution optimale. Par contre, pour $|\mathcal{F}| = 30$, (resp. $|\mathcal{F}| = 20$), une des cinq instances testées n'a pas été résolue.

Les instances ayant plus de 35 sommets et $|\mathcal{F}| \geq 20$ semblent plus difficiles à résoudre. En fait, on peut remarquer que pour 40 et 45 sommets et $|\mathcal{F}| \geq 20$, au plus deux instances ont été résolues dans le temps imparti. Pour $|\mathcal{F}| = 30$, aucune instance n'a été résolue à l'optimalité. On peut également remarquer qu'en général, le problème devient difficile quand $|\mathcal{F}|$ augmente, ce qui explique l'augmentation du temps CPU avec celle de $|\mathcal{F}|$. Ceci est naturel car pour une famille $\mathcal{F}$ donnée, on doit résoudre $|\mathcal{F}|$ problèmes de sous-graphe connexe.

Nous pouvons également remarquer que, pour la plupart des instances, un nombre

$\|V^1\|$	$\|\mathcal{F}\|$	NC	NCC	NSP	NP	NT	o/p	Gap	TT
10	10	59.6	1.0	0.0	28.0	9.8	5/5	1.80	0 :00 :00.88
10	15	40.8	1.2	0.2	38.8	5.4	5/5	0.92	0 :00 :01.03
10	20	50.6	1.6	0.2	46.8	2.6	5/5	0.32	0 :00 :01.35
10	30	66.0	2.4	0.0	62.6	7.0	5/5	1.39	0 :00 :02.61
15	10	185.4	2.4	0.0	71.8	15.0	5/5	1.11	0 :00 :05.06
15	15	176.6	3.0	0.2	86.2	11.8	5/5	2.13	0 :00 :08.63
15	20	158.0	3.6	0.2	90.0	10.6	5/5	1.25	0 :00 :10.33
15	30	124.2	9.2	0.0	198.0	43.8	5/5	3.43	0 :00 :51.39
20	10	194.8	2.0	0.2	216.6	45.8	5/5	1.57	0 :00 :41.08
20	15	271.8	5.8	0.0	300.8	99.0	5/5	2.82	0 :02 :03.77
20	20	230.2	10.0	0.2	885.6	294.2	5/5	3.69	0 :08 :58.00
20	30	264.2	16.2	0.2	1897.8	575.8	5/5	5.16	0 :26 :17.80
25	10	315.6	3.2	0.2	328.6	71.8	5/5	1.65	0 :02 :29.52
25	15	281.4	10.0	0.6	924.0	236.6	5/5	2.56	0 :12 :02.02
25	20	295.0	10.2	0.2	879.6	140.2	5/5	2.46	0 :11 :34.43
25	30	320.6	20.6	0.0	2338.4	792.2	5/5	4.86	0 :59 :55.25
30	10	419.4	2.4	0.0	364.8	27.0	5/5	0.58	0 :02 :22.14
30	15	485.4	5.4	0.2	750.2	1406.3	5/5	1.55	0 :10 :25.21
30	20	485.6	7.0	0.2	941.4	143.4	5/5	2.02	0 :17 :36.16
30	30	352.6	20.0	0.6	3259.2	654.07	4/5	-	1 :48 :30.62
35	10	473	4.2	0.2	787.4	79.8	5/5	1.22	0 :13 :48.24
35	15	492	10.0	1.0	2958.0	527.8	5/5	2.35	1 :38 :06.83
35	20	461.6	649.4	1.0	3042.8	491.0	4/5	-	2 :55 :06.68
35	30	398.8	396.8	0.0	3676.0	502.2	0/5	-	5 :00 :00.00
40	10	605.8	2.6	0.4	776.0	113.8	5/5	1.05	0 :20 :37.71
40	15	574.8	11.6	1.4	2102.4	382.6	4/5	-	1 :37 :53.85
40	20	511.4	22.4	2.4	3323.2	458.6	2/5	-	3 :11 :11.58
40	30	522.0	518.2	0.8	3070.4	257.4	0/5	-	5 :00 :00.00
45	10	562.6	5.8	0.4	1270.6	189.8	5/5	1.28	0 :57 :51.81
45	15	585.0	30.6	1.8	2830.8	333.8	2/5	-	3 :31 :00.00
45	20	498.4	132.2	0.4	3740.4	236.2	0/5	-	5 :00 :00.00
45	30	548.2	137.4	0.8	2504.2	129.8	0/5	-	5 :00 :00.00

TAB. 4.1 – Résultats pour les instances aléatoires

significatif de contraintes de coupe, de coupe-cycle et de partition ont été générées. Cela implique que ces inégalités sont utiles pour résoudre les instances aléatoires. De plus, les contraintes d'étoile-partition ne semblent pas jouer un rôle très important pour ce type d'instances. Cela peut être expliqué par le fait que, pour ces instances, les arêtes n'appartiennent pas nécessairement à un ensemble F_i, et donc, il est difficile de trouver une contrainte d'étoile-partition satisfaisant les conditions d'une configuration d'étoile-partition.

Pour conclure sur ce type d'instances, nous pouvons remarquer que pour la plupart des problèmes, nos contraintes ne sont pas suffisantes pour résoudre le problème dans la phase de coupe. On peut noter, de plus, que le gap est relativement petit quand $|\mathcal{F}| \leq 20$. (Le gap n'apparaît pas dans la table lorsqu'aucune borne supérieure n'a pu être obtenue pour une des cinq instances testées).

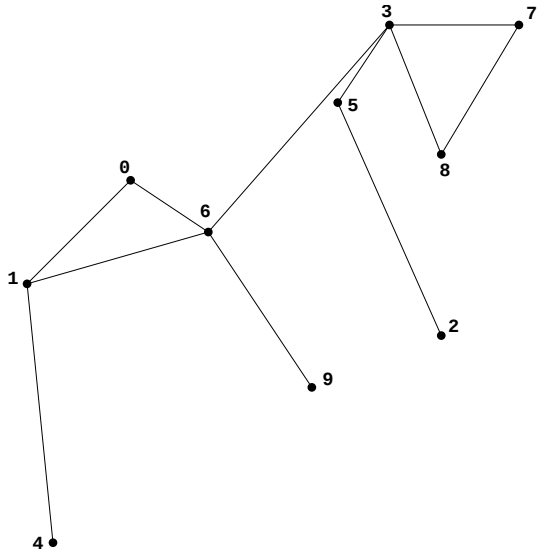
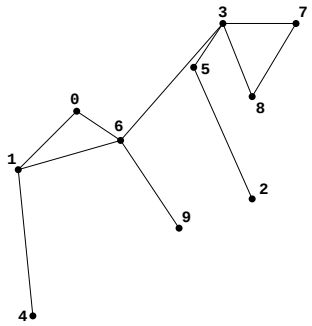


FIG. 4.1 – Instance aléatoire à 10 sommets et $|\mathcal{F}| = 10$

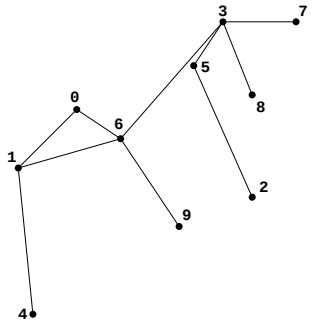
Afin d'illustrer ces résultats, nous donnons une solution obtenue à l'aide de notre algorithme de coupes et branchements, pour une des instances ayant 10 sommets et $|\mathcal{F}| = 10$ (voir figure 4.1). Cette instance contient les 10 premiers sommets de l'instance bier127.tsp de la TSP-Library [76]. Seules apparaissent les arêtes qui ont été prises dans la solution optimale. Soit x^* la solution optimale trouvée, $G_{x^*} = (V, E_{x^*})$ où E_{x^*} est l'ensemble des arêtes $e \in E$ telles que $x^*(e) > 0$ est appelé le *graphe support* de x^*. La figure 4.2 présente les ensembles d'arêtes F_i qui ont été considérés ainsi que les graphes représentant $G_{x^*} \backslash F_i$ pour $i = 1, \ldots, 10$. On remarque que tous ces graphes sont bien connexes, ce qui vérifie la faisabilité de notre solution x^*. On peut aussi remarquer que certains graphes comme $G_{x^*} \backslash F_1$, $G_{x^*} \backslash F_7$ et $G_{x^*} \backslash F_8$ sont identiques à G_{x^*}. Cela signifie que les arêtes de F_1, F_7 et F_8 ne sont pas prises dans la solution. De même, les graphes $G_{x^*} \backslash F_2$ et $G_{x^*} \backslash F_9$ sont identiques, mais différents de G_{x^*}. Cela signifie que $F_2 \cap E_{x^*} = F_9 \cap E_{x^*}$.

La figure 4.3 présente une instance avec 40 sommets et $|\mathcal{F}| = 10$. Cette instance est tirée de l'instance eil51.tsp de la TSP-Library [76] dont les coûts sur les arêtes sont des distances euclidiennes entre les villes. Cette instance a été résolue à l'optimum en un peu moins de 12 minutes. On peut remarquer que la solution n'est pas une solution

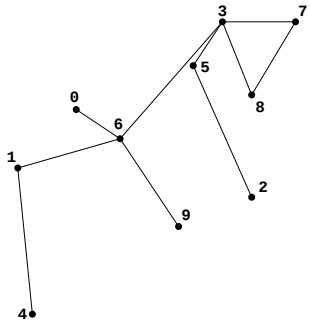
$F_1 = \{(1,7)\}$,
$F_2 = \{(0,9),(2,8),(4,8),(5,8),(6,7),(7,8)\}$,
$F_3 = \{(0,1),(0,7),(0,8),(2,6),(4,8),(8,9)\}$,
$F_4 = \{(0,4),(2,3),(2,9),(4,6),(4,9),(6,8),(7,8)\}$,
$F_5 = \{(0,4),(0,5),(0,7),(1,6),(2,6),(3,7),(3,9),(5,6)\}$,
$F_6 = \{(0,4),(2,6),(2,7),(3,4),(3,7),(5,9)\}$,
$F_7 = \{(0,7),(1,9),(4,7)\}$,
$F_8 = \{(0,2),(0,7),(1,2),(1,7),(1,9),(2,9),(4,8),(4,9)\}$,
$F_9 = \{(2,8),(2,9),(7,8)\}$,
$F_{10} = \{(0,1),(3,8)\}$,
où (i,j) représente l'arête entre les sommets i et j.



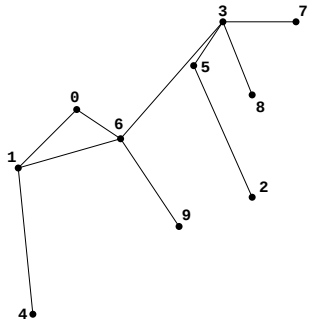
$G_{x^*} \backslash F_1$



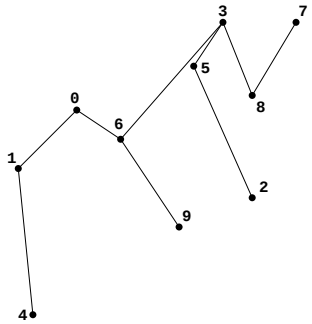
$G_{x^*} \backslash F_2$



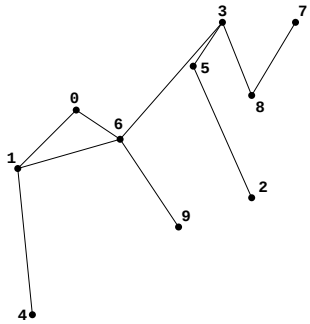
$G_{x^*} \backslash F_3$



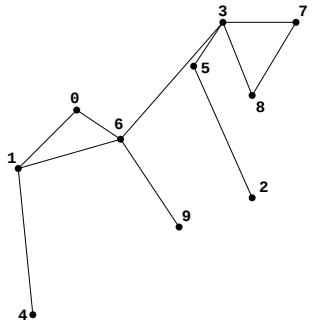
$G_{x^*} \backslash F_4$



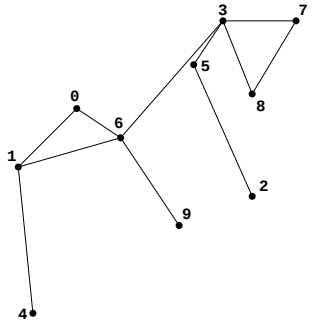
$G_{x^*} \backslash F_5$



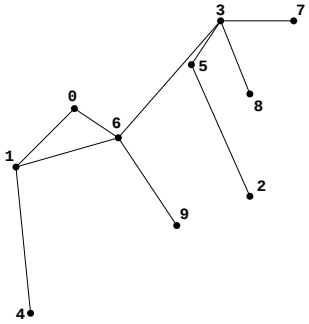
$G_{x^*} \backslash F_6$



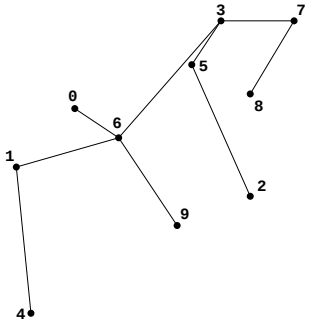
$G_{x^*} \backslash F_7$



$G_{x^*} \backslash F_8$



$G_{x^*} \backslash F_9$



$G_{x^*} \backslash F_{10}$

FIG. 4.2 – Graphes $G_{x^*} \backslash F_i$ pour l'instance à 10 sommets et $|\mathcal{F}| = 10$ de la figure 4.1

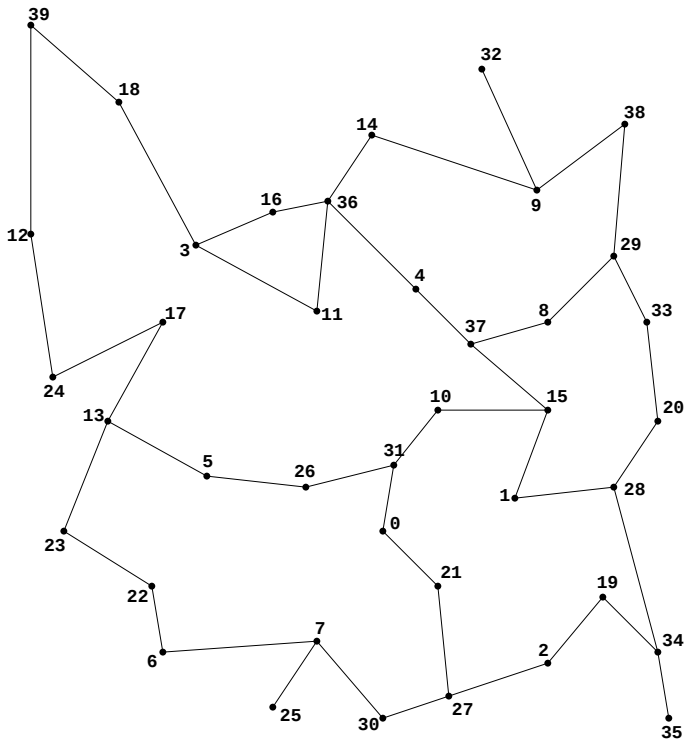


FIG. 4.3 – Instance aléatoire à 40 sommets et $|\mathcal{F}| = 10$

du problème du sous-graphe 2-arête connexe. Cela vient du fait que les ensembles $F_i, i = 1, \ldots, 10$ sont générés aléatoirement et qu'une arête peut n'appartenir à aucun de ces ensembles. Cela signifie que cette arête ne tombera jamais en panne. C'est par exemple le cas de l'arête entre les sommets 9 et 32.

4.2.4 Instances réelles

Les tables 4.2 à 4.4 présentent les résultats pour les instances réelles obtenus pour différentes fonctions coût. Le coût associé à chaque lien dans le réseau client est directement lié au chemin de routage correspondant dans le réseau de transport, et par conséquent, il dépend du coût de ce chemin. Le coût $c(f)$ d'un lien f dans le réseau IP est donné par

$$c(f) = c + \omega(f),$$

où c est un coût fixe représentant la construction de f dans le réseau IP et les équipements nécessaires à cette installation, et $\omega(f)$ est un coût dépendant du réseau optique.

4.2.4.1 Fonction coût $c_1(.)$

L'installation d'un segment optique implique un coût fixe à chaque extrémité du segment. Donc, une première estimation possible du coût optique $\omega(f)$ est la somme des coûts fixes associés aux segments optiques de P_f. Comme ces coûts peuvent être considérés comme identiques pour un réseau optique considéré, une bonne approche serait de considérer un coût $\omega(f)$ proportionnel au nombre de segments optiques de P_f.

$\|V^1\|$	$\|\mathcal{F}\|$	NC	NCC	NSP	NP	NT	Copt	Gap	TT
18	31	70	5	5	31	1	111	0.00	0 :00 :11.72
25	39	32	3	1	39	1	150	0.00	0 :00 :24.83
25	40	44	5	17	68	11	154	0.00	0 :01 :12.98
28	45	311	15	5	762	1	168	0.00	0 :05 :24.17
28	46	176	10	13	189	3	171	0.00	0 :01 :45.34
30	55	176	14	18	220	1	185	0.00	0 :02 :43.25
31	46	133	0	0	46	1	188	0.00	0 :00 :29.93
32	56	180	17	39	1182	17	196	0.00	0 :19 :41.83
33	54	74	6	1	54	1	198	0.00	0 :02 :28.36
34	62	53	6	0	62	1	206	0.00	0 :01 :33.73
35	63	614	17	110	2678	705	*223*	3.14	5 :00 :00.00
36	68	581	15	9	68	1	218	0.00	0 :07 :19.62
38	65	303	19	13	249	11	229	0.00	0 :15 :57.94
40	76	432	24	40	1263	23	245	0.00	1 :03 :55.41
41	73	453	21	45	977	19	252	0.00	0 :38 :51.53
42	70	546	25	8	937	9	254	0.00	0 :35 :03.26
43	74	635	20	11	252	11	260	0.00	0 :31 :20.90
45	78	468	24	7	319	9	272	0.00	0 :40 :24.27
45	86	253	16	13	186	9	274	0.00	0 :33 :11.47
46	85	652	25	38	432	37	281	0.00	1 :11 :58.58
47	84	878	17	2	84	1	282	0.00	0 :25 :29.03
48	91	851	29	43	852	43	291	0.00	1 :52 :48.22
49	88	342	26	24	128	7	295	0.00	1 :07 :56.12
50	83	725	33	34	2539	67	304	0.33	2 :53 :14.94
50	95	419	20	27	191	7	304	0.33	0 :56 :53.31
55	95	768	34	14	2689	5	334	0.00	2 :58 :26.05
60	102	588	37	23	2335	17	-	-	5 :00 :00.00

TAB. 4.2 – Résultats pour les instances réelles avec la fonction coût $c_1(.)$

Par conséquent, nous avons défini une première fonction coût $\omega(f)$ qui consiste en le nombre de liens du chemin optique entre les brasseurs correspondant aux extrémités

de l'arête f. Ici, nous supposons qu'il y a un coût fixe associé à chaque lien optique. Ce coût est considéré à chaque fois que le lien optique correspondant est utilisé. Donc, le coût $c(f)$ est donné dans ce cas par $c_1(f) = c + |P_f|$.

La table 4.2 rapporte les résultats obtenus quand le coût sur chaque arête f de la couche IP est donné par $c_1(f)$. On peut remarquer que toutes les instances ont été résolues à l'optimum excepté celles à 35 et 60 sommets. Pour la plupart de ces instances, le gap est égal à 0. Néanmoins, l'agorithme a eu besoin d'effectuer une phase branchement pour obtenir une solution optimale réalisable.

Toutes les instances avec moins de 38 sommets, excepté celle à 35 sommets, ont été résolues à l'optimum en moins de 20 minutes. Le reste des instances, excepté celle à 60 sommets, ont pu être résolues en moins de 3 heures.

Pour l'instance à 35 sommets, une solution réalisable de valeur 223 (donnée en italique) a été obtenue, et on obtient un gap relativement petit de 3.14%. Pour l'instance à 60 sommets, nous n'avons pas obtenu de borne supérieure (solution réalisable) en moins de 5 heures.

4.2.4.2 Fonction coût $c_2(.)$

Les opérateurs de télécommunications considèrent souvent que la longueur d'un segment optique doit être considérée dans le coût du segment. Par conséquent, ils proposent un modéle de coût sur les chemins linéairement dépendant de la longueur kilométrique du chemin. Dans ce cas, $c(f)$ peut être défini par

$$c_2(f) = c + \sum_{e \in P_f} l(e),$$

où $l(e)$ est la longueur de e. Pour nos expérimentations, nous avons considéré $l(e)$ comme étant la distance réelle en kilomètres entre les extrémités de e.

La table 4.3 donne les résultats pour les instances réelles avec la fonction coût $c_2(.)$. On peut noter que toutes les instances, excepté celle à 60 sommets, ont été résolues à l'optimum et en moins de 2 heures. La plupart des instances avec moins de 40 sommets ont été résolues en moins de 10 minutes. On peut également remarquer que le gap ne dépasse pas 1.5% pour la plupart des instances.

$\|V^1\|$	$\|\mathcal{F}\|$	NC	NCC	NSP	NP	NT	Copt	Gap	TT
18	31	84	5	4	28	101	14999.70	3.13	0 :00 :51.46
25	39	82	5	0	0	1	8968.24	0.00	0 :00 :14.73
25	40	127	6	18	0	19	17538.70	0.03	0 :01 :10.05
28	45	113	8	3	71	3	10764.60	0.05	0 :01 :02.66
28	46	425	12	37	218	163	18035.50	2.57	0 :12 :57.25
30	55	25	5	14	0	1	19554.30	0.00	0 :00 :45.30
31	46	103	3	6	0	1	5775.00	0.00	0 :00 :30.62
32	56	236	15	60	977	405	19406.60	2.83	1 :17 :58.67
33	54	200	9	2	85	3	12472.80	0.04	0 :01 :36.00
34	62	386	11	18	151	9	18466.90	0.71	0 :09 :00.50
35	63	62	9	34	0	25	20574.30	0.10	0 :07 :47.10
36	68	128	5	6	0	1	18061.30	0.00	0 :01 :30.68
38	65	226	16	5	258	13	14778.90	0.06	0 :08 :52.99
40	76	377	11	31	294	45	21048.10	1.38	0 :41 :04.27
41	73	283	8	10	0	1	21137.80	0.00	0 :04 :16.04
42	70	335	22	34	368	21	6164.22	0.51	0 :29 :58.31
43	74	324	16	7	327	1	16536.30	0.00	0 :13 :46.26
45	78	412	16	10	245	1	16768.20	0.00	0 :17 :00.81
45	86	101	1	0	0	1	1152.79	0.00	0 :01 :15.16
46	85	484	10	26	4	15	22028.80	0.53	0 :20 :34.91
47	84	656	21	4	294	1	16762.20	0.00	0 :21 :34.26
48	91	510	9	22	4	15	20811.20	0.56	0 :28 :01.64
49	88	667	22	4	88	1	17115.50	0.00	0 :20 :18.75
50	83	573	14	29	620	103	6699.38	0.40	1 :58 :09.95
50	95	485	8	6	333	9	21847.50	0.05	0 :21 :47.15
55	95	227	18	24	95	23	6975.36	0.06	1 :24 :12.13
60	102	785	31	22	846	87	-	-	5 :00 :00.00

TAB. 4.3 – Résultats pour les instances réelles avec la fonction coût $c_2(.)$

4.2.4.3 Fonction coût $c_3(.)$

La dernière fonction coût utilisée pour les instances réelles est définie comme suit. Soit $f = uv$ une arête de G^1. Soit u' et v' les sommets de G^2 correspondant à u et v respectivement. Soit

$$c_3(f) = \begin{cases} \frac{c_2(f)}{|F_e|} & \text{si } e = u'v' \in E_2, \\ c_2(f) & \text{sinon.} \end{cases}$$

La fonction coût $c_3(.)$ dépend du nombre de chemins de routage utilisant $e = u'v'$ ($|F_e|$) dans le réseau optique, si $e \in E_2$. Notre motivation pour considérer cette fonction coût vient du fait que lorsque $|F_e|$ est grand, le lien f peut être essentiel dans le réseau IP. Dans ce cas, $c_3(f)$ devient petit, et donc on peut espérer que f apparaitra dans la solution optimale.

$\|V^1\|$	$\|\mathcal{F}\|$	NC	NCC	NSP	NP	NT	Copt	Gap	TT
18	31	33	0	0	0	1	1573.180	0.00	0 :00 :00.60
25	39	5	0	0	0	1	584.670	0.00	0 :00 :02.73
25	40	0	0	0	0	1	1518.800	0.00	0 :00 :01.15
28	45	50	0	0	0	1	791.918	0.00	0 :00 :03.59
28	46	148	2	0	1	3	1111.630	0.27	0 :00 :26.68
30	55	11	0	0	0	1	1701.370	0.00	0 :00 :07.54
31	46	6	0	0	0	1	526.308	0.00	0 :00 :08.19
32	56	12	0	0	0	1	1027.510	0.00	0 :00 :09.58
33	54	62	1	0	0	1	1293.830	0.00	0 :00 :14.49
34	62	85	1	0	0	1	1055.990	0.00	0 :00 :32.75
35	63	8	0	0	0	1	1590.780	0.00	0 :00 :22.67
36	68	6	0	0	0	1	1053.930	0.00	0 :00 :13.64
38	65	69	3	0	0	1	1300.670	0.00	0 :00 :45.73
40	76	0	0	0	0	1	1077.520	0.00	0 :00 :08.29
41	73	23	1	0	0	1	1455.400	0.00	0 :01 :16.68
42	70	13	3	0	0	1	184.324	0.00	0 :01 :00.03
43	74	130	5	0	0	1	1600.370	0.00	0 :03 :18.41
45	78	185	7	0	0	1	1522.030	0.00	0 :03 :24.01
45	86	101	1	0	0	1	1152.790	0.00	0 :01 :15.16
46	85	12	0	0	0	1	1029.461	0.00	0 :00 :45.42
47	84	546	10	0	0	1	1545.770	0.00	0 :05 :08.76
48	91	15	0	0	0	1	701.019	0.00	0 :01 :21.16
49	88	326	4	0	0	1	971.864	0.00	0 :05 :24.76
50	83	100	2	0	0	1	186.281	0.00	0 :03 :05.83
50	95	0	0	0	0	1	716.747	0.00	0 :00 :24.12
55	95	0	1	0	0	1	185.075	0.00	0 :01 :39.83
60	102	140	6	0	0	1	166.458	0.00	0 :10 :08.39
62	106	68	5	0	0	1	152.727	0.00	0 :11 :49.14
64	108	48	5	3	0	1	153.397	0.00	0 :23 :51.52
66	110	70	6	3	0	1	148.888	0.00	0 :23 :57.35
68	112	52	7	3	0	1	143.885	0.00	0 :30 :18.52
70	115	63	6	2	0	1	144.642	0.00	0 :35 :59.39
73	120	57	5	3	0	1	220.098	0.00	0 :36 :29.80
76	125	66	5	3	0	1	134.406	0.00	0 :44 :40.26
78	130	56	4	2	0	1	146.768	0.00	0 :38 :15.57
82	139	81	6	1	0	1	201.154	0.00	0 :52 :21.51
85	146	67	4	1	0	1	243.363	0.00	1 :00 :08.06
88	152	80	4	2	0	1	254.725	0.00	1 :00 :00.05
95	165	83	1	1	0	1	372.082	0.00	1 :29 :30.30
100	175	95	2	0	0	1	592.832	0.00	1 :04 :00.23
110	119	60	2	1	0	1	692.232	0.00	1 :36 :36.00
112	205	80	6	1	0	1	531.180	0.00	2 :43 :10.53
115	211	81	4	1	0	1	392.770	0.00	3 :45 :49.09
118	218	156	7	0	0	1	-	-	5 :00 :00.00
120	221	143	7	0	0	1	-	-	5 :00 :00.00

TAB. 4.4 – Résultats pour les instances réelles avec la fonction coût $c_3(.)$

Les résultats obtenus avec cette fonction coût sont présentés dans la table 4.4. Il semble qu'ici le problème est plus facile à résoudre. En effet, toutes les instances de moins de 60 sommets ont été résolues à l'optimum en moins de 10 minutes. On peut également remarquer qu'aucune contrainte de partition et aucune contrainte d'étoile-partition n'ont été générée pour toutes ces instances, excepté celle à 28 sommets.

Les instances entre 60 et 115 sommets ont toutes été résolues à l'optimum en moins de 5 heures. On peut remarquer que pour ces instances, des contraintes d'étoile-partition sont utiles et il y a toujours, comme pour les instances plus petites, des contraintes de coupe-cycle générées. Le temps d'exécution augmente progressivement avec la taille de l'instance jusqu'à l'instance à 118 sommets qui n'est pas résolue dans le temps imparti.

La résolution de toutes les instances, excepté celle à 28 sommets et $|\mathcal{F}| = 46$ s'est terminée dans la phase de coupe (sans branchement). Aucune contrainte de partition n'est généré. Pour cette instance particulière, nous avons un tout petit gap de 0.27%.

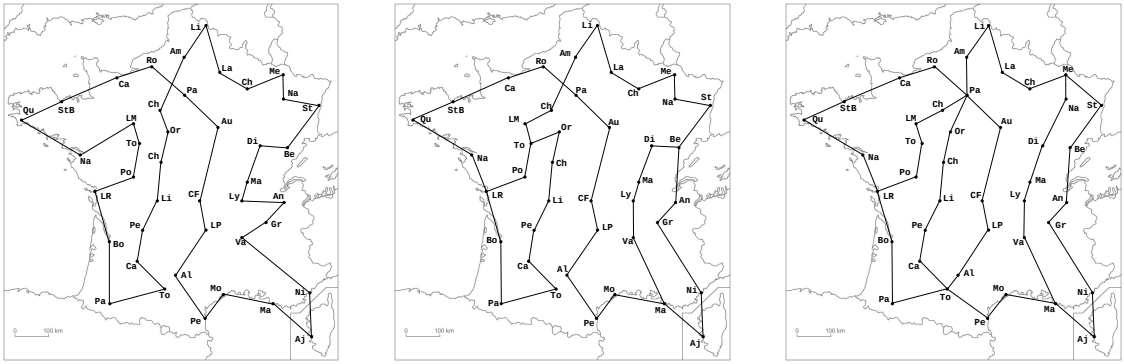
Ces trois tables permettent de constater que la difficulté pour résoudre les instances réelles dépend énormément de la fonction coût associée aux arêtes du réseau IP. Il apparaît également que, plus la fonction coût est uniforme, plus il existe de solutions équivalentes, ce qui entraîne une dégénérescence du problème le rendant difficile à résoudre. C'est ce qui explique les différences entre les résultats des différentes tables.

4.2.4.4 Une instance française

Nous présentons maintenant, plus en détail, une instance composée de 42 sommets représentant des villes françaises et dont la couche transport comprend 70 liaisons entre ces villes. La figure 4.4 représente la couche optique pour cette instance. La figure 4.5 donne les solutions trouvées pour les trois fonctions coûts.

Cette instance a été résolue en une demi heure pour les fonctions coût $c_1(.)$ et $c_2(.)$ et en une minute pour la fonction coût $c_3(.)$. Pour la fonction coût $c_1(.)$, la solution est un tour, c'est à dire que la solution comprend exactement 2 arêtes incidentes à chaque sommet du graphe représentant la couche cliente. Pour la fonction coût $c_3(.)$, la solution est beaucoup plus dense et centrée sur Paris. On peut remarquer que certaines liaisons installées sur la couche cliente ne sont pas présentes sur la couche transport. C'est par exemple le cas de la liaison entre Amiens et Chartres pour les fonctions coût $c_1(.)$ et $c_2(.)$.

FIG. 4.4 – Couche transport pour l'instance réelle à 42 sommets et 70 arêtes



Fonction coût $c_1(.)$ Fonction coût $c_2(.)$ Fonction coût $c_3(.)$

FIG. 4.5 – Solution pour la couche cliente avec les différentes fonctions coût

4.2.4.5 Utilité des contraintes de coupe-cycle et d'étoile-partition

Pour les fonctions coût $c_1(.)$ et $c_2(.)$, un nombre significatif de contraintes de coupe-cycle et d'étoile-partition ont été générées. Dans le but d'évaluer l'impact de ces inégalités ainsi que celui des opérations de réduction $\theta_1, \ldots, \theta_4$ sur la performance de l'algorithme, nous avons reporté dans les tables 4.5, 4.6 et 4.7, les résultats obtenus pour des instances réelles sans utiliser ni ces contraintes, ni les opérations de réduction. Pour ces 3 tables, chaque instance est représentée par deux lignes. La première donne les résultats sans utiliser ni les contraintes de coupe-cycle, ni les contraintes d'étoile-partition. La deuxième ligne donne les résultats quand on sépare également ces contraintes.

$\|V^1\|$	$\|\mathcal{F}\|$	NC	NCC	NSP	NP	NT	Copt	Gap	TT
30	55	229	-	-	1133	33	185	0.54	0 :09 :42.13
30	55	176	14	18	220	1	185	0.00	0 :02 :43.25
35	63	213	-	-	6404	127	-	-	5 :00 :00.00
35	63	614	17	110	2678	705	*223*	3.14	5 :00 :00.00
41	73	231	-	-	397	17	-	-	5 :00 :00.00
41	73	453	21	45	977	19	252	0.00	0 :38 :51.53
46	85	305	-	-	6942	43	-	-	5 :00 :00.00
46	85	652	25	38	432	37	281	0.00	1 :11 :58.58
48	91	307	-	-	5248	65	-	-	5 :00 :00.00
48	91	851	29	43	852	43	291	0.00	1 :52 :48.22
50	95	297	-	-	3305	143	-	-	5 :00 :00.00
50	95	419	20	27	191	7	304	0.33	0 :56 :53.31

TAB. 4.5 – Résultats pour des instances réelles avec la fonction coût $c_1(.)$, avec ou sans les contraintes de coupe-cycle, ni les contraintes d'étoile-partition

Pour la fonction coût $c_1(.)$, cinq instances de la table 4.5 n'ont pas été résolues dans la limite de 5 heures. En utilisant les opérations de réduction et en ajoutant les contraintes de coupe-cycle et d'étoile-partition, quatre de ces cinq instances ont été résolues en moins de 2 heures. Pour la dernière, qui n'a pas été résolue, comme indiqué à la table 4.2, nous avons obtenu une solution réalisable. De plus, pour l'instance à 30 sommets, 33 nœuds ont été générés dans l'arbre de branchement, alors que la résolution de cette instance, table 4.2, a été achevée dans la phase de coupe (sans branchement).

Pour le deuxième ensemble d'instances de la table 4.6, qui utilisent la fonction coût $c_2(.)$, les instances qui ont été résolues à l'optimum en utilisant les contraintes de coupe-cycle et d'étoile-partition, ont également été résolues sans ces inégalités, mais le temps de calcul a plus que doublé. Par exemple, le problème avec 50 sommets qui était résolu en moins de 22 minutes, en utilisant les contraintes de coupe-cycle et d'étoile-partition, demande plus de 2 heures de calcul sans utiliser ces inégalités. De plus, on

$\|V^1\|$	$\|\mathcal{F}\|$	NC	NCC	NSP	NP	NT	Copt	Gap	TT
30	55	128	-	-	204	37	19554.3	0.05	0 :03 :36.32
30	55	25	5	14	0	1	19554.30	0.00	0 :00 :45.30
31	46	148	-	-	130	15	5775.0	0.09	0 :01 :58.04
31	46	103	3	6	0	1	5775.00	0.00	0 :00 :30.62
35	63	196	-	-	706	295	20574.3	0.41	0 :47 :31.20
35	63	62	9	34	0	25	20574.30	0.10	0 :07 :47.10
41	73	267	-	-	765	39	21137.8	0.18	0 :25 :31.08
41	73	283	8	10	0	1	21137.80	0.00	0 :04 :16.04
48	91	229	-	-	330	47	20811.2	0.71	0 :47 :23.91
48	91	510	9	22	4	15	20811.20	0.56	0 :28 :01.64
50	95	285	-	-	1870	71	21847.5	1.34	2 :07 :04.73
50	95	485	8	6	333	9	21847.50	0.05	0 :21 :47.15

TAB. 4.6 – Résultats pour des instances réelles avec la fonction coût $c_2(.)$, avec ou sans les contraintes de coupe-cycle, ni les contraintes d'étoile-partition

peut remarquer que la taille de l'arbre de branchement a énormémement augmenté pour toutes les instances. On peut également observer le fait que les instances avec 30, 31 et 41 sommets ont été résolues dans la table 4.3 sans aucun branchement.

$\|V^1\|$	$\|\mathcal{F}\|$	NC	NCC	NSP	NP	NT	Copt	Gap	TT
28	46	69	-	-	23	7	1111.6	1.15	0 :00 :37.50
28	46	148	2	0	1	3	1111.630	0.27	0 :00 :26.68
41	73	24	-	-	0	5	1455.4	0.02	0 :02 :13.68
41	73	23	1	0	0	1	1455.400	0.00	0 :01 :16.68
64	108	55	-	-	8	35	153.4	1.26	1 :19 :37.87
64	108	48	5	3	0	1	153.397	0.00	0 :23 :51.52
76	125	63	-	-	0	27	134.41	0.90	1 :46 :48.23
76	125	66	5	3	0	1	134.406	0.00	0 :44 :40.26
88	152	80	-	-	0	29	254.73	0.47	3 :34 :03.96
88	152	80	4	2	0	1	254.725	0.00	1 :00 :00.05
110	199	60	-	-	0	15	692.23	0.18	4 :24 :22.52
110	119	60	2	1	0	1	692.232	0.00	1 :36 :36.00
112	205	80	-	-	0	15	-	-	5 :00 :00.00
112	205	80	6	1	0	1	531.180	0.00	2 :43 :10.53

TAB. 4.7 – Résultats pour des instances réelles avec la fonction coût $c_3(.)$, avec ou sans les contraintes de coupe-cycle, ni les contraintes d'étoile-partition

La dernière série d'instances de la table 4.7 correspondent à la fonction coût $c_3(.)$. On peut également remarquer que les contraintes de coupe-cycle sont utiles pour résoudre ces problèmes. On peut remarquer que l'instance à 41 sommets a nécessité 5 nœuds dans l'arbre de branchement, sans utiliser les contraintes de coupe-cycle et d'étoile-

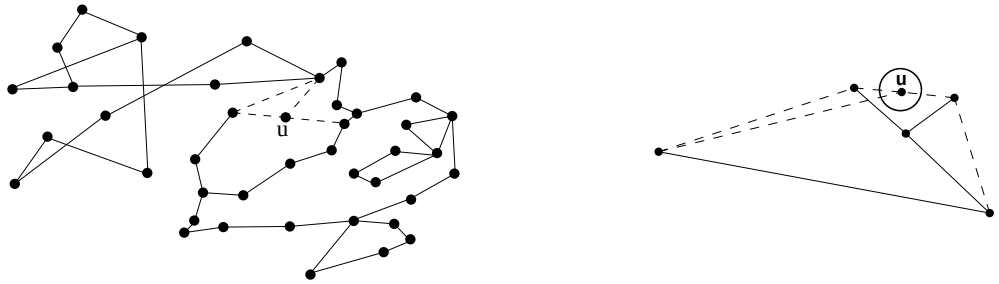


FIG. 4.6 – Point fractionnaire pour l'instance à 41 sommets et 73 arêtes avec le coût $c_3(.)$ sans l'ajout des contraintes de coupe-cycle

partition. La figure 4.6 donne le point fractionnaire avant branchement ainsi que le graphe réduit associé. L'ajout de la contrainte de coupe-cycle associée au sommet u permet d'obtenir la solution optimale représentée sur la figure 4.7 qui est donc obtenue sans aucun branchement.

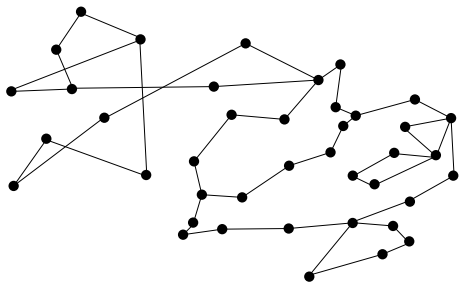


FIG. 4.7 – Solution optimale pour l'instance à 41 sommets et 73 arêtes avec le coût $c_3(.)$

On peut également remarquer que les contraintes de coupe-cycle ainsi que les contraintes d'étoile-partition ont été très utiles pour résoudre les instances de plus de 60 sommets. En effet, l'introduction des contraintes de coupe-cycle et d'étoile-partition réduit considérablement le temps d'exécution ainsi que le nombre de nœuds dans l'arbre de branchement. Par exemple, l'instance à 64 sommets et celle à 88 sommets ont été résolues en 3 fois moins de temps en utilisant les contraintes de coupe-cycle et d'étoile-partition. Il en est de même pour l'instance à 110 sommets qui nécessitait 15 nœuds dans l'arbre de branchement et une durée de presque 4h30 pour être résolue sans utiliser les contraintes de coupe-cycle et d'étoile-partition alors qu'elle est résolue en un peu plus de 1h30 et sans aucun branchement en utilisant ces contraintes. De même, l'instance à 112 sommets qui n'était pas résolue au bout de 5 heures et 15 nœuds dans l'arbre de branchement, est résolue en moins de 3 heures et sans aucun branchement avec l'introduction des contraintes de coupe-cycle et d'étoile-partition.

Finalement, notons qu'un grand nombre des contraintes de partition, de coupe-cycle et d'étoile-partition, coupant les solutions fractionnaires dans les expérimentations, définissent des facettes dans G' et donc dans G.

4.3 Conclusion

Dans ce chapitre, nous avons décrit un algorithme de coupes et branchements pour le problème de sécurisation multicouche du réseau IP. Nous avons appliqué cet algorithme à des instances aléatoires basées sur les instances de la TSP Library [76] et nous avons également pu résoudre des instances réelles fournies par France Telecom. Lorsque notre algorithme n'a pas pu résoudre certaines instances dans le temps qui lui était imparti, une très bonne solution heuristique a néanmoins été obtenue et le gap entre la solution fractionnaire avant branchement et la meilleure solution réalisable est toujours très faible. Nous avons montré l'impact que peuvent avoir les contraintes de coupe-cycle et les contraintes d'étoile-partition, ainsi que les opérations de réduction, sur la résolution des instances. En approfondissant cette étude, il sera possible de résoudre des instances de plus grande taille.

Chapitre 5

Sécurisation multicouche avec capacités du réseau IP

Dans ce chapitre, nous considérons un problème de dimensionnement et de sécurisation multicouche du réseau IP. Ce nouveau problème est une extension du problème MSIPND. En plus de la sécurisation de la couche cliente, ce problème consiste également à déterminer les capacités à installer sur les liaisons du réseau. Nous présentons tout d'abord deux versions du problème de sécurisation multicouche avec capacités du réseau IP et le modélisons en termes de graphes. Nous donnons ensuite deux formulations sous forme de programmes linéaires mixtes pour chaque version du problème. Nous introduisons de nouvelles classes de contraintes valides pour les quatre formulations et nous discutons de conditions nécessaires et de conditions suffisantes pour qu'elles définissent des facettes. Ces résultats seront utilisés dans le chapitre suivant pour développer des algorithmes de coupes et branchements ainsi que des algorithmes de coupes, génération de colonnes et branchements pour résoudre le problème.

5.1 Présentation du problème MCSIPND

On suppose que la topologie et le routage de la couche transport sont fixés et satisfont des conditions de fiabilité. En effet, on dispose d'un ensemble de brasseurs optiques reliés par des fibres optiques représentant la couche transport. On connait de plus l'ensemble des routeurs IP constituant la couche cliente. A chaque routeur est associé un brasseur de la couche optique. Le routage de la couche transport étant connu, on peut déterminer, pour chaque liaison e de la couche transport, l'ensemble des liaisons

de la couche cliente qui seront affectées par la panne de la liaison optique e. Ces données du problème sont similaires à celles du problème MSIPND et sont illustrées par la figure 3.1 de la page 46.

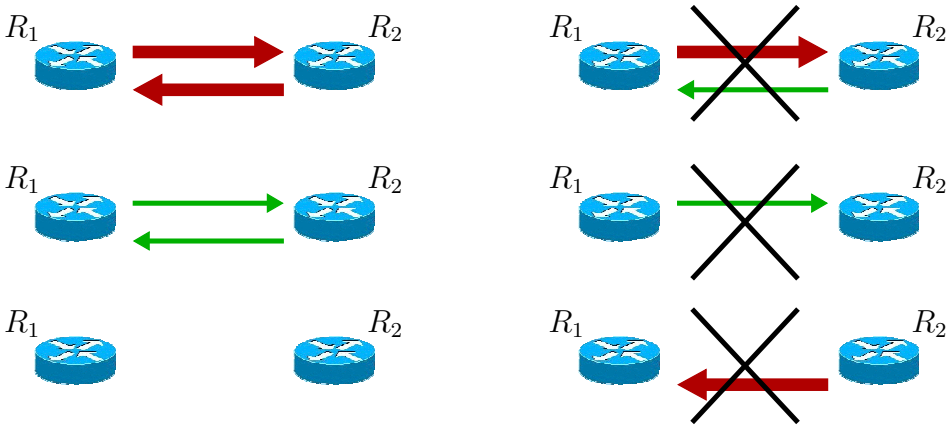


FIG. 5.1 – Capacités symétriques

Pour ce nouveau problème, on suppose que l'on peut construire des liaisons dans la couche cliente entre chaque paire de routeurs, chacune pouvant avoir une capacité de 2.5 ou de 10 Gbit/s dans chaque sens. Les capacités installées doivent être symétriques, c'est-à-dire que l'on doit installer la même capacité dans chacun des deux sens. La figure 5.1 présente les configurations possibles à gauche et les configurations interdites à droite. On peut par exemple installer une capacité de 10 Gbit/s dans les deux sens ou bien une capacité de 2.5 Gbit/s dans les deux sens ou aucune capacité. En revanche, on ne peut pas installer une capacité de 10 Gbit/s dans un sens et une de 2.5 Gbit/s dans l'autre sens, ou bien une capacité dans un sens et aucune dans l'autre. La figure 5.2 donne une autre façon de représenter les capacités sur les liaisons. Un cylindre de diamètre différent va alors représenter une liaison ayant une capacité de 10 Gbit/s dans les deux sens ou une capacité de 2.5 Gbit/s dans les deux sens. Chacune de ces liaisons porte un certain coût fixe d'installation qui dépend entre autre de la capacité.

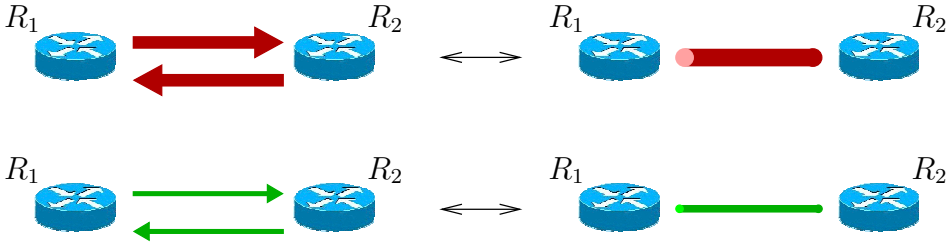


FIG. 5.2 – Capacités symétriques

Comme pour le problème MSIPND, chaque liaison de la couche cliente correspond à

un chemin de la couche transport. Deux liaisons parallèles (de 2.5 ou 10 Gbit/s) seront associées au même chemin optique. On dispose également d'un ensemble de demandes entre les routeurs IP ainsi que le volume de chacune d'entre-elles. On suppose que la capacité des liens optiques est infinie. On remarque que si l'on considère une capacité infinie sur les liaisons de la couche transport optique, tout flot circulant sur la couche cliente pourra être routé sur la couche transport.

Le *problème de sécurisation multicouche avec capacités du réseau IP* (*Multilayer Capacitated Survivable IP Network Design Problem*, MCSIPND Problem) consiste à déterminer les liaisons devant être installées sur la couche cliente pour que la construction soit de coût minimum et le réseau puisse écouler les demandes sur la couche cliente quelque soit la liaison de la couche transport tombant en panne.

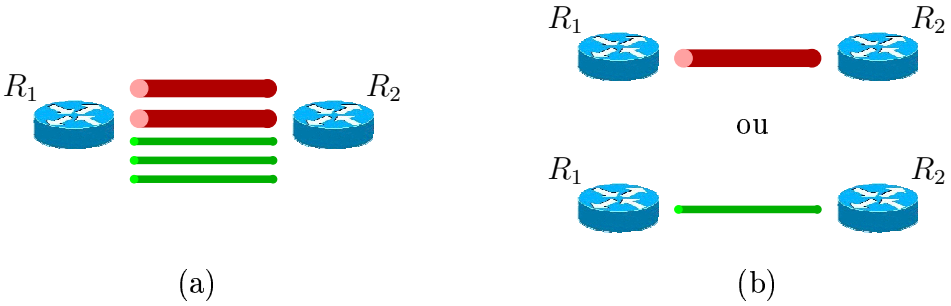


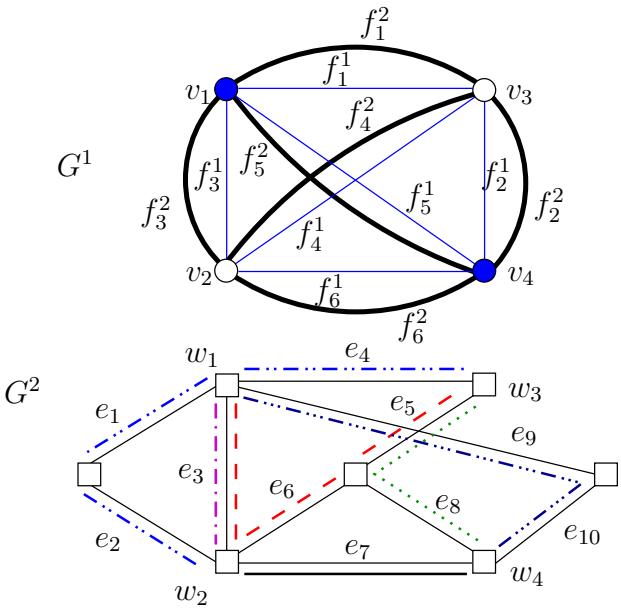
FIG. 5.3 – Deux versions du problème

On distingue deux versions du problème : la version *multiple* et la version *simple*. Dans la première version, on peut installer plusieurs liaisons en parallèle entre deux routeurs de la couche cliente, qu'elles soient de capacité 2.5 Gbit/s ou de capacité 10 Gbit/s. On note ce problème MCSIPND$_m$. La figure 5.3(a) illustre ce cas de figure. Par contre, dans la deuxième version, on n'autorise qu'une seule liaison entre deux routeurs. On peut donc avoir soit une liaison de capacité 10 Gbit/s, soit une liaison de capacité 2.5 Gbit/s, soit aucune liaison (voir figure 5.3(b)). On note ce deuxième problème MCSIPND$_s$. On remarque qu'une solution réalisable pour le problème MCSIPND$_s$ est réalisable pour le problème MCSIPND$_m$. La réciproque n'est bien sûr pas vraie. Par abus de langage, on note MCSIPND quand on considère l'un des deux problèmes MCSIPND$_m$ ou MCSIPND$_s$ indifféremment.

Ce problème, sous ses deux versions, peut être modélisé en termes de graphes en utilisant une modélisation proche de celle du problème MSIPND. On associe à la couche cliente, un graphe $G^1 = (V^1, E^1)$ où les sommets de V^1 correspondent aux routeurs et les arêtes aux liaisons possibles entre les routeurs. On associe à la couche transport, un graphe $G^2 = (V^2, E^2)$ où les sommets de V^2 correspondent aux brasseurs optiques

et les arêtes aux fibres optiques. A chaque sommet $v_i \in V^1$ est associé un sommet $w_i \in V^2$. Chaque arête f de E^1 est associée à un chemin du graphe G^2. Pour chaque arête $e \in E^2$, les ensembles F_e représentant l'ensemble des liaisons de la couche cliente coupées lors d'une panne de l'arête e sont calculés de la même façon que pour le problème MSIPND. Si on note par P_f l'ensemble des arêtes du chemin de G^2 associé à l'arête $f \in E^1$, $F_e = \{f \in E^1 \mid e \in P_f\}$ pour tout $e \in E^2$. On suppose que le routage est fixé et donc P_f est connu pour chaque $f \in E^1$. On a donc F_e connu pour chaque $e \in E^2$. On note $\mathcal{F} = \{F_e, e \in E^2\}$. Deux capacités différentes peuvent être installées sur les liaisons. On considère donc deux arêtes parallèles entre chaque paire de sommets. L'une d'entre elles est dite *de type 1* et représente une liaison de capacité 2.5 Gbit/s, l'autre est dite *de type 2* et représente une liaison de capacité 10 Gbit/s. On note $c^l_{ij} > 0$, $ij \in E^1$ et $l = 1, 2$, le coût fixe d'installation de la liaison entre les sommets i et j de type l. On note K l'ensemble des demandes entre des paires origine-destination de sommets de G^1. On remarque que ces demandes pourraient être modélisées également sous la forme d'un graphe G^0 dont les sommets seraient les sommets de G^1, les arêtes représenteraient les différentes demandes et le poids sur les arêtes représenterait le volume de la demande associée à l'arête.

Le problème MCSIPND consiste à trouver un sous-graphe H de G^1 tel que pour toute arête $e \in E^2$, le sous-graphe obtenu à partir de H auquel on a enlevé les arêtes de F_e puisse écouler les demandes de K et que le coût total soit minimum.



2.5 Gbits/s (type 1)

10 Gbits/s (type 2)

Arête f	Chemin	P_f
f_1^1, f_1^2	– – –	e_3, e_5, e_6
f_2^1, f_2^2	·····	e_5, e_8
f_3^1, f_3^2	–·–·–	e_3
f_4^1, f_4^2	–··–··	e_1, e_2, e_4
f_5^1, f_5^2	–···–···	e_9, e_{10}
f_6^1, f_6^2	———	e_7

$K = \{(v_1, v_4, 9)\}$

FIG. 5.4 – Exemple du problème MCSIPND

La figure 5.4 présente un exemple du problème basé sur la configuration du réseau

multicouche de la figure 3.1 de la page 46. Le graphe G^1 présente la couche cliente. Pour faciliter la lecture de la figure, les arêtes f_i^1 associées à une liaison de capacité 2.5 Gbit/s sont représentées en trait simple tandis que les arêtes f_i^2 en gras représentent les liaisons de capacité 10 Gbit/s. Chaque arête correspond à un chemin dans le graphe G^2 associé à la couche transport. Les arêtes parallèles ont bien sûr le même chemin associé. On suppose que l'on a qu'une seule demande allant du sommet v_1 au sommet v_4 et ayant un volume de 9 unités.

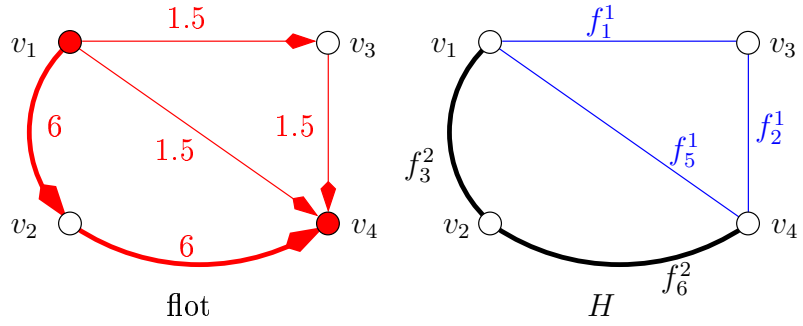


FIG. 5.5 – Flot possible (sans panne)

Les problèmes de dimensionnement et de routage sont très souvent liés à des problèmes de flots. Si l'on considère qu'il ne peut pas y avoir de panne ($i.e$ $\mathcal{F} = \emptyset$), le flot de la figure 5.5 est réalisable pour l'exemple de la figure 5.4. A gauche, on dispose d'un flot réalisable. Le graphe de droite représente l'état du réseau permettant d'écouler ce flot. On remarque que les arêtes f_4^1 et f_4^2 ne supportent aucun flot et donc elles n'apparaissent pas dans H. Il en est de même pour f_1^2, f_2^2, f_3^1, f_5^2 et f_6^1.

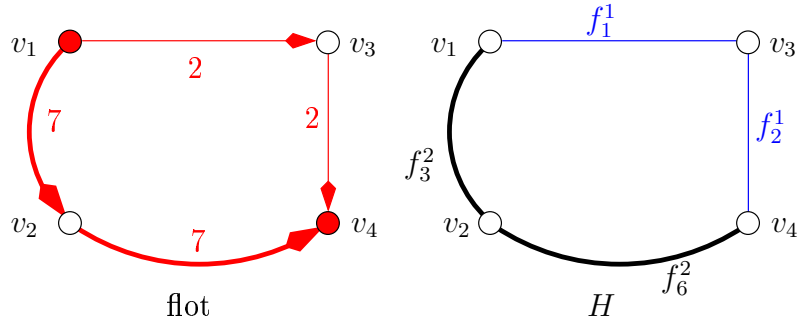


FIG. 5.6 – Flot donnant un réseau moins coûteux (sans panne)

Il existe cependant un autre flot, décrit par la figure 5.6, coutant moins cher et permettant d'assurer la demande entre v_1 et v_4 (toujours en ne considérant aucune panne). En effet, le graphe H obtenu permet toujours l'écoulement de la demande entre v_1 et v_4 de volume 9 et ce nouveau graphe est un sous-graphe de celui de la figure 5.5, il est donc moins coûteux.

On considère maintenant les pannes possibles du problème de la figure 5.4. On sait que $\mathcal{F} = \{F_{e_1}, F_{e_2}, \ldots, F_{e_{10}}\}$ avec $F_{e_1} = F_{e_2} = F_{e_4} = \{f_4^1, f_4^2\}$, $F_{e_3} = \{f_1^1, f_1^2, f_3^1, f_3^2\}$, $F_{e_5} = \{f_1^1, f_1^2, f_2^1, f_2^2\}$, $F_{e_6} = \{f_1^1, f_1^2\}$, $F_{e_7} = \{f_6^1, f_6^2\}$, $F_{e_8} = \{f_2^1, f_2^2\}$ et $F_{e_9} = F_{e_{10}} = \{f_5^1, f_5^2\}$. On peut remarquer que si l'arête e_1 vient à tomber en panne, alors les arêtes f_4^1 et f_4^2 sont coupées, mais le flot de la figure 5.6 continue à être écoulé. C'est encore le cas lorsque l'une des arêtes $e_2, \ldots, e_6$, e_8, e_9 et e_{10} tombe en panne. Par contre, si l'arête e_7 tombe en panne, alors l'arête f_6^1 est coupée et les arêtes f_1^1 et f_2^1 ne possèdent pas une capacité suffisante pour acheminer la demande entre v_1 et v_4. Le graphe H de la figure 5.6 n'est donc pas une solution réalisable du problème pour l'exemple de la figure 5.4 lorsque l'on considère les pannes définies par $\mathcal{F}$.

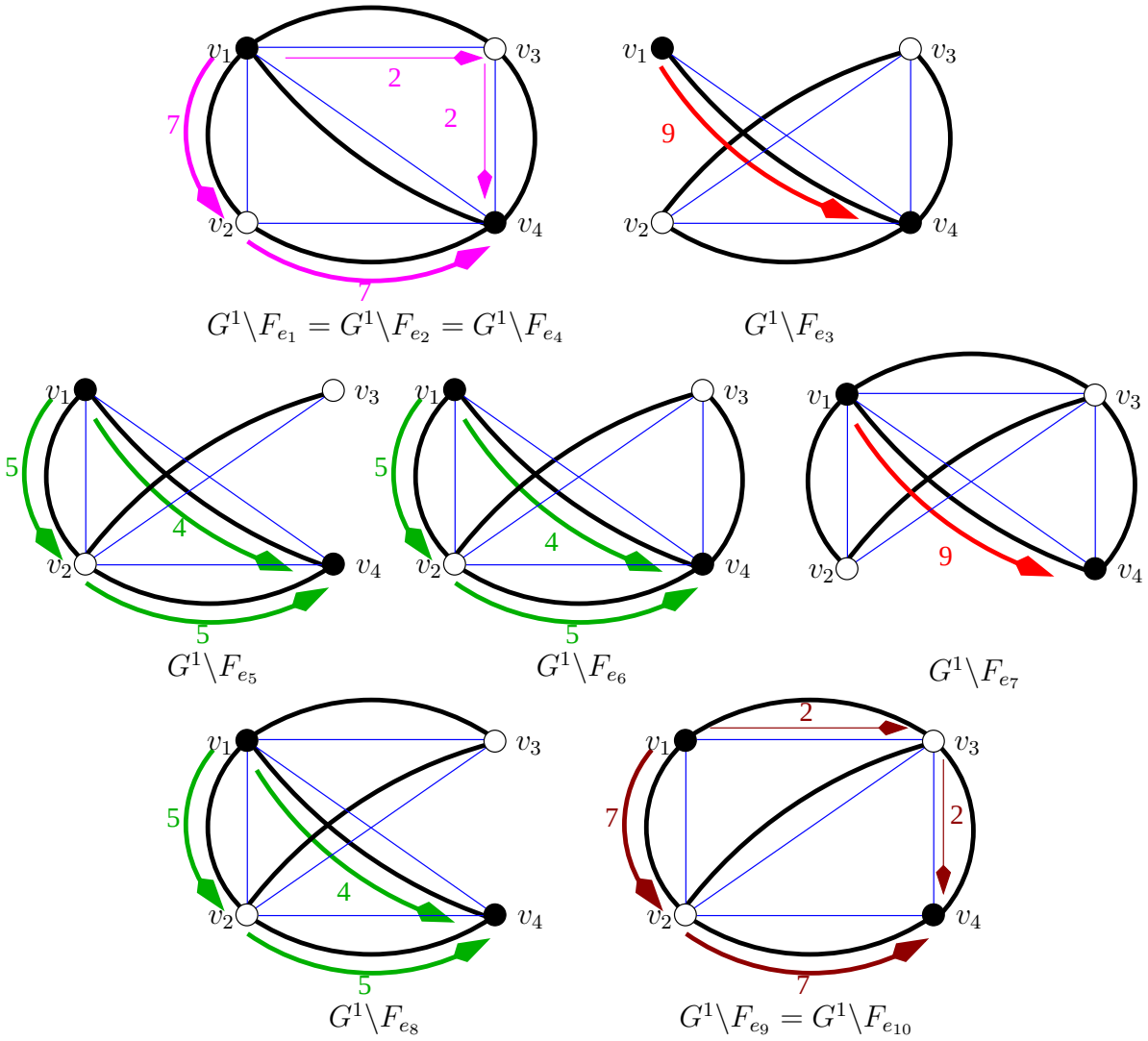
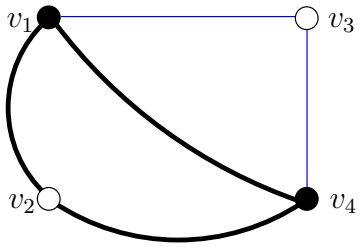


FIG. 5.7 – Flots pour les différents cas de panne

Néanmoins une solution existe. En effet, on peut remarquer que quelque soit l'ensemble d'arêtes $F_e \subseteq E^1$, $e \in E^2$ que l'on supprime dans G^1, il existe toujours un flot permettant d'écouler la demande. La figure 5.7 donne un flot réalisable pour chaque panne possible. On peut remarquer que l'on a ici 4 flots différents.

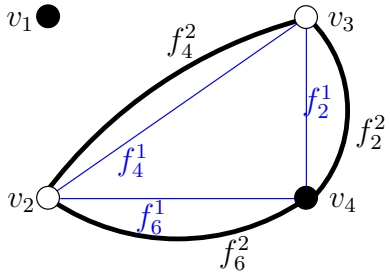


H

FIG. 5.8 – Solution réalisable pour les problèmes MCSIPND$_s$ et MCSIPND$_m$

Les arêtes f_4^1 et f_4^2 ne sont jamais traversées par un flot, donc elles peuvent être supprimées de la solution. Il en est de même pour les arêtes f_1^2, f_2^2, f_3^1 et f_6^1. Le graphe de la figure 5.8 est une solution réalisable du problème MCSIPND$_s$ et par conséquent du problème MCSIPND$_m$.

On peut également montrer qu'il n'existe pas de flot qui soit réalisable quelque soit la panne. La figure 5.9 présente les effets sur G^1 d'une panne simultanée des arêtes e_3 et e_9 de G^2. Il est clair qu'il n'est pas possible de trouver un flot permettant d'écouler la demande entre v_1 et v_4. Ceci montre qu'il n'est pas possible de trouver un flot unique valide pour toutes les pannes.



$G^1 \backslash (F_{e_3} \cup F_{e_9})$

FIG. 5.9 – Panne simultanée de e_3 et e_9

Supposons maintenant que la demande allant de v_1 à v_4 ait un volume strictement supérieur à 10. Il n'existe pas de solution réalisable pour le problème MCSIPND$_s$. En effet, en cas de panne de e_3, la seule liaison permettant de relier v_1 au reste du réseau est la liaison directe entre v_1 et v_4 et la capacité maximum de cette liaison est de 10

Gbit/s. Celle-ci ne pourra donc pas écouler la nouvelle demande. Il existe néanmoins des solutions réalisables pour MCSIPND$_m$. La figure 5.10 donne deux solutions réalisables pour ce problème en considérant une demande de volume 12 entre v_1 et v_4.

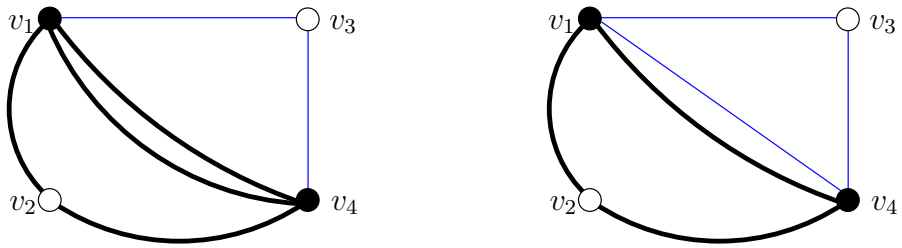


FIG. 5.10 – Solutions réalisables pour le problème MCSIPND$_m$

Dans les sections suivantes, nous présentons des formulations en termes de programmes linéaires mixtes, pour chacune des deux versions du problème (avec ou sans liaisons multiples). Pour chaque variante du problème, nous allons proposer une formulation arcs-sommets et une formulation arcs-chemins. Pour cela, nous construisons le graphe orienté $D^1 = (V^1, A^1)$ à partir du graphe G^1, en remplaçant chaque arête uv de E^1 par les arcs (u, v) et (v, u). On note $\overline{\mathcal{F}} = \{\overline{F}_e, e \in E^2\}$ où $\overline{F}_e = \{(u, v), (v, u) \in A^1 \mid uv \in F_e\}$. Soit K l'ensemble des demandes. Pour $k \in K$, on note par (o_k, d_k) la paire origine-destination de la demande k et par ω_k le volume de cette demande. Soient $\mu^l > 0$, $l = 1, 2$, les capacités possibles pour les arêtes de E^1. Si l'on considère des capacités de 2,5 Gbit/s (type 1) et 10 Gbits/s (type 2), on aura $\mu^1 = 2.5$ et $\mu^2 = 10$. Soit $c^l_{uv} > 0$ le coût d'installation de l'arête $uv \in E^1$ de capacité μ^l, $l = 1, 2$.

5.2 Formulations arcs-sommets

Nous présentons tout d'abord pour chacun des problèmes MCSIPND$_m$ et MCSIPND$_s$ une formulation faisant intervenir des variables de flot sur les arcs du graphe D^1.

5.2.1 Formulations

Notons par $f^{k,e}_{uv}$ le flot du produit k sur l'arc (u, v) de u vers v lorsque l'arête $e \in E^2$ est en panne (i.e. lorsque les arcs de $\overline{F}_e$ sont supprimés dans D^1). Soit $x \in \mathbb{R}^{2|E^1|}$ un vecteur tel que x^l_{uv} représente le nombre de liaisons entre u et v de capacité μ^l, $l = 1, 2$

appartenant à la solution. Soit

$$b_k^v = \begin{cases} -\omega_k & \text{si } v = o_k, \\ 0 & \text{si } v \neq o_k, d_k, \\ \omega_k & \text{si } v = d_k, \end{cases} \qquad \text{pour tout } v \in V^1, \text{ pour tout } k \in K.$$

Le problème MCSIPND$_m$ est équivalent au programme linéaire mixte suivant :

$$\text{Minimiser} \sum_{l=1,2} \sum_{uv \in E^1} c_{uv}^l x_{uv}^l$$

$$\sum_{u:(u,v)\in A^1 \setminus \overline{F}_e} f_{uv}^{k,e} - \sum_{u:(v,u)\in A^1 \setminus \overline{F}_e} f_{vu}^{k,e} = b_k^v \quad \forall v \in V^1, \forall k \in K, \forall e \in E^2, \tag{5.1}$$

$$\sum_{k \in K} f_{uv}^{k,e} \leq \sum_{l=1,2} \mu^l x_{uv}^l \qquad \forall uv \in E^1, \forall e \in E^2, \tag{5.2}$$

$$\sum_{k \in K} f_{vu}^{k,e} \leq \sum_{l=1,2} \mu^l x_{uv}^l \qquad \forall uv \in E^1, \forall e \in E^2, \tag{5.3}$$

$$x_{uv}^l \geq 0 \text{ et entier} \qquad \forall uv \in E^1,\ l = 1, 2, \tag{5.4}$$

$$f_{uv}^{k,e}, f_{vu}^{k,e} \geq 0 \qquad \forall uv \in E^1, \forall k \in K, \forall e \in E^2. \tag{5.5}$$

Les contraintes (5.1) sont appelées les *contraintes de conservation de flot*. Les contraintes (5.2) et (5.3) sont les *contraintes de capacité* : pour chacun des deux sens, la somme des flots pour toutes les demandes $k \in K$ sur une arête doit être inférieure à la capacité de l'arête. Les contraintes (5.4) et (5.5) sont les *contraintes de non négativité* et les *contraintes d'intégrité*.

En ajoutant, les contraintes

$$x_{uv}^1 + x_{uv}^2 \leq 1 \quad \forall uv \in E^1, \tag{5.6}$$

et en remplaçant les contraintes (5.4) par

$$x_{uv}^l \in \{0, 1\} \quad \forall uv \in E^1,\ l = 1, 2, \tag{5.7}$$

on obtient une formulation pour le problème MCSIPND$_s$.

Les contraintes (5.6) imposent que l'on ne peut utiliser qu'une seule liaison entre deux sommets donnés, donc qu'un seul des deux types de liaisons sur une même arête et les contraintes (5.7) imposent à x de prendre ses valeurs dans {0,1}. On a alors $x_{uv}^l = 1$ si l'arête uv de capacité μ^l, $l = 1, 2$ appartient à la solution et 0 sinon.

La formulation ci-dessus peut être renforcée en ajoutant les contraintes dites de capacité résiduelle que l'on présente dans la suite.

5.2.2 Contraintes de capacité résiduelle

Les contraintes de capacité résiduelle ont été introduites par Magnanti, Mirchandani et Vachani [60] lors de leur étude sur le problème NLP (voir page 19). Ces contraintes ont été introduites dans le cadre du sous-problème du problème NLP consistant à ne considérer qu'un seul arc. Ils ont également étendu ces contraintes au problème TFLP (voir page 20). Dans ce qui suit, nous présentons une adaptation de ces contraintes au problème MCSIPND.

Théorème 5.1 *Soit $L \subseteq K$. On note*

$$Q_L = \frac{\sum_{k \in L} \omega_k}{2.5},\ \mu_L = \left\lceil \frac{Q_L}{4} \right\rceil \ \textit{et}$$

$$s_L = \begin{cases} 4 \textit{ si } Q_L \textit{ mod } 4 = 0, \\ Q_L \textit{ mod } 4 \textit{ sinon.} \end{cases}$$

On a alors $Q_L = 4(\mu_L - 1) + s_L$.
Soient $u, v \in V^1$ et $e \in E^2$, l'inégalité

$$\frac{1}{2.5} \sum_{k \in L} (f_{uv}^{k,e} + f_{vu}^{k,e}) - 2x_{uv}^1 - s_L \times 2x_{uv}^2 \leq (\mu_L - 1)(4 - s_L) \tag{5.8}$$

est valide pour le polytope associé au problème MCSIPND.

Preuve. La contrainte (5.8) peut également s'écrire

$$\frac{1}{2.5} \sum_{k \in L} (f_{uv}^{k,e} + f_{vu}^{k,e}) \leq Q_L - s_L(\mu_L - 2x_{uv}^2) + 2x_{uv}^1$$

pour $L \subseteq K$, $u, v \in V^1$ et $e \in E^2$ car $(\mu_L - 1)(4 - s_L) = Q_L - \mu_L s_L$.

- Si $2x_{uv}^2 \geq \mu_L$ alors $Q_L - s_L(\mu_L - 2x_{uv}^2) + 2x_{uv}^1 \geq Q_L$.
  On sait que les contraintes $f_{uv}^{k,e} + f_{vu}^{k,e} \leq \omega_k$ pour tout $k \in L$ sont valides pour le problème. En additionnant ces contraintes, on obtient

  $$\sum_{k \in L} (f_{uv}^{k,e} + f_{vu}^{k,e}) \leq \sum_{k \in L} \omega_k,$$

  donc

  $$\frac{1}{2.5} \sum_{k \in L} (f_{uv}^{k,e} + f_{vu}^{k,e}) \leq Q_L,$$

  ce qui implique la validité de la contrainte (5.8).

– Si $2x^2_{uv} \leq \mu_L - 1$, alors
$Q_L - s_L(\mu_L - 2x^2_{uv}) + 2x^1_{uv}$
$= 4(\mu_L - 1) + s_L - s_L(\mu_L - 2x^2_{uv}) + 2x^1_{uv}$
$= 4(\mu_L - 1) + s_L(2x^2_{uv} - (\mu_L - 1)) + 2x^1_{uv}$.
En posant $s_L = 4 - t$ avec $0 \leq t < 4$, on obtient
$Q_L - s_L(\mu_L - 2x^2_{uv}) + 2x^1_{uv}$
$= 4(\mu_L - 1) + (4 - t)(2x^2_{uv} - (\mu_L - 1)) + 2x^1_{uv}$
$= 8x^2_{uv} + 2x^1_{uv} + t(-2x^2_{uv} + (\mu_L - 1))$.
Ainsi

$$Q_L - s_L(\mu_L - 2x^2_{uv}) + 2x^1_{uv} \geq 8x^2_{uv} + 2x^1_{uv}, \tag{5.9}$$

car $t \geq 0$ et $2x^2_{uv} \leq \mu_L - 1$. Or

$$\sum_{k \in K} f^{k,e}_{uv} \leq 2.5x^1_{uv} + 10x^2_{uv} \text{ et } \sum_{k \in K} f^{k,e}_{vu} \leq 2.5x^1_{uv} + 10x^2_{uv},$$

donc $\frac{1}{2.5}\sum_{k \in K}(f^{k,e}_{uv} + f^{k,e}_{vu}) \leq 2x^1_{uv} + 8x^2_{uv}$ et par conséquent

$$\frac{1}{2.5}\sum_{k \in L}(f^{k,e}_{uv} + f^{k,e}_{vu}) \leq 2x^1_{uv} + 8x^2_{uv}. \tag{5.10}$$

D'après (5.9) et (5.10), il s'ensuit que la contrainte (5.8) est valide.
La contrainte (5.8) est donc valide pour le polytope associé au problème. □

Les inégalités (5.8) seront appelées *contraintes de capacité résiduelle.*

5.3 Formulations arcs-chemins et génération de colonnes

Nous donnons maintenant pour chacun des deux problèmes une formulation basée sur un nombre non-polynomial de variables. Cette formulation sera à la base d'un algorithme de génération de colonnes.

5.3.1 Formulations

De façon similaire à la formulation arcs-sommets, on définit le vecteur $x \in I\!R^{2|E^1|}$ tel que x^l_{uv} représente le nombre de liaisons entre u et v de capacité μ^l, $l = 1, 2$

appartenant à la solution. Soient e une arête de E^2 et k une demande, on note $\mathcal{P}_k^e$ l'ensemble des chemins allant de o_k vers d_k dans le graphe $D^1 \backslash \overline{F}_e$ (c'est-à-dire lorsque l'arête $e \in E^2$ tombe en panne). Pour un chemin P de $\mathcal{P}_k^e$, la variable $y_k^e(P)$ représente la quantité de flot de la demande k transitant sur le chemin P lors d'une panne de e. Soit $\tau_{uv}(P) = \begin{cases} 1 & \text{si l'arc } (u,v) \text{ appartient à } P, \\ 0 & \text{sinon,} \end{cases}$ pour tout $u, v \in V$. Alors pour tout $u, v \in V$, $k \in K$, et $e \in E^2$, on a

$$f_{uv}^{k,e} = \sum_{P \in \mathcal{P}_k^e} \tau_{uv}(P) y_k^e(P) = \sum_{P \in \mathcal{P}_k^e \mid (u,v) \in P} y_k^e(P).$$

Le problème MCSIPND$_m$ est donc équivalent au programme linéaire mixte suivant :

$$P_m : \text{Minimiser} \sum_{l=1,2} \sum_{uv \in E^1} c_{uv}^l x_{uv}^l$$

$$\sum_{P \in \mathcal{P}_k^e} y_k^e(P) = \omega_k \qquad \forall k \in K, \forall e \in E^2, \tag{5.11}$$

$$\sum_{k \in K} \sum_{P \in \mathcal{P}_k^e \mid (u,v) \in P} y_k^e(P) \leq \sum_{l=1,2} \mu^l x_{uv}^l \qquad \forall uv \in E^1, \forall e \in E^2, \tag{5.12}$$

$$\sum_{k \in K} \sum_{P \in \mathcal{P}_k^e \mid (v,u) \in P} y_k^e(P) \leq \sum_{l=1,2} \mu^l x_{uv}^l \qquad \forall uv \in E^1, \forall e \in E^2, \tag{5.13}$$

$$y_k^e(P) \geq 0 \qquad \forall e \in E^2, \forall k \in K, \forall P \in \mathcal{P}_k^e, \tag{5.14}$$

$$x_{uv}^l \geq 0 \text{ et entier} \qquad \forall uv \in E^1, l = 1, 2. \tag{5.15}$$

Les contraintes (5.11), appelées *contraintes de demande*, imposent au flot de permettre l'écoulement de la demande. En effet, si l'on considère la panne de l'arête $e \in E^2$ et une demande $k \in K$, $\sum_{P \in \mathcal{P}_k^e} y_k^e(P)$ représente la quantité totale de flot de la demande k passant sur l'ensemble des chemins allant de o_k vers d_k. Ce flot doit être égal à la demande ω_k entre o_k et d_k. Les contraintes (5.12) et (5.13) sont appelées les *contraintes de capacité*. Le flot transitant sur l'arête uv ne doit pas excéder la capacité de l'arête et ce dans le sens u vers v (contraintes (5.12)) comme dans le sens v vers u (contraintes (5.13)). On considère pour cela tous les chemins empruntant l'arc (u,v) et on vérifie que la somme des flots ne dépasse pas $\sum \mu^l x_{uv}^l$ qui représente la quantité totale de capacités installées sur l'arête $uv \in E^1$. Les contraintes (5.14) et (5.15) sont les *contraintes de non négativité* et les *contraintes d'intégrité*.

En ajoutant les contraintes

$$x_{uv}^1 + x_{uv}^2 \leq 1 \quad \forall uv \in E^1, \tag{5.16}$$

et en remplaçant les contraintes (5.15) par

$$x^l_{uv} \in \{0,1\} \qquad \forall uv \in E^1,\, l = 1,2. \tag{5.17}$$

on obtient une formulation pour le problème MCSIPND$_s$. On note P_s ce nouveau programme linéaire mixte.

Malheureusement, si la formulation arcs-chemins contient relativement peu de contraintes, elle contient un nombre exponentiel de variables. Une méthode de résolution appropriée pour ce type de formulation est la génération de colonnes. Dans la section suivante nous décrivons une telle approche dans le cas du problème entier puis dans le cas binaire.

5.3.2 Génération de colonnes

La technique de génération de colonnes permet d'utiliser un nombre réduit de variables. En effet, cette approche consiste à résoudre itérativement des programmes linéaires ne contenant qu'une partie des variables. A chaque itération un problème auxiliaire détermine les variables devant être rajoutées au programme linéaire courant. Lorsque le problème auxiliaire peut être résolu de manière exacte, et que celui-ci ne trouve pas de variables à rajouter, la solution optimale du programme linéaire courant est la solution optimale du programme linéaire avec toutes les variables (voir section 1.1.4).

Les approches par génération de colonnes pour les deux versions du problème ne sont pas identiques, en particulier en ce qui concerne la procédure de construction d'une solution réalisable de départ. Nous présentons tout d'abord cette procédure dans le cas entier puis dans le cas binaire. Pour ce dernier cas, la détermination d'une solution réalisable de départ se ramène à la résolution d'un programme linéaire par génération de colonnes. Nous décrivons par la suite, le problème auxiliaire qui est le même pour les deux versions du problème. Cette approche sera utilisée dans le chapitre 6 afin de développer un algorithme coupes, génération de colonnes et branchements pour le problème.

5.3.2.1 Solution initiale pour le problème MCSIPND$_m$

Proposition 5.2 *Il existe une solution réalisable pour le problème MCSIPND$_m$ si et seulement si pour toute panne $e \in E^2$, et pour toute demande $k \in K$, o_k et d_k appartiennent à la même composante connexe du graphe $G^1 \backslash F_e$.*

Preuve. Soit F une solution du problème MCSIPND$_m$. Alors, pour toute panne $e \in E^2$, et pour toute demande $k \in K$, on sait qu'il existe au moins un chemin dans $G^1 \backslash F_e$ permettant d'acheminer un flot de o_k vers d_k. Donc o_k et d_k sont dans la même composante connexe de $G^1 \backslash F_e$.
Soient $t^e_{uv} = 0$ pour tout $(u,v) \in A^1$ et $e \in E^2$. Supposons que pour $k \in K$ et $e \in E^2$, o_k et d_k sont dans la même composante connexe de $G^1 \backslash F_e$. Il existe donc un chemin de o_k vers d_k dans $G^1 \backslash F_e$. On ajoute $\lceil \frac{\omega_k}{2.5} \rceil$ à t^e_{uv} pour toutes les arêtes (u,v) composant ce chemin. Soient $T_{uv} = max\{t^e_{uv}, t^e_{vu}, e \in E^2\}$ pour $uv \in E^1$. On construit alors une solution réalisable en considérant dans la solution toutes les arêtes uv de E^1 telles que $T_{uv}>0$ avec une capacité T_{uv}. □

Lorsqu'il existe une solution réalisable du problème MCSIPND$_m$, il est donc facile de trouver un ensemble de chemins tels que le programme P_m réduit aux variables correspondant à ces chemins ait une solution. Pour chaque panne $e \in E^2$, et pour toute demande $k \in K$, on cherche un plus court chemin reliant o_k et d_k dans le graphe $G^1 \backslash F_e$. Les coûts étant strictement positifs, ce problème est polynomial. Ces chemins permettent alors de trouver une solution réalisable de départ.

5.3.2.2 Solution initiale pour le problème MCSIPND$_s$

Malheureusement, une solution réalisable pour le problème MCSIPND$_m$ n'est pas toujours réalisable pour le problème MCSIPND$_s$. Même lorsque les conditions de connexités sont vérifiées, le problème MCSIPND$_s$ peut ne pas avoir de solution. En effet, on ne peut pas installer autant de capacité que l'on veut sur les arêtes et les capacités que l'on peut installer peuvent être insuffisantes.

Afin de déterminer une base réalisable pour le problème, on considère le problème suivant :

S_s : Minimiser ε

$$\sum_{P \in \mathcal{P}^e_k} y^e_k(P) = \omega_k \qquad \forall k \in K,\ \forall e \in E^2,$$

$$\sum_{k \in K} \sum_{P \in \mathcal{P}^e_k \,|\, (u,v) \in P} y^e_k(P) \leq \mu^2 + \varepsilon \qquad \forall (u,v) \in A^1 \backslash \overline{F}_e,\ \forall e \in E^2,$$

$$y^e_k(P) \geq 0 \qquad \forall e \in E^2,\ \forall k \in K, \forall P \in \mathcal{P}^e_k,$$

$$\varepsilon \geq 0.$$

Le programme S_s est obtenu à partir du programme P_s en fixant les variables x_{uv}^l à 0 si $l = 1$ et à 1 si $l = 2$. Cela revient à prendre un graphe complet avec la plus grande capacité (10 Gbit/s) installée sur chaque arête. Les contraintes (5.16) et (5.17) sont bien respectées. De plus, la variable ε représente une augmentation possible de la capacité des arêtes.

Pour résoudre ce problème, nous utilisons une méthode de génération de colonnes. La valeur optimale de ε représente la capacité manquante. Si $\varepsilon = 0$ alors le problème maître admet une solution réalisable et l'ensemble des chemins dont les variables suffisent à obtenir $\varepsilon = 0$ assure une solution réalisable de départ au problème maître. Si ce n'est pas le cas, le problème MCSIPND$_s$ n'admet aucune solution réalisable.

Notons que le programme S_s peut être réécrit de la manière suivante :

S'_s : Minimiser ε

$$\sum_{P \in \mathcal{P}_k^e} y_k^e(P) = \omega_k \qquad \forall k \in K,\ \forall e \in E^2, \tag{5.18}$$

$$-\sum_{k \in K} \sum_{P \in \mathcal{P}_k^e \mid (u,v) \in P} y_k^e(P) + \varepsilon \geq -\mu^2 \qquad \forall (u,v) \in A^1 \backslash \overline{F}_e,\ \forall e \in E^2, \tag{5.19}$$

$$y_k^e(P) \geq 0 \qquad \forall e \in E^2,\ \forall k \in K, \forall P \in \mathcal{P}_k^e,$$

$$\varepsilon \geq 0.$$

Une méthode de résolution pour le programme S'_s peut être décrite comme suit :

1) On choisit un ensemble de chemins de départ J comme étant les plus courts chemins entre l'origine et la destination de toutes les demandes dans les graphes $G^1 \backslash F_e$ pour toute panne $e \in E^2$.
2) Soit x^* la solution optimale de la relaxation linéaire du programme restreint aux chemins de J. On détermine les valeurs des variables duales α_k^e associée aux contraintes (5.18) et $\beta_{(u,v)}^e$ associées aux contraintes (5.19).
3) Si $r_P^{k,e} = \sum_{(u,v) \in P} \beta_{(u,v)}^e - \alpha_k^e \geq 0$ pour tout chemin $P \in \mathcal{P}_k^e \backslash J$, $k \in K$ et $e \in E^2$, alors une solution optimale est trouvée et l'algorithme se termine.
4) Sinon, ajouter dans J des variables associées aux chemins P tels que $r_P^{k,e} < 0$.
5) Aller en 2).

L'étape 3) est appelée *problème auxiliaire (pricing problem)* et consiste donc ici en des problèmes de plus court chemin dans les graphes $G^1 \backslash F_e$ avec le coût $\beta_{(u,v)}^e$ sur les

arêtes, pour toute panne $e \in E^2$. On remarque que les coûts étant positifs, le problème du plus court chemin est polynomial.

Si la solution optimale trouvée donne $\varepsilon = 0$ alors il existe une solution réalisable pour le problème (P) de départ et celle-ci est donnée par la solution optimale du problème (S_s). Si ce n'est pas le cas $(\varepsilon > 0)$ alors il n'y a pas de solution réalisable au problème et on a terminé.

5.3.2.3 Résolution du problème auxiliaire

Les problèmes auxiliaires utilisés dans les approches par génération de colonnes pour les problèmes MCSIPND$_m$ et MCSIPND$_s$ sont identiques et similaires à celui utilisé pour la résolution du programme S_s. On le présente ici pour le problème MCSIPND$_m$.

On choisit un ensemble de chemins de départ $\overline{J}$ comme étant les chemins représentant la solution réalisable précédemment décrite. Soit $\overline{x}$ la solution optimale de la relaxation linéaire du programme P_m restreint aux chemins de $\overline{J}$. On peut réécrire le programme P_m de la manière suivante :

$$\text{Minimiser} \sum_{l=1,2} \sum_{uv \in E^1} c^l_{uv} x^l_{uv}$$

$$\sum_{P \in \mathcal{P}^e_k} y^e_k(P) = \omega_k \qquad \forall k \in K, \forall e \in E^2, \tag{5.20}$$

$$-\sum_{k \in K} \sum_{P \in \mathcal{P}^e_k \,|\, (u,v) \in P} y^e_k(P) + \sum_{l=1,2} \mu^l x^l_{uv} \geq 0 \qquad \forall uv \in E^1, \forall e \in E^2, \tag{5.21}$$

$$-\sum_{k \in K} \sum_{P \in \mathcal{P}^e_k \,|\, (v,u) \in P} y^e_k(P) + \sum_{l=1,2} \mu^l x^l_{uv} \geq 0 \qquad \forall uv \in E^1, \forall e \in E^2, \tag{5.22}$$

$$y^e_k(P) \geq 0 \qquad \forall e \in E^2, \forall k \in K, \forall P \in \mathcal{P}^e_k,$$

$$x^l_{uv} \geq 0 \text{ et entier} \qquad \forall uv \in E^1, l = 1, 2.$$

Soient γ^e_k, $\vartheta^e_{(u,v)}$ et $\vartheta^e_{(v,u)}$ les variables duales associées aux contraintes (5.20), (5.21) et (5.22) respectivement. Soient $k \in K$ et $e \in E^2$, le coût réduit d'une variable associée au chemin $P \in \mathcal{P}^e_k$ est égal à

$$R^{k,e}_P = \sum_{(u,v) \in P} \vartheta^e_{(u,v)} - \gamma^e_k.$$

Le problème auxiliaire consiste à trouver, pour tout $k \in K$ et $e \in E^2$, un chemin P de $\mathcal{P}^e_k$ tel que $R^{k,e}_P = \min_{P' \in \mathcal{P}^e_k} R^{k,e}_{P'}$ et tel que $R^{k,e}_P < 0$. Il s'agit donc d'un problème de

plus court chemin dans le graphe $D^1 \backslash \overline{F}_e$ quand le coût est donné par $\vartheta^e_{(u,v)}$, $(u,v) \in A^1 \backslash \overline{F}_e$, pour chaque $e \in E^2$. Les variables duales $\vartheta^e_{(u,v)}$ étant positives, ce problème est polynomial.

Il est bien établi que pour des problèmes NP-difficiles de grande taille (en nombre de variables) une approche basée sur les techniques de génération de colonnes et les techniques polyédrales est souvent appropriée. Les premières permettent, comme il a été mentionné, de réduire le nombre de variables utilisées en résolvant des relaxations linéaires succéssives. Et les dernières permettent de renforcer la relaxation linéaire en rajoutant des coupes. Cette approche sera utilisée pour résoudre la formulation arcs-chemins du problème. Pour la formulation arcs-sommets, nous utiliserons une technique de coupes et branchements pour résoudre le problème. Pour mettre en place les algorithmes, une étude polyédrale a été menée pour ces problèmes. Celle-ci est présentée dans la section suivante.

5.4 Polyèdres associés

Dans la suite, on considère un graphe $G = (V, E)$ et le graphe $D = (V, A)$ obtenu à partir de G en remplaçant toute arête de E par deux arcs. On considère également $\mathcal{F} = \{F_1, \ldots, F_t\} \subseteq 2^E$, $t \geq 2$ une famille de sous-ensembles d'arêtes de E et $\overline{\mathcal{F}} = \{\overline{F}_1, \ldots, \overline{F}_t\} \subseteq 2^A$ la famille de sous-ensembles d'arcs associée à $\mathcal{F}$. Soit K un ensemble de demandes. Chaque demande $k \in K$ est définie par son origine o_k, sa destination d_k et son volume ω_k et on pose

$$b^v_k = \begin{cases} -\omega_k & \text{si } v = o_k \\ 0 & \text{si } v \neq o_k, d_k \\ \omega_k & \text{si } v = d_k \end{cases} \quad \forall v \in V.$$

Pour tout $i \in \{1, \ldots, t\}$, nous noterons par $G_i = (V, E_i)$ (resp. $D_i(V, A_i)$) le sous-graphe de G (resp. de D) obtenu en supprimant les arêtes de F_i (resp. $\overline{F}_i$). Dans la suite on considère $\mu^1 = 2.5$ et $\mu^2 = 10$.

Considérons les inégalités suivantes :

$$\sum_{u:(u,v)\in A\backslash\overline{F}_i} f^{k,i}_{uv} - \sum_{u:(v,u)\in A\backslash\overline{F}_i} f^{k,i}_{vu} = b^v_k \qquad \forall v \in V, \forall k \in K,\ i = 1, \ldots, t, \tag{5.23}$$

$$\sum_{k\in K} f^{k,i}_{uv} \leq 2.5x^1_{uv} + 10x^2_{uv} \qquad \forall uv \in E,\ i = 1, \ldots, t, \tag{5.24}$$

$$\sum_{k\in K} f_{vu}^{k,i} \leq 2.5x_{uv}^1 + 10x_{uv}^2 \qquad \forall uv \in E,\ i = 1,\ldots,t, \tag{5.25}$$

$$x_{uv}^l \geq 0 \qquad \forall uv \in E,\ l = 1,2, \tag{5.26}$$

$$f_{uv}^{k,i}, f_{vu}^{k,i} \geq 0 \qquad \forall uv \in E, \forall k \in K, i = 1,\ldots,t, \tag{5.27}$$

$$x_{uv}^1 + x_{uv}^2 \leq 1 \qquad \forall uv \in E. \tag{5.28}$$

Soient

$$\text{MCSIPND}_m^{as}(G,\mathcal{F},K) = \{(x,f) \in \mathcal{D}_m^{as} \mid x \text{ et } f \text{ satisfont (5.23)-(5.27)}\},$$

$$\text{MCSIPND}_s^{as}(G,\mathcal{F},K) = \{(x,f) \in \mathcal{D}_s^{as} \mid x \text{ et } f \text{ satisfont (5.23)-(5.28)}\}$$

avec

$$\mathcal{D}_m^{as} = \{x \in I\!N^{2|E|}, f \in I\!R^{2|E|*|K|*|\mathcal{F}|}\}$$

et

$$\mathcal{D}_s^{as} = \{x \in \{0,1\}^{2|E|}, f \in I\!R^{2|E|*|K|*|\mathcal{F}|}\}.$$

Si $G = G^1$ et $\mathcal{F} = \{F_e,\ e \in E^2\}$, alors $\text{MCSIPND}_m^{as}(G,\mathcal{F},K)$ (resp. $\text{MCSIPND}_s^{as}(G,\mathcal{F},K)$) n'est rien d'autre que le polytope associé au problème MCSIPND_m (resp. MCSIPND_s).

Soit $\mathcal{P}_k^i$ l'ensemble des chemins allant de o_k vers d_k dans le graphe $D\backslash\overline{F}_i$ pour $i \in \{1,\ldots,t\}$. Soit $\text{MCSIPND}_m^{ac}(G,\mathcal{F},K)$ l'enveloppe convexe des solutions entières du système défini par les contraintes :

$$\sum_{P\in\mathcal{P}_k^i} y_k^i(P) = \omega_k \qquad \forall k \in K,\ i = 1,\ldots,t, \tag{5.29}$$

$$\sum_{k\in K} \sum_{P\in\mathcal{P}_k^i \mid (u,v)\in P} y_k^i(P) \leq 2.5x_{uv}^1 + 10x_{uv}^2 \qquad \forall uv \in E,\ i = 1,\ldots,t, \tag{5.30}$$

$$\sum_{k\in K} \sum_{P\in\mathcal{P}_k^i \mid (v,u)\in P} y_k^i(P) \leq 2.5x_{uv}^1 + 10x_{uv}^2 \qquad \forall uv \in E,\ i = 1,\ldots,t, \tag{5.31}$$

$$x_{uv}^l \geq 0 \qquad \forall uv \in E,\ l = 1,2, \tag{5.32}$$

$$y_k^i(P) \geq 0 \qquad \forall k \in K,\ i = 1,\ldots,t,\ \forall P \in \mathcal{P}_k^i. \tag{5.33}$$

En ajoutant les contraintes (5.28) à $\text{MCSIPND}_m^{ac}(G,\mathcal{F},K)$, on obtient $\text{MCSIPND}_s^{ac}(G,\mathcal{F},K)$.

On peut remarquer que si $G = G^1$ et $\mathcal{F} = \{F_e, e \in E^2\}$, MCSIPND$_m^{ac}(G, \mathcal{F}, K)$ (resp. MCSIPND$_s^{ac}(G, \mathcal{F}, K)$) n'est rien d'autre que le polytope associé au problème MCSIPND$_m$ (resp. MCSIPND$_s$).

Par abus de langage, on notera MCSIPND$_m(G, \mathcal{F}, K)$ quand on parlera de l'un des deux polytopes MCSIPND$_m^{as}(G, \mathcal{F}, K)$ ou MCSIPND$_m^{ac}(G, \mathcal{F}, K)$ indifféremment. Le polytope MCSIPND$_m(G, \mathcal{F}, K)$ sera alors associé au problème MCSIPND$_m$. Il en est de même pour le polytope MCSIPND$_s(G, \mathcal{F}, K)$ qui sera associé au problème MCSIPND$_s$, et qui représentera indiféremment les polytopes MCSIPND$_s^{as}(G, \mathcal{F}, K)$ et MCSIPND$_s^{ac}(G, \mathcal{F}, K)$. Les polytopes MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$ seront appelés les *polytopes des réseaux multicouches fiables avec capacités* (*Multilayer Capacitated Survivable IP Network Design Polytopes*).

Le théorème suivant donne la dimension du polytope MCSIPND$_m^{as}(G, \mathcal{F}, K)$.

Théorème 5.3 $dim(MCSIPND_m^{as}(G, \mathcal{F}, K)) = 2|E| + 2|\mathcal{F}| \times |K| - (|N| - 1)|\mathcal{F}| \times |K|$.

Preuve. La formulation arcs-sommets du problème MCSIPND$_m$ contient $2|E| + 2|\mathcal{F}| \times |E| \times |K|$ variables et $(|N| - 1)|\mathcal{F}| \times |K|$ contraintes égalités non redondantes. Donc $dim(MCSIPND_m^{as}(G, \mathcal{F}, K)) \leq 2|E| + 2|\mathcal{F}| \times |E| \times |K| - (|N| - 1)|\mathcal{F}| \times |K|$. La preuve de $dim(MCSIPND_m^{as}(G, \mathcal{F}, K)) \geq 2|E| + 2|\mathcal{F}| \times |E| \times |K| - (|N| - 1)|\mathcal{F}| \times |K|$ découle de la preuve du théorème 5.12. □

5.5 Contraintes valides

Dans la suite, nous introduisons plusieurs classes d'inégalités valides pour les polytopes MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$. Nous donnons des conditions nécessaires et des conditions suffisantes pour que ces inégalités définissent des facettes de MCSIPND$_m^{as}(G, \mathcal{F}, K)$.

Soit $W \subseteq V$, on note par $\gamma^+(W)$ (resp. $\gamma^-(W)$) l'ensemble des demandes ayant leur origine (resp. destination) dans W et leur destination (resp. origine) dans $V \backslash W$. Et on désigne par $\gamma(W)$ l'ensemble $\gamma^+(W) \cup \gamma^-(W)$.

5.5.1 Contraintes de coupe

5.5.1.1 Contraintes de coupe topologiques

Proposition 5.4 *Soient $F_i \in \mathcal{F}$ et $W \subseteq V$, $\emptyset \neq W \neq V$ tel que $\gamma(W) \neq \emptyset$, l'inégalité*

$$x^1(\delta_{G_i}(W)) + x^2(\delta_{G_i}(W)) \geq 1 \tag{5.34}$$

est valide pour $MCSIPND_m(G, \mathcal{F}, K)$ et $MCSIPND_s(G, \mathcal{F}, K)$.

Preuve. Triviale. □

Les inégalités (5.34) sont appelées les *contraintes de coupe topologique*. Elles imposent la connexité entre les origines et les destinations de chaque demande.

Proposition 5.5 *Soient $F_i \in \mathcal{F}$ et $W \subseteq V$, $\emptyset \neq W \neq V$. Les contraintes*

$$x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) \geq max \left\{ \frac{\sum_{k\in\gamma^+(W)} \omega_k}{2.5}, \frac{\sum_{k\in\gamma^-(W)} \omega_k}{2.5} \right\} \tag{5.35}$$

sont redondantes par rapport aux contraintes (5.23)-(5.27).

Preuve. D'après les contraintes (5.23), on sait que

$$\sum_{uv\in\delta_{G_i}(W)\,|\,u\in W, v\in\overline{W}} \sum_{k\in\gamma^+(W)} f_{uv}^{k,i} \;\geq\; \sum_{k\in\gamma^+(W)} \omega_k,$$

et d'après les contraintes (5.24),

$$\sum_{uv\in\delta_{G_i}(W)\,|\,u\in W, v\in\overline{W}} \sum_{k\in\gamma^+(W)} f_{uv}^{k,i} \;\leq\; \sum_{uv\in\delta_{G_i}(W)} (2.5x_{uv}^1 + 10x_{uv}^2).$$

Donc

$$\sum_{uv\in\delta_{G_i}(W)} (x_{uv}^1 + 4x_{uv}^2) \geq \frac{\sum_{k\in\gamma^+(W)} \omega_k}{2.5}.$$

On montre de la même façon que

$$\sum_{uv\in\delta_{G_i}(W)} (x_{uv}^1 + 4x_{uv}^2) \geq \frac{\sum_{k\in\gamma^-(W)} \omega_k}{2.5}.$$

□

Dans la suite, nous présentons une nouvelle classe de contraintes basées sur les contraintes (5.35).

5.5.1.2 Contraintes de coupe avec capacités

Soit $W \subseteq V$, on note $D_W = \left\lceil \max \left\{ \frac{\sum_{k\in\gamma^+(W)} \omega_k}{2.5}, \frac{\sum_{k\in\gamma^-(W)} \omega_k}{2.5} \right\} \right\rceil$. D_W représente une borne supérieure de la valeur maximum du volume des demandes passant à travers la coupe $\delta(W)$ de W vers $V\backslash W$ et de $V\backslash W$ vers W, divisée par 2.5.

Soient $F_i \in \mathcal{F}$ et $W \subseteq V$, $\emptyset \neq W \neq V$. Considérons l'inégalité

$$x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) \geq D_W. \tag{5.36}$$

Théorème 5.6 *L'inégalité (5.36) est valide pour MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$.*

Preuve. Le volume agrégé des capacités installées sur la coupe doit être supérieur ou égal à la demande devant circuler à travers cette même coupe de W vers $V\backslash W$ et de $V\backslash W$ vers W. Donc

$$2.5x^1(\delta_{G_i}(W)) + 10x^2(\delta_{G_i}(W)) \geq \text{Max} \left\{ \sum_{k\in\gamma^+(W)} \omega_k, \sum_{k\in\gamma^-(W)} \omega_k \right\}.$$

En divisant cette inégalité par 2.5 et en arrondissant le membre de droite à l'entier supérieur, on obtient l'inégalité (5.36). □

Dans la suite, nous présentons de nouvelles contraintes valides obtenues en combinant les contraintes (5.36) et les contraintes triviales par des procédures de Chvátal-Gomory.

Lemme 5.7 *Soient $F_i \in \mathcal{F}$ et $W \subseteq V$, $\emptyset \neq W \neq V$. Alors l'inégalité*

$$x^1(\delta_{G_i}(W)) + 2x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{D_W}{2} \right\rceil \tag{5.37}$$

est valide pour MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$.

Preuve. Les inégalités suivantes sont valides pour MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$.

$$\begin{aligned} x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) &\geq D_W, \\ x^1_{uv} &\geq 0 \qquad \text{pour tout } uv \in \delta_{G_i}(W). \end{aligned}$$

En sommant ces inégalités, on obtient l'inégalité

$$2x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) \geq D_W.$$

En divisant par 2 et en arrondissant le membre de droite à l'entier supérieur, on obtient l'inégalité (5.37). □

Théorème 5.8 *Soient $F_i \in \mathcal{F}$ et $W \subseteq V$, $\emptyset \neq W \neq V$. Alors l'inégalité*

$$x^1(\delta_{G_i}(W)) + x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{D_W}{4} \right\rceil \tag{5.38}$$

est valide pour MCSIPND$_m$$(G, \mathcal{F}, K)$ et MCSIPND$_s$$(G, \mathcal{F}, K)$.

Preuve. En ajoutant les inégalités

$$x^1_{uv} \geq 0 \qquad \text{pour tout } uv \in \delta_{G_i}(W),$$

à l'inégalité (5.37), on obtient

$$2x^1(\delta_{G_i}(W)) + 2x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{D_W}{2} \right\rceil.$$

En divisant cette inégalité par 2 et en arrondissant le membre de droite à l'entier supérieur, on obtient

$$x^1(\delta_{G_i}(W)) + x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{\left\lceil \frac{D_W}{2} \right\rceil}{2} \right\rceil.$$

Comme $\left\lceil \frac{\left\lceil \frac{D_W}{2} \right\rceil}{2} \right\rceil = \left\lceil \frac{D_W}{4} \right\rceil$, on obtient l'inégalité (5.38). □

Théorème 5.9 *Soient $F_i \in \mathcal{F}$ et $W \subseteq V$, $\emptyset \neq W \neq V$. Alors l'inégalité*

$$x^1(\delta_{G_i}(W)) + 3x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{3D_W}{4} \right\rceil \tag{5.39}$$

est valide pour MCSIPND$_m$$(G, \mathcal{F}, K)$ et MCSIPND$_s$$(G, \mathcal{F}, K)$.

Preuve. Les inégalités suivantes sont valides pour MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$.

$$\begin{aligned} x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) &\geq D_W, \\ x^1(\delta_{G_i}(W)) + 2x^2(\delta_{G_i}(W)) &\geq \left\lceil \frac{D_W}{2} \right\rceil. \end{aligned}$$

En sommant ces inégalités, on obtient

$$2x^1(\delta_{G_i}(W)) + 6x^2(\delta_{G_i}(W)) \geq D_W + \left\lceil \frac{D_W}{2} \right\rceil.$$

En divisant cette inégalité par 2 et en arrondissant le membre de droite à l'entier supérieur, on obtient

$$x^1(\delta_{G_i}(W)) + 3x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{\lceil \frac{3D_W}{2} \rceil}{2} \right\rceil .$$

Comme $\left\lceil \frac{\lceil \frac{3D_W}{2} \rceil}{2} \right\rceil = \lceil \frac{3D_W}{4} \rceil$, on obtient l'inégalité (5.39). □

Théorème 5.10 *Soient $F_i \in \mathcal{F}$ et $W \subseteq V$, $\emptyset \neq W \neq V$. Alors l'inégalité*

$$x^1(\delta_{G_i}(W)) + 2x^2(\delta_{G_i}(W)) \geq \begin{cases} \lceil \frac{D_W}{2} \rceil + 1 & \textit{si } D_W \textit{ mod } 4 = 2, \\ \lceil \frac{D_W}{2} \rceil & \textit{sinon}, \end{cases} \tag{5.40}$$

est valide pour MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$.

Preuve. Les inégalités suivantes sont valides pour MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$.

$$\begin{aligned} x^1(\delta_{G_i}(W)) + x^2(\delta_{G_i}(W)) &\geq \left\lceil \frac{D_W}{4} \right\rceil , \\ x^1(\delta_{G_i}(W)) + 3x^2(\delta_{G_i}(W)) &\geq \left\lceil \frac{3D_W}{4} \right\rceil . \end{aligned}$$

En sommant ces inégalités, on obtient

$$2x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{D_W}{4} \right\rceil + \left\lceil \frac{3D_W}{4} \right\rceil .$$

En divisant cette inégalité par 2 et en arrondissant le membre de droite à l'entier supérieur, on obtient

$$x^1(\delta_{G_i}(W)) + 2x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{\lceil \frac{3D_W}{4} \rceil + \lceil \frac{D_W}{4} \rceil}{2} \right\rceil .$$

Comme $\left\lceil \frac{\lceil \frac{3D_W}{4} \rceil + \lceil \frac{D_W}{4} \rceil}{2} \right\rceil = \begin{cases} \lceil \frac{D_W}{2} \rceil + 1 & \text{si } D_W \bmod 4 = 2, \\ \lceil \frac{D_W}{2} \rceil & \text{sinon}, \end{cases}$
on obtient l'inégalité (5.40). □

Les inégalités (5.36), (5.38), (5.39), (5.40) seront appelées *contraintes de coupe avec capacités.*

On pourrait générer d'autres contraintes valides basées sur les coupes en considérant les inégalités de type (5.36), (5.38), (5.39), (5.40) et les contraintes triviales. Néanmoins, comme nous allons le montrer ci-après, toutes les inégalités obtenues de cette façon sont redondantes par rapport à ces contraintes. Pour montrer cela, on construit le graphe de la figure 5.12 dont on décrit la construction dans la suite.

Les nœuds du graphe représentent des contraintes valides définies sur la coupe $\delta_{G_i}(W)$. Chaque nœud est défini par un triplet (a, b, c) qui représente une contrainte de la forme $\sum_{e\in\delta_{G_i}(W)}(ax^1(e)+bx^2(e)) \geq c$. La construction du graphe commence par un premier niveau comportant trois nœuds correspondant à la contrainte $\sum_{e\in\delta_{G_i}(W)}(x^1(e)+4x^2(e)) \geq D_W$, soit avec notre formalisme $(1, 4, D_W)$, la contrainte $\sum_{e\in\delta_{G_i}(W)} x^1(e) \geq 0$ $((1, 0, 0))$ et la contrainte $\sum_{e\in\delta_{G_i}(W)} x^2(e) \geq 0$ $((0, 1, 0))$. Ces dernières sont obtenues en additionnant respectivement les contraintes $x^1(e) \geq 0$ pour tout $e \in \delta_{G_i}(W)$ et les contraintes $x^2(e) \geq 0$ pour tout $e \in \delta_{G_i}(W)$. Ce premier niveau va servir pour créer les autres nœuds du graphe. Plus concrètement, le graphe va comporter un certain nombre de niveaux. Les sommets d'un certain niveau sont créés à partir des nœuds des niveaux supérieurs.

La création d'un nœud se fait à partir de deux nœuds des niveaux supérieurs en combinant les deux contraintes associées par une procédure de Chvatal-Gomory. L'ajout de contraintes du type $x^1(e) \geq 0$ ou $x^2(e) \geq 0$ pour $e \in E^1$ est effectué si nécessaire.

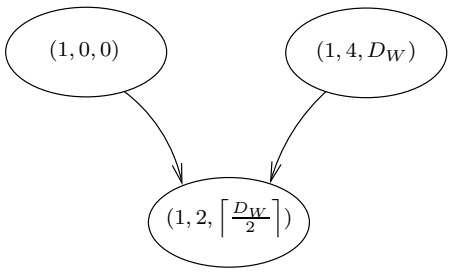


FIG. 5.11 – Création d'une nouvelle contrainte

Par exemple, la figure 5.11 représente la création de la contrainte $(1, 2, \lceil\frac{D_W}{2}\rceil)$ à partir des contraintes $(1, 0, 0)$ et $(1, 4, D_W)$ du premier niveau. La contrainte $(1, 0, 0)$ est elle-même formée à partir des contraintes triviales. En effet, en ajoutant les contraintes

$$\begin{aligned} x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) &\geq D_W, \\ x^1(e) &\geq 0, \quad \forall e \in \delta_{G_i}(W), \end{aligned}$$

on obtient l'inégalité

$$2x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) \geq D_W.$$

En divisant par 2 et en arrondissant le membre de droite à l'entier supérieur, on obtient l'inégalité

$$x^1(\delta_{G_i}(W)) + 2x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{D_W}{2} \right\rceil .$$

Lors de la création d'un nœud, la contrainte obtenue peut être dominée par une autre contrainte. Dans ce cas, le noeud sera considéré comme inactif. De même, la contrainte créée peut dominer d'autres contraintes, rendant les noeuds correspondants inactifs.

Le graphe de la figure 5.12 est construit de la manière suivante :
- pour chaque nœud courant, on crée des nœuds descendant du nœud courant et de chacun des nœuds actifs des niveaux supérieurs (on considère pour cela les nœuds de haut en bas et de gauche à droite),
- à la création d'un nœud, on vérifie s'il est ou non dominé. On le rend inactif s'il est dominé,
- on vérifie également s'il domine un nœud actif du graphe. Si un tel nœud existe, on le rend inactif.

Les nœuds inactifs ont été grisés. On peut remarquer que les nœuds actifs, en blanc, sont ceux correspondant aux contraintes de coupe avec capacités de type (5.36), (5.38), (5.39) et (5.40), ainsi que les contraintes $\sum_{e\in\delta_{G_i}(W)} x^1(e) \geq 0$ et $\sum_{e\in\delta_{G_i}(W)} x^2(e) \geq 0$ de départ.

Les relations de dominance qui ont été utilisées pour rendre inactifs certains des nœuds du graphe sont données par la figure 5.13, chaque arête représentant une relation de dominance. Chaque arête est numérotée suivant son ordre d'apparition lors de la construction du graphe de la figure 5.12. Les preuves de relations de dominance qui ne sont pas triviales sont données ci-dessous.

- (5) : Il est clair que $\left\lceil \frac{D_W}{4} \right\rceil \leq \left\lceil \frac{D_W}{2} \right\rceil$ donc $\left\lceil \frac{\left\lceil \frac{D_W}{4} \right\rceil + \left\lceil \frac{D_W}{2} \right\rceil}{2} \right\rceil \leq \left\lceil \frac{D_W}{2} \right\rceil$. Et par conséquent, la contrainte $(1, 2, \left\lceil \frac{\left\lceil \frac{D_W}{4} \right\rceil + \left\lceil \frac{D_W}{2} \right\rceil}{2} \right\rceil)$ est dominée par la contrainte $(1, 2, \left\lceil \frac{D_W}{2} \right\rceil)$.
- (7) : Soit $D_W = 4t + r$ avec $r \in \{0, 1, \ldots, 3\}$.

$$\left\lceil \frac{D_W}{2} \right\rceil = \left\lceil \frac{4t+r}{2} \right\rceil = \begin{cases} 2t & \text{si } r = 0, \\ 2t+1 & \text{si } r = 1, 2, \text{ et} \\ 2t+2 & \text{si } r = 3, \end{cases}$$

$$\begin{aligned} \left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil &= \left\lceil \frac{\left\lceil \frac{12t+3r}{4} \right\rceil + \left\lceil \frac{4t+r}{4} \right\rceil}{2} \right\rceil \\ &= \left\lceil \frac{4t+r+\left\lceil \frac{r}{4} \right\rceil}{2} \right\rceil = \begin{cases} 2t & \text{si } r = 0, \\ 2t+1 & \text{si } r = 1, \\ 2t+2 & \text{si } r = 2, 3. \end{cases} \end{aligned}$$

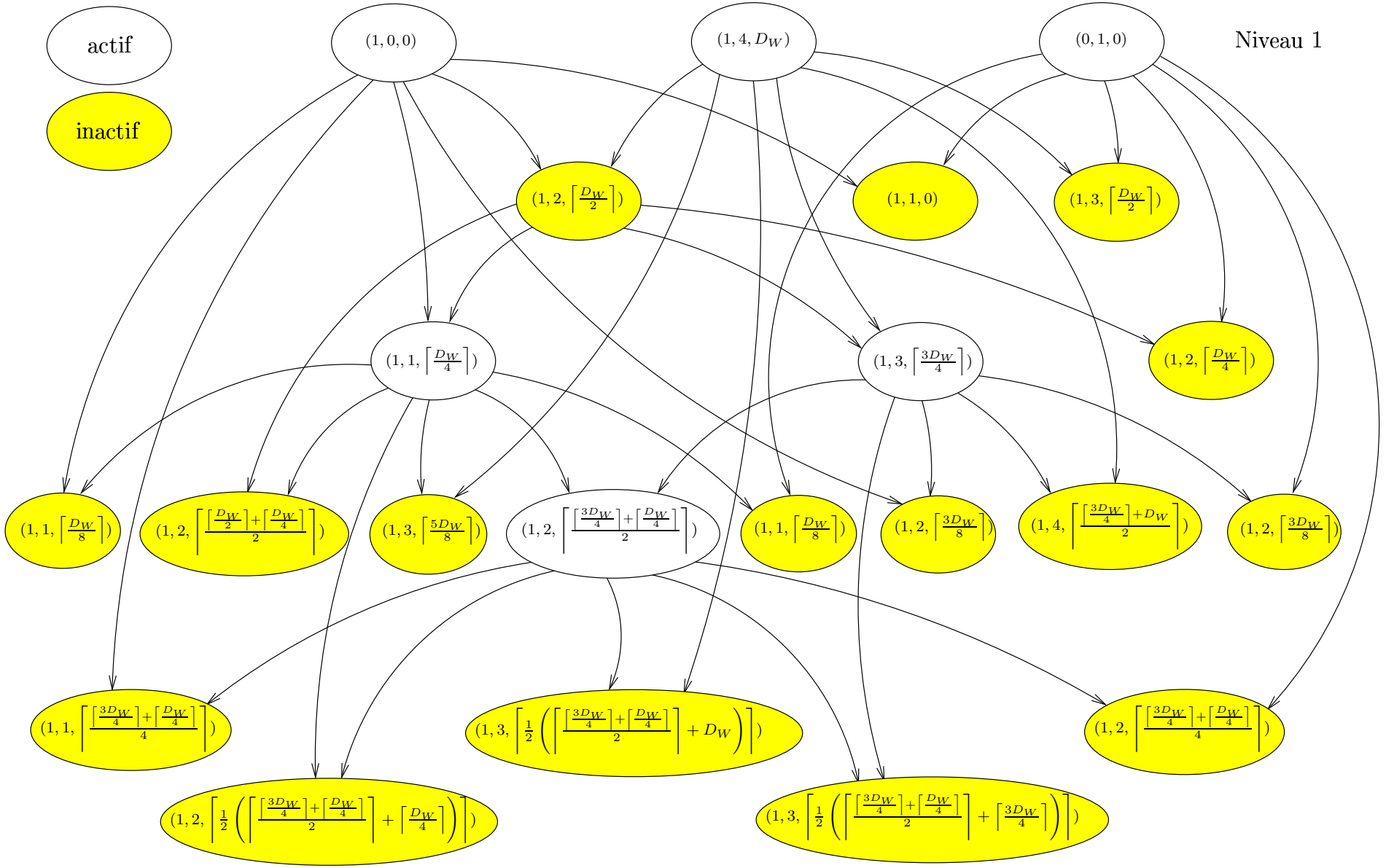


FIG. 5.12 – Création de nouvelles contraintes

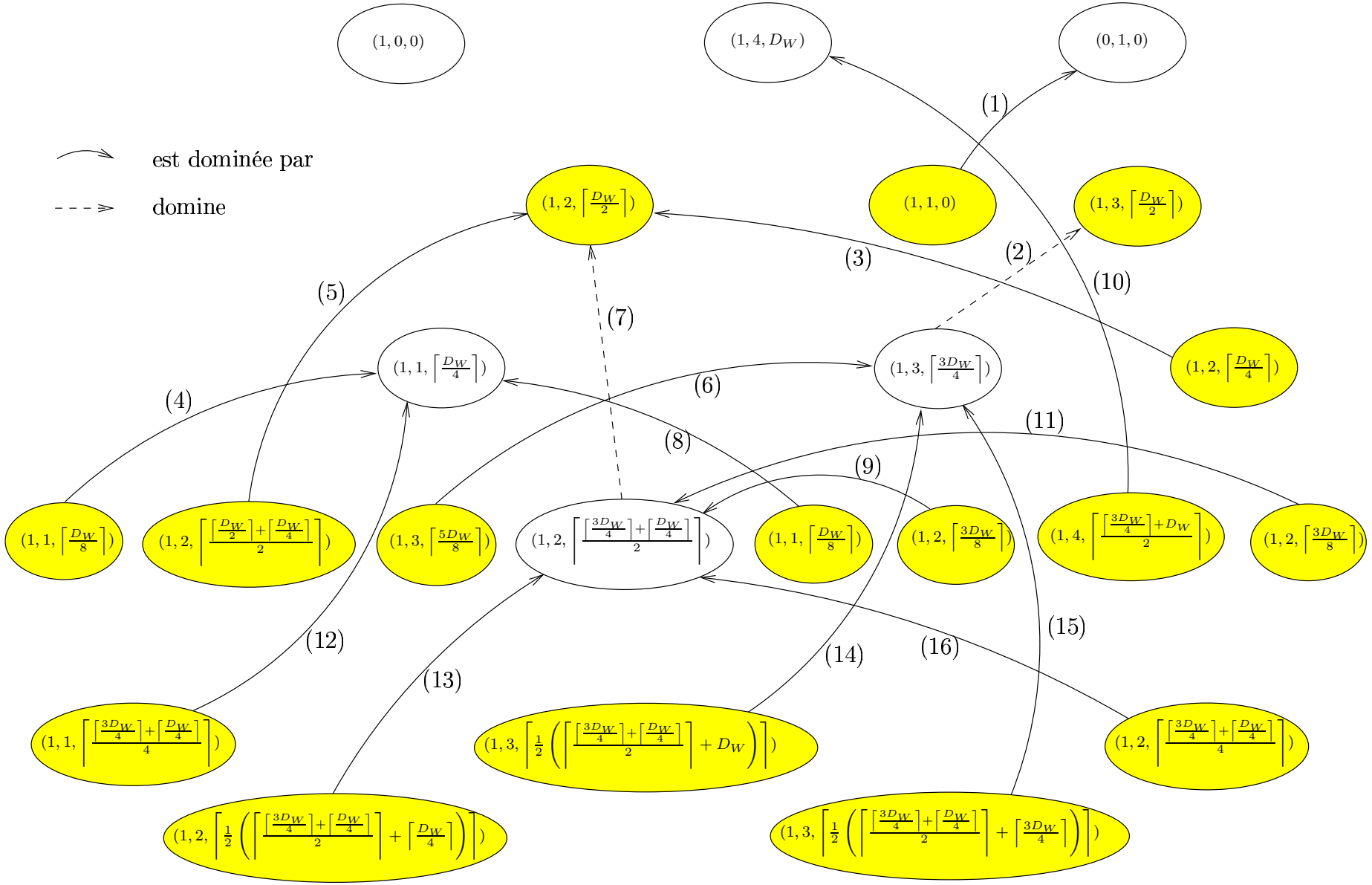


FIG. 5.13 – Relations de dominance obtenues lors de la contraction

Donc $\left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil \geq \left\lceil \frac{D_W}{2} \right\rceil$.

Et par conséquent, la contrainte $(1, 2, \left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil)$ domine la contrainte $(1, 2, \left\lceil \frac{D_W}{2} \right\rceil)$.

- (9) : On a $\left\lceil \frac{3D_W}{8} \right\rceil \leq \left\lceil \frac{D_W}{2} \right\rceil$ et on sait que $\left\lceil \frac{D_W}{2} \right\rceil \leq \left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil$ (voir preuve (7)).

  Donc la contrainte $(1, 2, \left\lceil \frac{3D_W}{8} \right\rceil)$ est dominée par la contrainte $(1, 2, \left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil)$.
- (10) : Comme $\left\lceil \frac{3D_W}{4} \right\rceil \leq D_W$, on a $\left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + D_W}{2} \right\rceil \leq D_W$ et par conséquent la contrainte $(1, 4, \left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + D_W}{2} \right\rceil)$ est dominée par la contrainte $(1, 4, D_W)$.
- (11) : identique à (9).
- (12) : Soit $D_W = 4t + r$ avec $r \in \{0, 1, 2, 3\}$.

$$\begin{aligned} \left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{4} \right\rceil &= \left\lceil \frac{\left\lceil \frac{12t+3r}{4} \right\rceil + \left\lceil \frac{4t+r}{4} \right\rceil}{4} \right\rceil \\ &= t + \left\lceil \frac{\left\lceil \frac{3}{4}r \right\rceil + \left\lceil \frac{r}{4} \right\rceil}{4} \right\rceil = \begin{cases} t & \text{si } r = 0, \\ t+1 & \text{si } r = 1, 2, 3, \end{cases} \end{aligned}$$

  et $\left\lceil \frac{D_W}{4} \right\rceil = \left\lceil \frac{4t+r}{4} \right\rceil = t + \left\lceil \frac{r}{4} \right\rceil = \begin{cases} t & \text{si } r = 0, \\ t+1 & \text{si } r = 1, 2, 3. \end{cases}$

  Donc $\left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{4} \right\rceil = \left\lceil \frac{D_W}{4} \right\rceil$. Et par conséquent, la contrainte $(1, 1, \left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{4} \right\rceil)$ est identique à la contrainte $(1, 1, \left\lceil \frac{D_W}{4} \right\rceil)$.
- (13) : Comme $\left\lceil \frac{3D_W}{4} \right\rceil \geq \left\lceil \frac{D_W}{4} \right\rceil$, on a $\left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil \geq \left\lceil \frac{D_W}{4} \right\rceil$

  et $\left\lceil \frac{1}{2} \left(\left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil \right) \right\rceil \leq \left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil$.

  Par conséquent, la contrainte $(1, 2, \left\lceil \frac{1}{2} \left(\left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil \right) \right\rceil)$ est dominée par la contrainte $(1, 2, \left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil)$.
- (14) : Soit $D_W = 4t + r$ avec $r \in \{0, 1, 2, 3\}$. Il s'ensuit $\left\lceil \frac{3D_W}{4} \right\rceil = \left\lceil \frac{12t+3r}{4} \right\rceil = 3t + r$.

  On a aussi

$$\begin{aligned} \left\lceil \frac{1}{2} \left(\left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil + D_W \right) \right\rceil &= \left\lceil \frac{1}{2} \left(\left\lceil \frac{\left\lceil \frac{12t+3r}{4} \right\rceil + \left\lceil \frac{4t+r}{4} \right\rceil}{2} \right\rceil + 4t + r \right) \right\rceil \\ &= \left\lceil \frac{1}{2} \left(6t + r + \left\lceil \frac{\left\lceil \frac{3r}{4} \right\rceil + \left\lceil \frac{r}{4} \right\rceil}{2} \right\rceil \right) \right\rceil \\ &= 3t + \left\lceil \frac{1}{2} \left(r + \left\lceil \frac{\left\lceil \frac{3r}{4} \right\rceil + \left\lceil \frac{r}{4} \right\rceil}{2} \right\rceil \right) \right\rceil \\ &= 3t + r. \end{aligned}$$

Donc $\left\lceil \frac{3D_W}{4} \right\rceil = \left\lceil \frac{1}{2}\left(\left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil + D_W \right) \right\rceil$ et par conséquent, la contrainte $(1, 3, \left\lceil \frac{3D_W}{4} \right\rceil)$ est identique à la contrainte $(1, 3, \left\lceil \frac{1}{2}\left(\left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil + D_W \right) \right\rceil)$.

– (15) : Comme $\left\lceil \frac{D_W}{4} \right\rceil \leq \left\lceil \frac{3D_W}{4} \right\rceil$, on a $\left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil \leq \left\lceil \frac{3D_W}{4} \right\rceil$. Il s'ensuit que $\left\lceil \frac{1}{2}\left(\left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil + \left\lceil \frac{3D_W}{4} \right\rceil \right) \right\rceil \leq \left\lceil \frac{3D_W}{4} \right\rceil$, et par la preuve de (14), la contrainte $(1, 3, \left\lceil \frac{1}{2}\left(\left\lceil \frac{\left\lceil \frac{3D_W}{4} \right\rceil + \left\lceil \frac{D_W}{4} \right\rceil}{2} \right\rceil + \left\lceil \frac{3D_W}{4} \right\rceil \right) \right\rceil)$ est dominée par la contrainte $(1, 3, \left\lceil \frac{3D_W}{4} \right\rceil)$.

Le graphe de la figure 5.12 donne de manière exhaustive, pour $W \subseteq V$, $W \neq \emptyset$, $W \neq V$ et $F_i \in \mathcal{F}$, toutes les contraintes qui pourraient être générées à l'aide d'une procédure de Chvatal-Gomory à partir des contraintes $\sum_{e \in \delta_{G_i}(W)} (x^1(e) + 4x^2(e)) \geq D_W$, $x^1(e) \geq 0$ pour tout $e \in \delta_{G_i}(W)$ et $x^2(e) \geq 0$ pour tout $e \in \delta_{G_i}(W)$ et qui ne sont pas directement dominées par d'autres. On peut remarquer que les nœuds actifs contiennent les contraintes de coupe avec capacités de type (5.36), (5.38), (5.39) et (5.40) définies précédemment. Cela montre bien que seules ces contraintes du type contraintes de coupe avec capacités sont intéressantes.

Dans ce qui suit, nous donnons des conditions nécessaires et des conditions suffisantes pour que les inégalités (5.36) définissent des facettes du polytope $\mathrm{MCSIPND}_m^{as}(G, \mathcal{F}, K)$.

Théorème 5.11 *L'inégalité (5.36) définit une facette de $MCSIPND_m^{as}(G, \mathcal{F}, K)$ seulement si*

1) $G_i(W)$ et $G_i(\overline{W})$ sont connexes,

2) il n'existe pas $j \in \{1, \ldots, t\} \backslash \{i\}$ tel que $F_i \cap \delta_G(W) \subset F_j \cap \delta_G(W)$,

3) $G(W)$ et $G(\overline{W})$ sont $\mathcal{F}$-connexes, si $\delta_G(W) \cap F_i = \emptyset$,

4) $D_W > max \left\{ \frac{\sum_{k \in \gamma^+(W)} \omega_k}{2.5}, \frac{\sum_{k \in \gamma^-(W)} \omega_k}{2.5} \right\}$,

5) $D_W \geq 4$.

Preuve. 1) Supposons s.p.d.g. que $G_i(W)$ n'est pas connexe. Il existe donc une partition W_1, W_2 de W telle que $\delta_{G_i}(W_1, W_2) = \emptyset$ (voir figure 5.14). Donc $\delta_{G_i}(W_1) = \delta_{G_i}(W_1, \overline{W})$ et $\delta_{G_i}(W_2) = \delta_{G_i}(W_2, \overline{W})$.
Cela implique que

$$\begin{aligned} & x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) \\ &= x^1(\delta_{G_i}(W_1, W)) + 4x^2(\delta_{G_i}(W_1, W)) + x^1(\delta_{G_i}(W_2, W)) + 4x^2(\delta_{G_i}(W_2, W)) \\ &= x^1(\delta_{G_i}(W_1)) + 4x^2(\delta_{G_i}(W_1)) + x^1(\delta_{G_i}(W_2)) + 4x^2(\delta_{G_i}(W_2)) \\ &\geq D_{W_1} + D_{W_2} \geq D_W. \end{aligned}$$

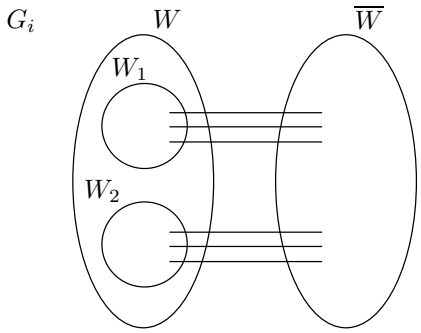


FIG. 5.14 –

Cette inégalité domine la contrainte (5.36). Il en résulte que (5.36) ne peut pas définir une facette.

2) Supposons le contraire. Soit $j \in \{1, \ldots, t\} \backslash \{i\}$ tel que $F_j \cap \delta_G(W) \supset F_i \cap \delta_G(W)$. Alors (5.36) est la combinaison linéraire des contraintes suivantes.

$$\begin{aligned} x^1(\delta_{G_j}(W)) + 4x^2(\delta_{G_j}(W)) &\geq D_W, \\ x^1_{uv} &\geq 0 \qquad \forall uv \in (F_j \backslash F_i) \cap \delta_G(W), \\ x^2_{uv} &\geq 0 \qquad \forall uv \in (F_j \backslash F_i) \cap \delta_G(W). \end{aligned}$$

Donc (5.36) ne peut pas définir une facette.

3) Supposons que $\delta_G(W) \cap F_i = \emptyset$ et, s.p.d.g., que $G(W)$ n'est pas $\mathcal{F}$-connexe. Il existe alors $j \in \{1, \ldots, t\}$ tel que $G_j(W)$ n'est pas connexe. Si $\delta_G(W) \cap F_j \neq \emptyset$, comme $\delta_G(W) \cap F_i = \emptyset$, alors, par la condition 2), (5.36) ne peut pas définir une facette. Supposons donc que $\delta_G(W) \cap F_j = \emptyset$. Alors $\delta_G(W) = \delta_{G_i}(W) = \delta_{G_j}(W)$, et par conséquent, il existe une partition W_1, W_2 de W telle que $\delta_{G_j}(W_1, W_2) = \emptyset$. Il en résulte que $\delta_{G_i}(W_1) = \delta_{G_i}(W_1, \overline{W})$ et $\delta_{G_i}(W_2) = \delta_{G_i}(W_2, \overline{W})$. Cela implique que

$$\begin{aligned} & x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) \\ &= x^1(\delta_{G_j}(W)) + 4x^2(\delta_{G_j}(W)) \\ &= x^1(\delta_{G_j}(W_1, W)) + 4x^2(\delta_{G_j}(W_1, W)) + x^1(\delta_{G_j}(W_2, W)) + 4x^2(\delta_{G_j}(W_2, W)) \\ &= x^1(\delta_{G_j}(W_1)) + 4x^2(\delta_{G_j}(W_1)) + x^1(\delta_{G_j}(W_2)) + 4x^2(\delta_{G_j}(W_2)) \\ &\geq D_{W_1} + D_{W_2} \geq D_W. \end{aligned}$$

Comme cette inégalité domine (5.36), cette dernière ne peut donc définir une facette.

4) Si $D_W = \text{Max} \left\{ \frac{\sum_{k \in \gamma^+(W)} \omega_k}{2.5}, \frac{\sum_{k \in \gamma^-(W)} \omega_k}{2.5} \right\}$, alors par la remarque 5.5, la contrainte (5.36) est redondante par rapport aux contraintes (5.1)-(5.3).

5) Si $D_W < 4$, alors dans toute solution du problème satisfaisant (5.36) à l'égalité, on a $x^2(e) = 0$ pour toute arête $e \in \delta_{G_i}(W)$.

Si $|\delta_{G_i}(W)| \geq 2$, alors il existe deux arêtes $e_1, e_2 \in \delta_{G_i}(W)$ telles que $x^2(e_1) = x^2(e_2) = 0$ dans toute solution satisfaisant (5.36) à l'égalité. Mais dans ce cas, on ne peut pas construire $\dim(\text{MCSIPND}_s^{as}(G, \mathcal{F}, K))$ solutions satisfaisant (5.36) à l'égalité et qui soient affinement indépendants. Cela implique que (5.36) ne peut pas définir une facette.

Supposons, par conséquent, que $\delta_{G_i}(W) = \{e = uv\}$ avec $u \in W$ et $v \in \overline{W}$. Alors dans toute solution vérifiant la contrainte (5.36) à l'égalité, on doit avoir

$$\begin{aligned} x^1(e) &= D_W, \\ x^2(e) &= 0. \end{aligned}$$

Aussi, dans toute solution vérifiant (5.36) à l'égalité, on doit également avoir

$$\begin{aligned} f_{uv}^{k,i} &= \begin{cases} \omega_k + \varepsilon^k \text{ si } k \in \gamma^+(W) \\ \varepsilon^k \text{ si } k \in \gamma^-(W) \end{cases} \\ f_{vu}^{k,i} &= \begin{cases} \omega_k + \varepsilon^k \text{ si } k \in \gamma^-(W) \\ \varepsilon^k \text{ si } k \in \gamma^+(W) \end{cases} \end{aligned}$$

où ε^k, $k \in \gamma(W)$ est strictement positif et suffisamment petit. En effet, on peut voir que les variables f n'interviennent pas dans la contrainte (5.36). De plus, du fait que par la condition 4), $D_W > \text{Max}\left\{\frac{\sum_{k\in\gamma^+(W)}\omega_k}{2.5}, \frac{\sum_{k\in\gamma^-(W)}\omega_k}{2.5}\right\}$, la capacité installée est toujours strictement supérieure à la demande traversant la coupe, dans un sens comme dans l'autre. Ceci donne la possibilité de faire passer un flot légèrement supérieur à la demande. Ce qui explique l'ajout de ε^k dans les deux sens.

Soit la matrice dans laquelle les colonnes correspondent aux solutions satisfaisant (5.36) à l'égalité et les lignes sont associées à $x^1(e)$, $x^2(e)$, la somme des variables $f_{uv}^{k,i}$ pour $k \in \gamma(W)$ et la somme des variables $f_{vu}^{k,i}$ pour $k \in \gamma(W)$. On note p le nombre de solutions vérifiant (5.36) à l'égalité. Cette matrice a donc la forme suivante :

$$\begin{array}{l} x^1(e) \\ x^2(e) \\ \displaystyle\sum_{k\in\gamma(W)} f_{uv}^{k,i} \\ \displaystyle\sum_{k\in\gamma(W)} f_{vu}^{k,i} \end{array} \begin{pmatrix} D_W & \cdots & D_W \\ 0 & \cdots & 0 \\ \displaystyle\sum_{k\in\gamma^+(W)} \omega_k + \sum_{k\in\gamma(W)} \varepsilon_1^k & \cdots & \displaystyle\sum_{k\in\gamma^+(W)} \omega_k + \sum_{k\in\gamma(W)} \varepsilon_p^k \\ \displaystyle\sum_{k\in\gamma^-(W)} \omega_k + \sum_{k\in\gamma(W)} \varepsilon_1^k & \cdots & \displaystyle\sum_{k\in\gamma^-(W)} \omega_k + \sum_{k\in\gamma(W)} \varepsilon_p^k \end{pmatrix}.$$

En soustrayant la quatrième ligne à la troisième, on obtient une ligne qui est multiple de la première ligne. Comme la seconde ligne est composée seulement de

zéros, on ne peut pas avoir suffisamment de solutions affinement indépendantes pour que (5.36) puisse définir une facette.

□

Théorème 5.12 *L'inégalité (5.36) définit une facette de* $MCSIPND_m^{as}(G, \mathcal{F}, K)$ *si*

1) les conditions 1), 2), 4), 5) du théorème 5.11 sont satisfaites,

2) $G(W)$ *et* $G(\overline{W})$ *sont* $\mathcal{F}$*-connexes.*

Preuve. La preuve utilise des idées similaires à celles développées dans [61]. Notons l'inégalité (5.36) par

$$a^1x^1 + a^2x^2 \quad \geq \quad \alpha, \tag{5.41}$$

et soit

$$b^1x^1 + b^2x^2 + \lambda f \quad \geq \quad \beta, \tag{5.42}$$

une inégalité définissant une facette de $\text{MCSIPND}_m^{as}(G, \mathcal{F}, K)$ telles que la face définie par (5.41) est contenue dans celle définie par (5.42). Soit $L = \{(x^1, x^2, f) \in \text{MCSIPND}_m^{as}(G, \mathcal{F}, K) \,|\, a^1x^1 + a^2x^2 = \alpha\}$.

Nous allons tout d'abord construire une solution réalisable satisfaisant (5.41) à l'égalité. Pour une demande k telle que o_k et d_k sont dans W, comme $G(W)$ est $\mathcal{F}$-connexe, pour tout $j \in \{1, \ldots, t\}$, il existe une chaîne P_j^k dans $G_j(W)$ reliant o_k et d_k. Si on installe $\lceil \frac{\omega_k}{10} \rceil$ grandes capactités sur chaque arête appartenant à au moins une chaîne P_j^k, $j = 1, \ldots, t$, alors le volume ω_k peut être écoulé de o_k vers d_k, quelle que soit la panne j. Ainsi on peut envoyer un flot de valeur ω_k sur P_j^k. En ajoutant successivement de la même façon la capacité nécessaire pour chacune des demandes, on obtient un dimensionnement réalisable pour les arêtes de $E(W)$ concernant les demandes de W. Pour les arêtes de $E(\overline{W})$, on peut associer d'une manière similaire, à chaque demande k telle que o_k et $d_k \in \overline{W}$ et chaque $j \in \{1, \ldots, t\}$, une chaîne $\overline{P}_j^k$ entre o_k et d_k. On dimensionne ces chaînes de la même manière que dans $G(W)$.

Maintenant, considérons une arête u_iv_i de $\delta_{G_i}(W)$ telle que $u_i \in W$ et $v_i \in \overline{W}$. Soit $I_i = \{j \in \{1, \ldots, t\} \backslash \{i\} \,|\, u_iv_i \in F_j\}$. Alors pour tout $j \in I_i$, il existe une arête disons u_jv_j dans $\delta(W) \cap F_i$ telle que $u_jv_j \notin F_j$. En effet, si ce n'est pas le cas, alors il existe $j \in I_i$ tel que $\delta(W) \cap F_i \subseteq \delta(W) \cap F_j$. Comme par définition de I_i, $u_iv_i \in F_j \backslash F_i$, on a $\delta(W) \cap F_i \subset F_j$, ce qui contredit la condition 2) du théorème 5.11 (voir figure 5.15).

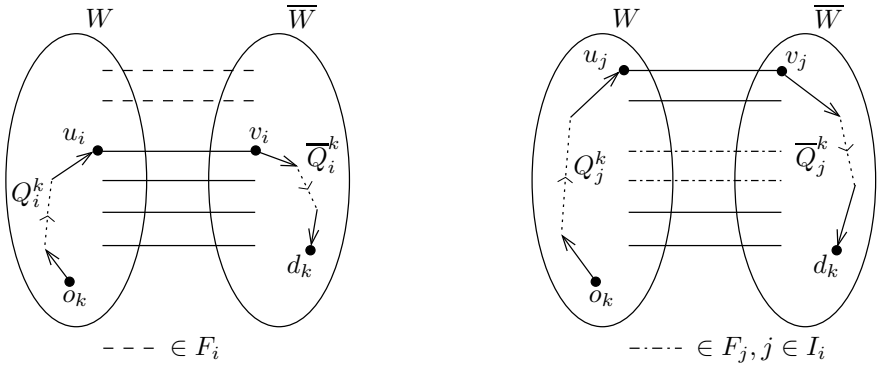


FIG. 5.15 –

Soit $k \in \gamma(W)$ une demande traversant la coupe. Comme le graphe $G(W)$ (resp. $G(\overline{W})$) est $\mathcal{F}$-connexe, pour tout $j \in \{1, \ldots, t\} \backslash I_i$, il existe une chaîne Q_j^k (resp. $\overline{Q}_j^k$) entre o_k et u_i (resp. v_i et d_k) dans $G_j(W)$ (resp. $\overline{G}_j(W)$). Pour $j \in I_i$, par la remarque ci-dessus, il existe une arête $u_j v_j$ de $(\delta(W) \cap F_i) \backslash F_j$. On suppose que $u_j \in W$ et $v_j \in \overline{W}$. Pour la demande k, et pour tout $j \in I_i$, d'une manière similaire, il existe une chaîne Q_j^k (resp. $\overline{Q}_j^k$) entre o_k et u_j (resp. v_j et d_k) dans $G_j(W)$ (resp. $\overline{G}_j(W)$).

Maintenant, on peut compléter le dimensionnement partiel déjà effectué à l'intérieur des ensembles W et $\overline{W}$. On installe pour la demande k, $\left\lceil \frac{\omega_k}{10} \right\rceil$ grandes capacités sur chaque arête appartenant à au moins une chaîne Q_j^k, $\overline{Q}_j^k$, $j = 1 \ldots, t$. On ajoute successivement par la suite ces capacités pour chaque demande de $\gamma(W)$. Ces nouvelles capacités rajoutées à celles déjà installées pour écouler les demandes de W et $\overline{W}$ nous permettent d'avoir un dimensionnement réalisable pour toutes les demandes et les arêtes de $E(W) \cup E(\overline{W})$. Ce dimensionnement des arêtes de $E(W) \cup E(\overline{W})$ peut être donné comme suit

$$
\begin{aligned}
x_{uv}^1 &= 0 \quad \text{pour tout } uv \in E(W) \cup E(\overline{W}), \\
x_{uv}^2 &= \sum_{\substack{k \,|\, o_k, d_k \in W \\ uv \in \cup_j P_j^k}} \left\lceil \frac{\omega_k}{10} \right\rceil + \sum_{\substack{k \in \gamma(W) \\ uv \in \cup_j Q_j^k}} \left\lceil \frac{\omega_k}{10} \right\rceil \quad \text{pour tout } uv \in E(W), \\
x_{uv}^2 &= \sum_{\substack{k \,|\, o_k, d_k \in W \\ uv \in \cup_j \overline{P}_j^k}} \left\lceil \frac{\omega_k}{10} \right\rceil + \sum_{\substack{k \in \gamma(W) \\ uv \in \cup_j \overline{Q}_j^k}} \left\lceil \frac{\omega_k}{10} \right\rceil \quad \text{pour tout } uv \in E(\overline{W}).
\end{aligned}
\tag{5.43}
$$

Comme $D_W \geq 4$, pour les arêtes de $\delta(W)$, on peut considérer le dimensionnement

suivant

$$\begin{array}{lcll} x^1_{u_jv_j} & = & D_W \bmod 4 & \text{pour tout } j \in I_i \cup \{i\}, \\ x^1_{uv} & = & 0 & \text{pour tout } uv \in \delta(W) \setminus \cup_{j\in I_i\cup\{i\}} \{u_jv_j\}, \\ x^2_{u_jv_j} & = & \left\lfloor \frac{D_W}{4} \right\rfloor & \text{pour tout } j \in I_i \cup \{i\}, \\ x^2_{uv} & = & 0 & \text{pour tout } uv \in \delta(W) \setminus \cup_{j\in I_i\cup\{i\}} \{u_jv_j\}. \end{array} \tag{5.44}$$

En ce qui concerne les flots, il suffit de les envoyer sur les chemins P^k_j, $\overline{P}^k_j$, Q^k_j, $\overline{Q}^k_j$, $j = 1, \ldots, t$. On peut les définir comme suit

$$\begin{array}{ll} f^{k,i}_{uv} = \omega_k & \text{pour tout } k \text{ tel que } o_k, d_k \in W \text{ (resp. } o_k, d_k \in \overline{W}), \\ & j = 1, \ldots, t,\ uv \in P^k_j, \text{ (resp. } uv \in \overline{P}^k_j) \text{ et } uv \text{ traversé de } u \\ & \text{vers } v \text{ dans } P^k_j \text{ (resp. } \overline{P}^k_j), \\ f^{k,i}_{uv} = \omega_k & \text{pour tout } k \in \gamma(W)\, j = 1, \ldots, t,\ uv \in Q^k_j, \text{ (resp. } uv \in \overline{Q}^k_j) \\ & \text{et } uv \text{ traversé de } u \text{ vers } v \text{ dans } Q^k_j \text{ (resp. } \overline{Q}^k_j), \\ f^{k,i}_{uv} = \omega_k & \text{pour tout } k \in \gamma^+(W) \text{ (resp. } \gamma^-(W)), \\ & u = u_j\,(v_j), v = v_j\,(u_j)\, j \in I_i \cup \{i\}, \\ f^{k,i}_{uv} = 0 & \text{sinon.} \end{array} \tag{5.45}$$

Ainsi, la solution (x^1, x^2, f) donnée par (5.43), (5.44) et (5.45) est réalisable pour $\text{MCSIPND}^{as}_m(G, \mathcal{F}, K)$.

Soit $pq \in E(W) \cup E(\overline{W}) \cup (F_i \cap \delta(W))$. Et soit $(x^{1'}, x^{2'}, f')$ telle que

$$\begin{array}{lcll} x^{1'}_{uv} & = & x^1_{uv} & \text{pour tout } uv \neq pq, \\ x^{1'}_{pq} & = & x^1_{pq} + 1, & \\ x^{2'} & = & x^2, & \\ f' & = & f. & \end{array}$$

$(x^{1'}, x^{2'}, f')$ est réalisable pour $\text{MCSIPND}^{as}_m(G, \mathcal{F}, K)$. De plus, (x^1, x^2, f) et $(x^{1'}, x^{2'}, f')$ satisfont la contrainte (5.41) à l'égalité. Par conséquent, elles satisfont également la contrainte (5.42) à l'égalité. En soustrayant les équations correspondantes on obtient $b^1_{pq} = 0$. D'une manière similaire, on montre que $b^2_{pq} = 0$. Comme pq a été choisi arbitrairement dans $E(W) \cup E(\overline{W}) \cup (F_i \cap \delta(W))$, on a $b^1_{uv} = b^2_{uv} = 0$ pour tout $uv \in E(W) \cup E(\overline{W}) \cup (F_i \cap \delta(W))$.

Considérons l'arête u_iv_i (introduite précédemment) et la solution $(\tilde{x}^1, \tilde{x}^2, \tilde{f})$ telle que

$$\begin{array}{lcll} \tilde{x}^1_{u_iv_i} & = & x^1_{u_iv_i} + 4, & \\ \tilde{x}^1_{uv} & = & x^1_{uv} & \text{pour tout } uv \in E \text{ tel que } uv \neq u_iv_i, \\ \tilde{x}^2_{u_iv_i} & = & x^2_{u_iv_i} - 1, & \\ \tilde{x}^2_{uv} & = & x^2_{uv} & \text{pour tout } uv \in E \text{ tel que } uv \neq u_iv_i, \\ \tilde{f} & = & f. & \end{array}$$

La solution $(\tilde{x}^1, \tilde{x}^2, \tilde{f})$ est définie à partir de (x^1, x^2, f) en remplaçant une grande capacité sur u_iv_i par 4 petites. Elle est donc réalisable. De plus $(\tilde{x}^1, \tilde{x}^2, \tilde{f})$ satisfait (5.41) à l'égalité, et donc également la contrainte (5.42). En soustrayant les équations données par cette dernière pour les solutions (x^1, x^2, f) et $(\tilde{x}^1, \tilde{x}^2, \tilde{f})$, on obtient $4b^1_{u_iv_i} = b^2_{u_iv_i}$. Comme u_iv_i est arbitraire dans $\delta_{G_i}(W)$, il en résulte que $b^2_{uv} = 4b^1_{uv}$ pour tout $uv \in \delta(W)\backslash F_i$.

Considérons de nouveau une arête $pq \in E(W) \cup E(\overline{W}) \cup F_i$ et la solution (x^1, x^2, f) introduite ci-dessus. Soient une demande $k' \in K$ et une panne $j' \in \{1, \ldots, t\}$ et soit la solution $(\hat{x}^1, \hat{x}^2, \hat{f})$ telle que

$$\begin{aligned}
\hat{x}^1 &= x^1, \\
\hat{x}^2 &= x^2, \\
\hat{f}^{k',j'}_{pq} &= f^{k',j'}_{pq} + \varepsilon, \\
\hat{f}^{k',j'}_{qp} &= f^{k',j'}_{qp} + \varepsilon, \\
\hat{f}^{k,j}_{uv} &= f^{k,j}_{uv} \text{ pour tout } uv \in E\backslash\{pq\}, \text{ pour tout } k \in K, \text{ pour tout } j \in \{1, \ldots, t\}.
\end{aligned}$$

où $0 < \varepsilon < \frac{1}{2}$. Noter que comme l'arête pq est surdimensionnée, il reste de la capacité résiduelle pour envoyer du flot. En conséquence, $(\hat{x}^1, \hat{x}^2, \hat{f})$ est réalisable. Comme elle vérifie la contrainte (5.41) à l'égalité, elle vérifie également (5.42) à l'égalité. Comme la la solution (x^1, x^2, f) satisfait (5.42) à l'égalité, en soustrayant les égalités correspondantes, on obtient $\lambda^{k',j'}_{pq}\varepsilon + \lambda^{k',j'}_{qp}\varepsilon = 0$. Comme $\varepsilon > 0$, ceci implique que $\lambda^{k',j'}_{pq} = -\lambda^{k',j'}_{qp}$. Comme pq est arbitraire dans $E(W) \cup E(\overline{W}) \cup F_i$, et k' et j' sont également choisis de façon arbitraire dans K et $\{1, \ldots, t\}$, respectivement, il en résulte que $\lambda^{k,j}_{uv} = -\lambda^{k,j}_{vu}$ pour tout $uv \in E(W) \cup E(\overline{W}) \cup F_i$, pour tout $k \in K$ et tout $j \in \{1, \ldots, t\}$.

Considérons de nouveau l'arête u_iv_i de $\delta(W)\backslash F_i$. Par la condition 4) du théoreme 5.11 et la définition de la solution (x^1, x^2, f), u_iv_i est surdimensionnée. Donc on peut ajouter du flot sur cette arête. D'une manière similaire on peut donc montrer que $\lambda^{k',j'}_{u_iv_i} = -\lambda^{k',j'}_{v_iu_i}$ et comme u_iv_i est arbitrairement choisie dans $\delta_{G_i}(W)$, tout comme k' et j' dans K et $\{1, \ldots, t\}$, on obtient $\lambda^{k,j}_{uv} = -\lambda^{k,j}_{vu}$ pour tout $uv \in \delta_{G_i}(W)$, pour tout $k \in K$ et tout $j \in \{1, \ldots, t\}$. On a ainsi montré que

$$\lambda^{k,i}_{uv} = -\lambda^{k,i}_{vu} \text{ pour tout } uv \in E, \text{ pour tout } k \in K \text{ et pour tout } j \in \{1, \ldots, t\}. \quad (5.46)$$

On montre à présent que $\sum_{k\in K}\sum_{uv\in E}(\lambda^{k,j}_{uv} f^{k,j}_{uv} + \lambda^{k,j}_{vu} f^{k,j}_{vu})$ est une constante pour toute panne $j = 1, \ldots, t$, en montrant que la somme des coefficients correspondant aux arcs d'un cycle est égale à zéro. Soit Δ l'ensemble des cycles dans $D = (V, A)$. Considérons une panne $j \in \{1, \ldots, t\}$ et un cycle $\xi \in \Delta$. Soit $\lambda^{k,j}_{\xi} = \sum_{(u,v)\in\xi} \lambda^{k,j}_{uv}$.

On va montrer que $\lambda_\xi^{k,j} = 0$ pour tout cycle $\xi \in \Delta$. Un cycle ξ est dit *s-intersectant* par rapport à la coupe $\delta_G(W)$ si ξ contient exactement s arcs de la coupe $\delta_G(W)$. On remarque que s est forcément pair.

Si ξ est un cycle 0-intersectant, cela signifie que ξ est complètement contenu dans W ou $\overline{W}$. On suppose s.p.d.g. que ξ est dans W. On considère une demande $k \in K$. Soit la solution $(\check{x}^1, \check{x}^2, \check{f})$ définie comme suit,

$$\begin{array}{rcll}
\check{x}^1_{uv} & = & x^1_{uv} + 1 & \text{pour tout } uv \in E(W) \text{ tel que } (u,v) \text{ ou } (v,u) \in \xi, \\
\check{x}^1_{uv} & = & x^1_{uv} & \text{pour tout } uv \in E(W) \text{ tel que } (u,v) \text{ ou } (v,u) \notin \xi, \\
\check{x}^2_{uv} & = & x^2_{uv} & \text{pour tout } uv \in E, \\
\check{f}^{k,j}_{uv} & = & f^{k,j}_{uv} + 1 & \text{pour tout } (u,v) \in \xi, \\
\check{f}^{k,j}_{uv} & = & f^{k,j}_{uv} & \text{pour tout } (u,v) \notin \xi, \\
\check{f}^{k,h}_{uv} & = & f^{k,h}_{uv} & \text{pour tout } (u,v) \in A, \text{ pour tout } h \in \{1,\ldots,t\}\backslash\{j\}.
\end{array}$$

Autrement dit, on construit la solution $(\check{x}^1, \check{x}^2, \check{f})$ à partir de la solution (x^1, x^2, f) en installant une petite capacité sur toute arête de ξ et en envoyant une unité additionnelle de flot sur ξ. La solution reste réalisable et satisfait la contrainte (5.41) à l'égalité. La solution $(\check{x}^1, \check{x}^2, \check{f})$ appartient donc à L et vérifie la contrainte (5.42) à l'égalité. En soustrayant les équations données par cette dernière pour les solutions (x^1, x^2, f) et $(\check{x}^1, \check{x}^2, \check{f})$, on obtient

$$\sum_{uv \in E(W)} b^1_{uv} x^1_{uv} - \sum_{uv \in E(W)} b^1_{uv} \check{x}^1_{uv} + \sum_{(u,v)\in\xi} (\lambda^{k,j}_{uv} f^{k,j}_{uv} - \lambda^{k,j}_{vu} f^{k,j}_{vu}) = 0,$$

et comme $b^1_{uv} = 0$ pour tout $uv \in E(W)$, on obtient $\sum_{(u,v)\in\xi} -\lambda^{k,j}_{uv} = 0$ et alors $\lambda^{k,j}_\xi = 0$. Comme le cycle ξ, la demande k et la panne j ont été choisis de façon arbitraire, on a montré que

$$\lambda^{k,j}_\xi = 0 \qquad \begin{array}{l}\text{pour tout cycle 0-intersectant } \xi \in \Delta, \text{ toute demande } k \in K \\ \text{et toute panne } j \in \{1,\ldots,t\}.\end{array} \tag{5.47}$$

Supposons que ξ est un cycle 2-intersectant et supposons aussi que $(p,\bar{p})$ et $(q,\bar{q})$ sont les deux arêtes appartenant à $\delta_G(W) \cap \xi$ avec $p, q \in W$ et $\bar{p}, \bar{q} \in \overline{W}$.
Si $p = q$ et $\bar{p} = \bar{q}$, ξ peut être décomposé en trois cycles : un cycle 0-intersectant disons ξ_1 contenu dans W, un autre cycle 0-intersectant disons ξ_2 contenu dans $\overline{W}$ et un cycle formé par les arcs $(p,\bar{p})$ et $(\bar{p},p) = (\bar{q},q)$. On a alors $\lambda^{k,j}_\xi = \lambda^{k,j}_{\xi_1} + \lambda^{k,j}_{\xi_2} + \lambda^{k,j}_{p\bar{p}} + \lambda^{k,j}_{\bar{p}p}$. Comme ξ_1 et ξ_2 sont des cycles 0-intersectants, $\lambda^{k,j}_{\xi_1} = \lambda^{k,j}_{\xi_2} = 0$ et par (5.46), on a $\lambda^{k,j}_{p\bar{p}} = -\lambda^{k,j}_{\bar{p}p}$. Donc $\lambda^{k,j}_\xi = 0$.
Supposons maintenant que $p\bar{p} \neq q\bar{q}$ et l'une des deux arêtes $p\bar{p}$ ou $q\bar{q}$ appartient à F_i. On suppose, s.p.d.g., que $q\bar{q} \in F_i$ et $p\bar{p} \notin F_i$. Soit l'arête $p\bar{p}$ qui appartient à $\delta_{G_i}(W)$, on

construit une solution réalisable $(\breve{x}^1, \breve{x}^2, \breve{f})$ de la même façon que (x^1, x^2, f) (construite précédemment pour l'arête $u_i v_i$). On remarque par exemple que l'on a $\breve{x}^1_{p\bar{p}} = D_W \bmod 4$ et $\breve{x}^2_{p\bar{p}} = \lfloor \frac{D_W}{4} \rfloor$. On construit une nouvelle solution à partir de $(\breve{x}^1, \breve{x}^2, \breve{f})$ en installant une petite capacité sur toute arête de $\xi \cap (E(W) \cup E(\overline{W}) \cup F_i)$ et en envoyant ε unité de flot sur ξ (pour un certain $\varepsilon > 0$). L'ajout de ce flot additionnel est possible, car par la condition 4) du théoreme 5.11, on sait que l'on dispose d'une capacité résiduelle non nulle le long de l'arête $p\bar{p}$. Cette nouvelle solution est réalisable. On remarque qu'une petite capacité a été installée sur $q\bar{q}$. Mais comme $q\bar{q} \in F_i$, cette solution vérifie toujours la contrainte (5.41) à l'égalité et donc également la contrainte (5.42). En soustrayant les équations correspondant à la contrainte (5.42) pour $(\breve{x}^1, \breve{x}^2, \breve{f})$ et cette nouvelle solution, on montre que $\lambda^{k,j}_{\xi} = 0$.
Si à présent, nous avons toujours $p\bar{p} = q\bar{q}$, mais que $p\bar{p}, q\bar{q} \notin F_i$. On construit une solution $(\dot{x}^1, \dot{x}^2, \dot{f})$ en posant pour les arêtes de la coupe $\delta_{G_i}(W)$

$$\begin{array}{lcll} \dot{x}^1_{p\bar{p}} & = & D_W \bmod 4, & \\ \dot{x}^1_{uv} & = & 0 & \text{pour tout } uv \in \delta_{G_i}(W), \\ \dot{x}^2_{p\bar{p}} & = & \lfloor \frac{D_W}{4} \rfloor - 1, & \\ \dot{x}^2_{q\bar{q}} & = & 1, & \\ \dot{x}^2_{uv} & = & 0 & \text{pour tout } uv \in \delta_{G_i}(W) \backslash \{p\bar{p}, q\bar{q}\}. \end{array}$$

Pour le dimensionnement de $E(W)$ et $E(\overline{W})$, on suppose que l'on installe une capacité suffisante permettant de faire passer les demandes de W et $\overline{W}$ sur des chemins entièrement contenus dans $G(W)$ et $G(\overline{W})$. Ces capacités peuvent être en fait prise aussi grande que l'on veut. Aussi, par la condition 4) du théoreme 5.11, les demandes de $\gamma(W)$ peuvent être routées de telle sorte que $p\bar{p}$ et $q\bar{q}$ contiennent au moins ε (pour un certain $\varepsilon > 0$) unités de capacité résiduelle. Cette solution est réalisable et vérifie la contrainte (5.41) donc la contrainte (5.42) à l'égalité. Comme on dispose d'une capacité résiduelle ε sur $p\bar{p}$ et $q\bar{q}$, on peut déterminer une nouvelle solution réalisable qui diffère de la solution $(\dot{x}^1, \dot{x}^2, \dot{f})$ par deux points : pour la demande $k \in K$, on ajoute un flot additionnel ε le long de ξ et on installe une petite capacité sur les arêtes de ξ qui ne sont pas dans $\delta_{G_i}(W)$. Cette nouvelle solution est toujours réalisable et vérifie les contraintes (5.41) et (5.42) à l'égalité. En soustrayant les équations correspondant à la contrainte (5.42) pour ces deux solutions et comme la demande $k \in K$, la panne $j \in \{1, \ldots, t\}$ et le cycle 2-intersectant $\xi \in \Delta$ ont été choisis arbitrairement, on montre que

$$\lambda^{k,j}_{\xi} = 0 \qquad \begin{array}{l} \text{pour tout cycle 2-intersectant } \xi \in \Delta, \text{ toute demande } k \in K \\ \text{et toute panne } j \in \{1, \ldots, t\}. \end{array} \tag{5.48}$$

Suppososns à présent que ξ est un cycle s-intersectant. Soient k une demande et j une panne. Soit ξ défini par $\{(s_1, s_2), (s_2, s_3), \ldots, (s_T, s_1)\}$ avec $s_1 \in W$. Soit (s_{t_1}, s_{t_2})

le premier arc du cycle ξ contenu dans la coupe $\delta(W)$ et (s_{t_3}, s_{t_4}) celui qui revient vers W. On sait alors que le cycle $\xi' = \{(s_{t_1}, s_{t_2}), \ldots, (s_{t_3}, s_{t_4}), (s_{t_4}, s_{t_1})\}$ est un cycle 2-intersectant. On a alors $\lambda^{k,j}_{\xi'\setminus(s_{t_1},s_{t_4})} = \lambda^{k,j}_{s_{t_1}s_{t_4}}$. Il en resulte que l'on peut remplacer le chemin $\xi'\setminus(s_{t_1}, s_{t_4})$ par l'arc (s_{t_1}, s_{t_4}). En répétant cet argument, on peut construire un cycle 0-intersectant ψ qui satisfait $\lambda^{k,j}_{\xi} = \lambda^{k,j}_{\psi}$. Comme la demande k et la panne j ont été choisis arbitrairement, nous avons montré que

$$\lambda^{k,j}_{\xi} = 0 \qquad \begin{array}{l}\text{pour tout cycle } \xi \in \Delta, \text{ toute demande } k \in K \\ \text{et toute panne } j \in \{1, \ldots, t\}.\end{array} \tag{5.49}$$

On a alors montré que $\sum_{k\in K}\sum_{uv\in E}(\lambda^{k,j}_{uv} f^{k,j}_{uv} + \lambda^{k,j}_{vu} f^{k,j}_{vu})$ est une constante que l'on notera λ_j pour toute panne $j \in \{1, \ldots, t\}$.

On peut maintenant montrer que $b^1_{uv} = \rho^1$ et $b^2_{uv} = \rho^2$ pour tout $uv \in \delta_{G_i}(W)$. On considère la solution (x^1, x^2, f) donné au début, une arête $pq \in \delta_{G_i}(W)$ telle que $pq \neq u_iv_i$, une demande $k' \in K$ et une panne $j \in \{1, \ldots, t\}$. Soit $P(u_i, p)$ un chemin allant de u_i à p dans W et $P(q, v_i)$ un chemin allant de q à v_i dans $\overline{W}$. On construit la solution $(\ddot{x}^1, \ddot{x}^2, \ddot{f})$ comme suit

$$\begin{array}{lcll}
\ddot{x}^1_{u_iv_i} & = & x^1_{u_iv_i} - 1 \;=\; D_W \bmod 4 - 1, & \\
\ddot{x}^1_{pq} & = & 1, & \\
\ddot{x}^1_{uv} & = & x^1_{uv} & \text{pour tout } uv \in \delta_G(W)\setminus\{u_iv_i, pq\}, \\
\ddot{x}^1_{uv} & = & x^1_{uv} + 1 & \text{pour tout } (u,v) \in P(u_i,p) \cup P(q,v_i), \\
\ddot{x}^1_{uv} & = & x^1_{uv} & \text{pour tout } (u,v) \in E(W) \cup E(\overline{W})\setminus(P(u_i,p) \cup P(q,v_i)), \\
\ddot{x}^2_{uv} & = & x^2_{uv} & \text{pour tout } uv \in E.
\end{array}$$

On définit également le flot comme suit

$$\begin{array}{lcl}
\ddot{f}^{k',j}_{uv} & = & f^{k',j}_{uv} + 1 \quad \text{pour tout } (u,v) \in P(u_i,p) \cup P(q,v_i), \\
\ddot{f}^{k',j}_{u_iv_i} & = & f^{k',j}_{u_iv_i} - 1, \\
\ddot{f}^{k',j}_{pq} & = & 1, \\
\ddot{f}^{k',j}_{uv} & = & f^{k',j}_{uv} \quad \text{pour tout } uv \in E\setminus(P(u_i,p) \cup P(q,v_i) \cup \{u_iv_i, pq\}), \\
\ddot{f}^{k,h}_{uv} & = & f^{k,h}_{uv} \quad \text{pour tout } uv \in E, \text{ pour tout } k \in K\setminus\{k'\}, \text{ pour tout } h \in \{1,\ldots,t\}\setminus\{j\}.
\end{array}$$

Les solutions (x^1, x^2, f) et $(\ddot{x}^1, \ddot{x}^2, \ddot{f})$ sont réalisables et satisfont la contrainte (5.41) à l'égalité. Par conséquent, elles satisfont (5.42) à l'égalité. En soustrayant les équations correspondant à (5.42) pour les solutions (x^1, x^2, f) et $(\ddot{x}^1, \ddot{x}^2, \ddot{f})$, on obtient

$$b^1_{u_iv_i} - b^1_{pq} - \sum_{uv\in P(u_i,p)\cup P(q,v_i)} \rho^1_{uv} - \sum_{(u,v)\in P(u_i,p)} \lambda^{k',j}_{uv} - \sum_{(u,v)\in P(q,v_i)} \lambda^{k',j}_{uv} + \lambda^{k',j}_{u_iv_i} - \lambda^{k',j}_{pq} = 0.$$

Comme $b^1_{uv} = 0$ pour tout $uv \in E(W) \cup E(\overline{w})$ et par (5.46) $\lambda^{k',j}_{u_iv_i} = -\lambda^{k',j}_{v_iu_i}$, on a

$$b^1_{u_iv_i} - b^1_{pq} - \sum_{(u,v)\in P(u_i,p)} \lambda^{k',j}_{uv} - \lambda^{k',j}_{pq} - \sum_{(u,v)\in P(q,v_i)} \lambda^{k',j}_{uv} - \lambda^{k',j}_{v_iu_i} = 0.$$

$P(u_i,p)$, pq, $P(q,v_i)$ et v_iu_i forment un cycle donc on a $b^1_{u_iv_i} - b^1_{pq} = 0$. Comme pq est arbitraire, il s'ensuit que $b^1_{uv} = \rho^1$ pour tout $uv \in \delta_{G_i}(W)$ et comme $b^2_{uv} = 4b^1_{uv}$ pour tout $uv \in \delta_{G_i}(W)$, on a également $b^2_{uv} = 4\rho^1$ pour tout $uv \in \delta_{G_i}(W)$.

La contrainte (5.42) est donc équivalente à

$$\rho^1 x^1(\delta_{G_i}(W)) + 4\rho^1 x^2(\delta_{G_i}(W)) + \sum_{j=1}^{t} \lambda_j = \lambda.$$

Ce qui implique que

$$\rho^1 x^1(\delta_{G_i}(W)) + 4\rho^1 x^2(\delta_{G_i}(W)) = \lambda'$$

où $\lambda' = \lambda - \sum_{j=1}^{t} \lambda_j$. Comme la face définie par la contrainte (5.42) est non vide, $\rho^1 \neq 0$ et on obtient

$$x^1(\delta_{G_i}(W)) + 4x^2(\delta_{G_i}(W)) = \frac{\lambda'}{\rho^1} = D_W,$$

car (5.42) est vérifiée à l'égalité par toute solution de L. □

Dans [61], Magnanti, Mirchandani et Vachani ont introduit des contraintes de coupe avec capacités pour le problème TFLP (voir page 20). Ces contraintes peuvent être étendues aux problèmes MCSIPND$_m$ et MCSIPND$_s$. Comme il s'avère dans la suite, elles sont des cas particuliers des contraintes de coupe avec capacités de type (5.36), (5.38), (5.39) et (5.40).

Théorème 5.13 *Soient $F_i \in \mathcal{F}$ et $W \subseteq V$, $\emptyset \neq W \neq V$ et soit*

$$r_W = \begin{cases} 4 \text{ si } D_W \bmod 4 = 0, \\ D_W \bmod 4 \text{ sinon.} \end{cases}$$

Alors l'inégalité

$$\sum_{e\in\delta_{G_{j_i}}(W)} (x^1_e + r_W x^2_e) \geq r_W \left\lceil \frac{D_W}{4} \right\rceil \tag{5.50}$$

est valide pour MCSIPND$_m(G,\mathcal{F},K)$ et MCSIPND$_s(G,\mathcal{F},K)$.

Preuve. – Si $r_W = 1$, l'inégalité (5.50) n'est rien d'autre que la contrainte (5.38).

- Si $r_W = 2$, $\left\lceil \frac{D_W}{2} \right\rceil + 1 = 2 \left\lceil \frac{D_W}{4} \right\rceil$ et donc l'inégalité (5.50) n'est rien d'autre que la contrainte (5.40).
- Si $r_W = 3$, $\left\lceil \frac{3D_W}{4} \right\rceil = 3 \left\lceil \frac{D_W}{4} \right\rceil$ et donc l'inégalité (5.50) n'est rien d'autre que la contrainte (5.39).
- Si $r_W = 4$, $D_W = 4 \left\lceil \frac{D_W}{4} \right\rceil$ et donc l'inégalité (5.50) n'est rien d'autre que la contrainte (5.36).

□

5.5.2 Contraintes de saturation

Cette nouvelle classe de contraintes n'utilise que les variables associées aux arêtes de grande capacité (10Gbit/s). Ces contraintes sont valides uniquement pour le problème MCSIPND$_s$. Elles sont basées sur l'idée que si toutes les arêtes de petite capacité (2.5 Gbit/s) sont saturées et si la demande ne peut toujours pas être écoulée, il faudra installer au moins une arête de grande capacité.

Théorème 5.14 *Soient $F_i \in \mathcal{F}$ et $W \subseteq V$, $\emptyset \neq W \neq V$. Alors l'inégalité*

$$x^2(\delta_{G_i}(W)) \geq \left\lceil \frac{Max \left\{ \sum_{k \in \gamma^+(W)} \omega_k, \sum_{k \in \gamma^-(W)} \omega_k \right\} - |\delta_{G_i}(W)| \times 2.5}{7.5} \right\rceil \qquad (5.51)$$

est valide pour MCSIPND$_s^{as}(G, \mathcal{F}, K)$ et MCSIPND$_s^{ac}(G, \mathcal{F}, K)$.

Preuve. Supposons que l'on fixe $x^1(e) = 1$ pour tout $e \in \delta_{G_i}(W)$. La capacité disponible sur la coupe $\delta_{G_i}(W)$ est donc égale à $|\delta_{G_i}(W)| \times 2.5$.
Soit $M = \text{Max} \left\{ \sum_{k \in \gamma^+(W)} \omega_k, \sum_{k \in \gamma^-(W)} \omega_k \right\} - |\delta_{G_i}(W)| \times 2.5$. M représente la capacité manquante sur la coupe $\delta_{G_i}(W)$.

- Si $M \leq 0$ (i.e la demande intersectant la coupe est inférieure à la capacité disponible sur les arêtes de petite capacité), les arêtes de capacité 2.5 Gbit/s suffisent à elles-seules pour écouler le flot. La contrainte (5.51), qui s'écrit alors $x^2(\delta_{G_i}(W)) \geq 0$, est clairement valide pour MCSIPND$_s(G, \mathcal{F}, K)$.
- Si $M > 0$, cela signifie que les arêtes de petite taille ne suffisent pas pour écouler le flot, même si elles sont toutes installées. Il faut donc installer un certain nombre d'arêtes de grande taille. Soit $f \in E^1$ une arête sur laquelle on installe une grande capacité au lieu d'une petite. En effet, comme $x^1(e) + x^2(e) \leq 1$ pour tout $e \in E^1$,

poser $x^2(f) = 1$ implique que $x^1(f) = 0$. Donc installer une grande capacité sur l'arête f permet d'ajouter une capacité disponible de 10-2.5=7.5 Gbit/s sur la coupe. La quantité $\left\lceil \frac{\text{Max } \{\sum_{k\in\gamma^+(W)} \omega_k, \sum_{k\in\gamma^-(W)} \omega_k\} - |\delta_{G_i}(W)|\times 2.5}{7.5} \right\rceil$ représente donc le nombre d'arêtes de grande taille minumum à installer dans la coupe $\delta_{G_i}(W)$. La contrainte (5.51) est donc valide pour MCSIPND$_s(G, \mathcal{F}, K)$.

□

5.5.3 Contraintes de coupe-cycle

Les contraintes introduites dans cette section sont basées sur une configuration de coupe-cycle telle que définie dans la section 3.3.3 (page 59). Une configuration de coupe-cycle (W, T_1, T_2) est définie dans un graphe $G = (V, E)$ par une coupe $\delta(W)$ et deux ensembles d'arêtes de la coupe $\delta_G(W)$, $T_1 = \{e_1, \ldots, e_s\}$, $s \geq 3$ et T_2 tels qu'il existe $1 \leq q < s$ un entier tel que pour tout $i = 1, \ldots, s$, il existe $j_i \in \{1, \ldots t\}$ tel que $F_{j_i} \cap T_1 = \{e_i, \ldots, e_{i+q-1}\}$ (les indices sont modulo s) et tels que $T_2 = \delta_G(W) \backslash (T_1 \cup (\bigcap_{i=1,\ldots,s} F_{j_i}))$.

5.5.3.1 Contraintes de coupe-cycle topologiques

Théorème 5.15 *Soit (W, T_1, T_2) une configuration de coupe-cycle telle que $\gamma(W) \neq \emptyset$. Pour tout $e \in \delta_G(W)$, on pose $r_e = |\{i \in \{1, \ldots, s\} \mid e \in \delta_G(W) \backslash F_{j_i}\}|$, et soit r le plus petit entier tel que $r(s-q) \geq \max_{e\in T_2}\{r_e\}$. Alors l'inégalité*

$$\sum_{l=1,2} (x^l(T_1) + r x^l(T_2)) \geq \left\lceil \frac{s}{s-q} \right\rceil \tag{5.52}$$

est valide pour MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$.

Preuve. Les inégalités suivantes sont valides pour MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$.

$$\sum_{l=1,2} x^l(\delta_{G_{j_i}}(W)) \geq 1 \qquad \text{pour } i = 1, \ldots, s,$$

$$(r(s-q) - r_e) x^l(e) \geq 0 \qquad \text{pour tout } e \in T_2, l = 1, 2.$$

En sommant ces inégalités, on obtient

$$\sum_{l=1,2} ((s-q) x^l(T_1) + r(s-q) x^l(T_2)) \geq s.$$

En divisant cette inégalité par $s-q$ et en arrondissant le membre de droite à l'entier supérieur, on obtient l'inégalité (5.52). □

Les contraintes de type (5.52) seront appelées *contraintes de coupe-cycle topologiques.*

5.5.3.2 Contraintes de coupe-cycle avec capacités

Théorème 5.16 *Soit (W, T_1, T_2) une configuration de coupe-cycle. Pour tout $e \in \delta_G(W)$, on pose $r_e = |\{i \in \{1,\dots,s\} \mid e \in \delta_G(W) \backslash F_{j_i}\}|$, et soit r le plus petit entier tel que $r(s-q) \geq \max_{e\in T_2}\{r_e\}$. Soit $r_W = \begin{cases} 4 \text{ si } D_W \bmod 4 = 0 \\ D_W \bmod 4 \text{ sinon.} \end{cases}$*

Alors, l'inégalité

$$x^1(T_1) + r_W x^2(T_1) + r x^1(T_2) + r \times r_W x^2(T_2) \geq \left\lceil \frac{s}{s-q} \times r_W \left\lceil \frac{D_W}{4} \right\rceil \right\rceil \qquad (5.53)$$

est valide pour MCSIPND$_m$$(G, \mathcal{F}, K)$ et MCSIPND$_s$$(G, \mathcal{F}, K)$.

Preuve. Par le théorème 5.13, les contraintes

$$\sum_{e\in\delta_{G_{j_i}}(W)} (x_e^1 + r_W x_e^2) \geq r_W \left\lceil \frac{D_W}{4} \right\rceil \qquad \text{pour } i = 1,\dots,s,$$

sont valides pour MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$. Considérons également les inégalités suivantes qui sont aussi valides pour MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$.

$$(r(s-q) - r_e)x^1(e) \geq 0 \qquad \text{pour tout } e \in T_2,$$
$$(r(s-q) - r_e)r_W x^2(e) \geq 0 \qquad \text{pour tout } e \in T_2.$$

En sommant toutes ces inégalités, on obtient

$$\sum_{e\in T_1} \left[(s-q)x_e^1 + (s-q)r_W x_e^2\right] + \sum_{e\in T_2} \left[r(s-q)x_e^1 + r(s-q)r_W x_e^2\right] \geq s \times r_W \left\lceil \frac{D_W}{4} \right\rceil.$$

En divisant cette inégalité par $s-q$ et en arrondissant le membre de droite à l'entier supérieur, on obtient l'inégalité (5.53). □

Les contraintes de type (5.53) seront appelées *contraintes de coupe-cycle avec capacité.*

Remarque 5.17 Dans le cas où $r_W = 1$ et $\left\lceil \frac{D_W}{4} \right\rceil = 1$, on obtient alors la contrainte de coupe-cycle topologique (5.52).

5.5.3.3 Contraintes de coupe-cycle généralisées

Théorème 5.18 *Soit (W, T_1, T_2) une configuration de coupe-cycle telle que $\gamma(W) \neq \emptyset$. Soit $\sum_{l=1,2} \alpha_i^l x^l(\delta_{G\setminus F_{j_i}}(W)) \geq \beta_i$ une contrainte de coupe topologique (5.34) ou de coupe avec capacités (5.36), (5.38), (5.39), (5.40) associée à $\delta_{G\setminus F_{j_i}}(W)$, pour $i = 1, \ldots, s$. Notons*

$$r_e^l = \sum_{\substack{i=1,\ldots,s\ :\\ e \in \delta_{G\setminus F_{j_i}}(W)}} \alpha_i^l$$

1) Supposons que les entiers r_e^l, $l = 1, 2$, $e \in T_1$ ne sont pas premiers entre eux. Soit R leur PGCD. Soit r^l, $l = 1, 2$, le plus petit entier tel que $r^l R \geq \max_{e\in T_2}\{r_e^l\}$. Alors l'inégalité

$$\sum_{e\in T_1}\left(\frac{r_e^1}{R}x_e^1 + \frac{r_e^2}{R}x_e^2\right) + \sum_{e\in T_2}(r^1 x_e^1 + r^2 x_e^2) \geq \left\lceil \frac{\sum_{i=1}^s \beta_i}{R} \right\rceil \tag{5.54}$$

est valide pour $MCSIPND_m(G, \mathcal{F}, K)$ et $MCSIPND_s(G, \mathcal{F}, K)$.

2) Supposons que les entiers r_e^l, $l = 1, 2$, $e \in T_1$ sont premiers entre eux. Soit

$$\lambda_e^l = \begin{cases} r_e^l & \text{si } r_e^l \text{ est pair,} \\ r_e^l + 1 & \text{sinon ,} \end{cases} \qquad l = 1, 2,\ e \in T_1 \cup T_2.$$

Soit R' le PGCD des entiers λ_e^l, $l = 1, 2$, $e \in T_1$. Soit λ^l, $l = 1, 2$, le plus petit entier tel que $\lambda^l R' \geq \max_{e\in T_2}\{\lambda_e^l\}$. Alors l'inégalité

$$\sum_{e\in T_1}\left(\frac{\lambda_e^1}{R'}x_e^1 + \frac{\lambda_e^2}{R'}x_e^2\right) + \sum_{e\in T_2}(\lambda^1 x_e^1 + \lambda^2 x_e^2) \geq \left\lceil \frac{\sum_{i=1}^s \beta_i}{R'} \right\rceil \tag{5.55}$$

est valide pour $MCSIPND_m(G, \mathcal{F}, K)$ et $MCSIPND_s(G, \mathcal{F}, K)$.

Preuve. 1) En sommant les inégalités

$$\sum_{e\in\delta_{G\setminus F_{j_i}}(W)} (\alpha_i^1 x_e^1 + \alpha_i^2 x_e^2) \geq \beta_i \qquad \text{pour } i = 1, \ldots, s,$$

$$(r^l R - r_e^l)x_e^l \geq 0 \qquad \text{pour tout } e \in T_2 \text{ et } l = 1, 2,$$

on obtient

$$\sum_{e\in T_1}(r_e^1 x_e^1 + r_e^2 x_e^2) + \sum_{e\in T_2}(r^1 R x_e^1 + r^2 R x_e^2) \geq \sum_{i=1}^s \beta_i.$$

En divisant par R et en arrondissant le membre de droite à l'entier supérieur, on obtient l'inégalité

$$\sum_{e\in T_1}\left(\frac{r_e^1}{R}x_e^1+\frac{r_e^2}{R}x_e^2\right)+\sum_{e\in T_2}(r^1x_e^1+r^2x_e^2)\geq\left\lceil\frac{\sum_{i=1}^s\beta_i}{R}\right\rceil.$$

Ce qui montre la validité de (5.54).

2) En sommant les inégalités

$$\begin{aligned}\sum_{e\in\delta_{G\setminus F_{j_i}}(W)}(\alpha_i^1x^1+\alpha_i^2x^2)\geq\beta_i \qquad & \text{pour tout } i=1,\ldots,s,\\ x_e^l\geq 0 \qquad & \text{pour } l=1,2, \text{pour tout } e\in T_1\\ & \text{tel que } \lambda_e^l=r_e^l+1,\\ (\lambda^lR'-\lambda_e^l)x_e^l\geq 0 \qquad & \text{pour tout } e\in T_2,\ l=1,2,\end{aligned}$$

on obtient

$$\sum_{e\in T_1}(\lambda_e^1x_e^1+\lambda_e^2x_e^2)+\sum_{e\in T_2}(r^1R'x_e^1+r^2R'x_e^2)\geq\sum_{i=1}^s\beta_i.$$

En divisant par R' et en arrondissant le membre de droite à l'entier supérieur, on obtient l'inégalité

$$\sum_{e\in T_1}\left(\frac{\lambda_e^1}{R'}x_e^1+\frac{\lambda_e^2}{R'}x_e^2\right)+\sum_{e\in T_2}(r^1x_e^1+r^2x_e^2)\geq\left\lceil\frac{\sum_{i=1}^s\beta_i}{R'}\right\rceil.$$

Ce qui montre la validité de (5.55).

□

Les contraintes de types (5.54) et (5.55) seront appelées *contraintes de coupe-cycle généralisées.*

Remarque 5.19 Dans le cas où les contraintes de coupe sont toutes des contraintes de coupe topologiques (*i.e.* $\alpha_i^1=\alpha_i^2=\beta_i=1$ pour tout $i=1,\ldots,p$), on a $r_e^l=s-q$ pour tout $l=1,2$ et $e\in T_1$. Il s'ensuit que $R=PGCD(r_e^l,l=1,2,e\in T_1)=s-q$ et r^l est le plus petit entier tel que $r^l(s-q)\geq\max_{e\in T_2}\{r_e^l\}$. Comme de plus, $\beta_i=1$, on a $\sum_{i=1}^s\beta_i=s$. On obtient alors l'inégalié

$$\sum_{e\in T_1}\left(\frac{s-q}{s-q}x_e^1+\frac{s-q}{s-q}x_e^2\right)+\sum_{e\in T_2}(r^1x_e^1+r^2x_e^2)\geq\left\lceil\frac{s}{s-q}\right\rceil.$$

Comme $r^1=r^2$, on retrouve bien la contrainte de coupe-cycle topologique (3.11).

Ces contraintes ne seront pas considérées dans la partie expérimentale. En effet, les contraintes précédentes nous ont semblé suffisantes et sont plus faciles à mettre en œuvre.

5.5.4 Contraintes d'étoile-partition

Les contraintes introduites dans cette section sont basées sur une configuration d'étoile-partition définie dans la section 5.5.4 (page 66). Une configuration d'étoile-partition $(V_0, V_1, \ldots, V_p, F)$ est définie pour un graphe $G = (V, E)$ et $\mathcal{F} = \{F_1, \ldots, F_t\}$, $t \geq 2$ une famille de sous-ensembles d'arêtes de E, par une partition $V_0, V_1, \ldots, V_p$ de V et un ensemble d'arêtes F entre les éléments de la partition tels que pour tout $i = 1, \ldots, p$, il existe $j_i \in \{1, \ldots, t\}$ tel que $F_{j_i} \cap \delta_G(V_i, V_0) \neq \emptyset$, $\Lambda = \{e \in E \,|\, e \in \delta_G(V_k, V_l) \cap F_{j_k} \cap F_{j_l}$, pour tout $k, l \in \{1, \ldots, p\}\}$ et $F = \bigcup_{i=1}^{p} (F_{j_i} \cap \delta_G(V_i, V_0)) \cup \Lambda$.

5.5.4.1 Contraintes d'étoile-partition topologiques

Théorème 5.20 *Soit une configuration d'étoile-partition avec p impair telle que $\gamma(W) \neq \emptyset$. L'inégalité*

$$\sum_{l=1,2} x^l(\delta_G(V_0, \ldots, V_p) \backslash F) \geq \left\lceil \frac{p}{2} \right\rceil \tag{5.56}$$

est valide pour MCSIPND$_m$$(G, \mathcal{F}, K)$ et MCSIPND$_s$$(G, \mathcal{F}, K)$.

Preuve. Il est clair que les inégalités suivantes sont valides pour MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$.

$$\begin{aligned}
\sum_{l=1,2} x^l(\delta_{G \backslash F_{j_i}}(V_i)) \geq 1 \qquad & \text{pour } i = 1, \ldots, p, \\
x^l(e) \geq 0 \qquad & \text{pour tout } e \in \delta(V_0) \backslash F,\ l = 1, 2, \\
x^l(e) \geq 0 \qquad & \text{pour tout } e \in (\delta_G(V_k, V_m) \cap F_{j_k}) \backslash F_{j_m}, \\
& k = 1, \ldots, p,\ m = 1, \ldots, p,\ k \neq m, \quad l = 1, 2.
\end{aligned}$$

En sommant ces inégalités, on obtient l'inégalité

$$2 \sum_{l=1,2} x^l(\delta(V_0, \ldots, V_p) \backslash F) \geq p.$$

En divisant par 2 et en arrondissant le membre de droite à l'entier supérieur, on obtient l'inégalité (5.56). □

Les contraintes de type (5.56) seront appelées *contraintes d'étoile-partition topologiques.*

5.5.4.2 Contraintes d'étoile-partition avec capacités

Théorème 5.21 *Soit $(V_0, V_1, \ldots, V_p, F)$ une configuration d'étoile-partition.*
Soit $r_{V_i} = \begin{cases} 4 \text{ si } D_{V_i} \bmod 4 = 0, \\ D_{V_i} \bmod 4 \text{ sinon,} \end{cases}$ *pour* $i = 1, \ldots, p$.
Notons

$$r_e = \sum_{\substack{i=1,\ldots,p\ :\\ e \in \delta_{G\setminus F_{j_i}}(V_i)}} r_{V_i} \quad et \quad \lambda_e = \begin{cases} r_e & \text{si } r_e \text{ est pair,} \\ r_e + 1 & \text{sinon ,} \end{cases} \quad e \in \delta_G(V_0, \ldots, V_p)\setminus F.$$

Alors l'inégalité

$$\sum_{e \in \delta_G(V_0,\ldots,V_p)\setminus F} \left(x_e^1 + \frac{\lambda_e}{2} x_e^2 \right) \geq \left\lceil \frac{\sum_{i=1}^{p} r_{V_i} \left\lceil \frac{D_{V_i}}{4} \right\rceil}{2} \right\rceil \tag{5.57}$$

est valide pour MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$.

Preuve. Les inégalités suivantes sont valides pour MCSIPND$_m(G, \mathcal{F}, K)$ et MCSIPND$_s(G, \mathcal{F}, K)$.

$$\begin{aligned}
\sum_{e \in \delta_{G\setminus F_{j_i}}(V_i)} (x_e^1 + r_{V_i} x_e^2) \geq r_{V_i} \left\lceil \frac{D_{V_i}}{4} \right\rceil \quad & \text{pour } i = 1, \ldots, p, \\
x_e^1 \geq 0 \quad & \text{pour tout } e \in \delta_G(V_0, \ldots, V_p)\setminus F, \\
x_e^2 \geq 0 \quad & \text{pour } i = 1, \ldots, p, \\
& \text{pour tout } e \in \delta_G(V_i, V_0)\setminus F, \\
& \text{tel que } r_{V_i} \text{ est impair ,} \\
x_e^2 \geq 0 \quad & \text{pour tout } e \in \delta_G(V_k, V_m)\setminus(F_{j_k} \cap F_{j_m}), \\
& \text{tel que } r_e \text{ est impair ,} \\
& k = 1, \ldots, p,\ m = 1, \ldots, p,\ k \neq m, \\
x_e^1 \geq 0 \quad & \text{pour tout } e \in (\delta_G(V_k, V_m) \cap F_{j_k})\setminus F_{j_m}, \\
& k = 1, \ldots, p,\ m = 1, \ldots, p,\ k \neq m, \\
x_e^2 \geq 0 \quad & \text{pour tout } e \in (\delta_G(V_k, V_m) \cap F_{j_k})\setminus F_{j_m}, \\
& \text{tel que } r_{V_m} \text{ est impair ,} \\
& k = 1, \ldots, p,\ m = 1, \ldots, p,\ k \neq m.
\end{aligned} \tag{5.58}$$

En sommant ces inégalités, on obtient

$$\sum_{e\in\delta_G(V_0,\ldots,V_p)\backslash F}(2x_e^1+\lambda_e x_e^2)\geq\sum_{i=1}^{s}r_{V_i}\left\lceil\frac{D_{V_i}}{4}\right\rceil.$$

Puisque les λ_e, $e\in\delta_G(V_0,\ldots,V_p)\backslash F$ sont tous pairs, en divisant par 2 et en arrondissant le membre de droite à l'entier supérieur, on obtient l'inégalité

$$\sum_{e\in\delta_G(V_0,\ldots,V_p)\backslash F}\left(x_e^1+\frac{\lambda_e}{2}x_e^2\right)\geq\left\lceil\frac{\sum_{i=1}^{s}r_{V_i}\left\lceil\frac{D_{V_i}}{4}\right\rceil}{2}\right\rceil.$$

□

Les contraintes de type (5.57) seront appelées *contraintes d'étoile-partition avec capacité.*

Remarque 5.22 Dans le cas où $r_{V_i}=1$ et $\left\lceil\frac{D_{V_i}}{4}\right\rceil=1$ pour $i=1,\ldots,p$, on obtient la contrainte d'étoile-partition topologique (5.56).

5.5.4.3 Contraintes d'étoile-partition généralisées

Théorème 5.23 *Soit $(V_0,V_1,\ldots,V_p,F)$ une configuration d'étoile-partition. Soit $\sum_{l=1,2}\alpha_i^l x^l(\delta_{G\backslash F_{j_i}}(W))\geq\beta_i$ une contrainte de coupe topologique (5.34) ou de coupe avec capacités (5.36), (5.38), (5.39), (5.40) associée à $\delta_{G\backslash F_{j_i}}(W)$, $i=1,\ldots,s$. Notons*

$$r_e^l=\sum_{\substack{i=1,\ldots,s:\\ e\in\delta_{G\backslash F_{j_i}}(V_i)}}\alpha_i^l$$

et

$$\lambda_e^l=\begin{cases}r_e^l & \text{si } r_e^l \text{ est pair,}\\ r_e^l+1 & \text{sinon ,}\end{cases}\quad l=1,2, e\in\delta_G(V_0,\ldots,V_p)\backslash F.$$

Soit R le PGCD des entiers λ_e^l, $l=1,2$, $e\in\delta_G(V_0,\ldots,V_p)\backslash F$. Alors l'inégalité

$$\sum_{l=1,2}\sum_{e\in\delta_G(V_0,\ldots,V_p)\backslash F}\frac{\lambda_e^l}{R}x_e^l\geq\left\lceil\frac{\sum_{i=1}^{s}\beta_i}{R}\right\rceil \tag{5.59}$$

est valide pour $MCSIPND_m(G,\mathcal{F},K)$ et $MCSIPND_s(G,\mathcal{F},K)$.

Preuve. Les inégalités suivantes sont valides pour $\text{MCSIPND}_m(G, \mathcal{F}, K)$ et $\text{MCSIPND}_s(G, \mathcal{F}, K)$.

$$\sum_{e \in \delta_{G \setminus F_{j_i}}(V_i)} (\alpha_i^1 x^1 + \alpha_i^2 x^2) \geq \beta_i \qquad \text{pour } i = 1, \ldots, s,$$

$$x_e^l \geq 0 \qquad \text{pour tout } l = 1, 2, \text{ et } e \in \delta_G(V_0, \ldots, V_p) \setminus F, \text{ tels que } \lambda_e^l = r_e^l + 1.$$

En sommant ces inégalités, on obtient

$$\sum_{e \in \delta_G(V_0, \ldots, V_p) \setminus F} (\lambda_e^1 x_e^1 + \lambda_e^2 x_e^2) \geq \sum_{i=1}^{s} \beta_i.$$

Puisque les λ_e^l, $l = 1, 2$, $e \in \delta_G(V_0, \ldots, V_p) \setminus F$ sont tous pairs, $R \geq 2$. En divisant par R et en arrondissant le membre de droite à l'entier supérieur, on obtient l'inégalité

$$\sum_{e \in \delta_G(V_0, \ldots, V_p) \setminus F} \left(\frac{\lambda_e^1}{R} x_e^1 + \frac{\lambda_e^2}{R} x_e^2 \right) \geq \left\lceil \frac{\sum_{i=1}^{s} \beta_i}{R} \right\rceil.$$

□

Les contraintes de type (5.59) seront appelées *contraintes d'étoile-partition généralisées.*

Remarque 5.24 Dans le cas où les contraintes de coupe sont toutes des contraintes de coupe topologiques (*i.e.* $\alpha_i^1 = \alpha_i^2 = \beta_i = 1$ pour $i = 1, \ldots, p$), on a $\lambda_e^l = 2$ car $\sum_{i=1,\ldots,s \mid e \in \delta_{G \setminus F_{j_i}}(V_i)} \alpha_i^l = 2$ pour tout $e \in \delta_G(V_0, \ldots, V_p) \setminus F$). Donc $R = 2$. De plus, $\sum_{i=1}^{s} \beta_i = p$ car $\beta_i = 1$ pour $i = 1, \ldots, p$.
On a alors la contrainte

$$\sum_{l=1,2} \sum_{e \in \delta_G(V_0, \ldots, V_p) \setminus F} \frac{2}{2} x_e^l \geq \left\lceil \frac{p}{2} \right\rceil.$$

On retrouve bien la contrainte d'étoile-partition topologique (5.56).

Comme dans le cas des contraintes de coupe-cycle, seules les contraintes d'étoile-partition avec capacités seront utilisées dans la partie expérimentale. En effet, les contraintes d'étoile-partition généralisées sont plus difficiles à implémenter et les contraintes d'étoile-partition avec capacités semblent suffire.

Chapitre 6

Algorithmes de résolution et résultats expérimentaux pour le problème MCSIPND

Dans ce chapitre, nous présentons des algorithmes pour résoudre le problème de sécurisation multicouche avec capacités du réseau IP sous ses deux versions (avec ou sans liaisons parallèles). Nous décrivons deux algorihtmes de coupes et branchements basés sur les formulations arcs-sommets, et deux algorithmes de coupes, génération de colonnes et branchements basé sur les formulations arcs-chemins. Ces algorithmes utilisant les résultats du chapitre précédent. Ce travail a pour but de comparer l'approche classique de la formulation arcs-sommets et l'approche par génération de colonnes de la formulation arcs-chemins. Il permet également de montrer l'efficacité des contraintes valides utilisées. Dans ce chapitre, nous présentons également des résultats expérimentaux obtenus à l'aide de ces algorithmes pour des instances aussi bien réelles qu'aléatoires.

6.1 Algorithmes de résolution

Supposons que l'on a deux graphes $G^1 = (V^1, E^1)$ et $G^2 = (V^2, E^2)$, représentant respectivement la couche IP et la coupe optique du réseau et supposons également que l'on dispose d'une famille $\mathcal{F} = \{F_e \subseteq E^1, e \in E^2\}$ d'ensembles d'arêtes de E^1. Chaque ensemble F_e est associé à une arête e de G^2. En effet, F_e est l'ensemble d'arêtes de E^1 affectées par la panne de l'arête e de E^2. Considérons aussi un ensemble de demandes

K dont on connait pour chaque demande le sommet origine, le sommet destination et le volume. Pour respecter les notations du chapitre précédent, on note les éléments de $\mathcal{F}$ par $F_1, \ldots, F_t$, où $t = |E^2|$ et $G^1 = (V^1, E^1)$ par $G = (V, E)$. Un vecteur poids associé aux arêtes $c \in I\!R^{2|E|}$ est également connu. Il représente le coût d'installation d'une liaison de capacité 2.5 Gbit/s ou 10 Gbit/s. Les données sont identiques pour les deux versions du problème que ce soit le cas multiple ou le cas simple. Dans la suite, nous décrivons ces algorithmes.

6.1.1 Algorithmes de coupes et branchements

Etant donné que les formulations arcs-sommets comportent un nombre polynomial de variables comme de contraintes, les algorithmes de coupes et branchements que nous allons décrire ci-après utilisent la relaxation linéaire du problème comme programme linéaire permettant de débuter l'optimisation. Ces programmes linéaires contiennent donc $2|E| \times (1+|K|+|\mathcal{F}|)$ variables et $|V| \times |K| \times |\mathcal{F}|$ contraintes de conservation de flot, $2|E| \times |\mathcal{F}|$ contraintes de capacités et $2|E| \times (1+|K|+|\mathcal{F}|)$ contraintes triviales. Pour la version sans liaisons parallèles, on a en plus des contraintes déjà citées, $|E|$ contraintes imposant l'unicité des liaisons entre toute paire de sommets et $|E|$ contraintes triviales supplémentaires.

Nous avons essayé de commencer l'optimisation avec des programmes linéaires plus petits, comportant un nombre limité de contraintes, pour rajouter par la suite les contraintes nécessaires à la formulation dans une phase de génération de coupes, mais cette approche s'est vite avérée peu efficace. En effet, une grande partie du temps de calcul est consommé par la résolution de la relaxation linéaire.

En résolvant la relaxation linéaire du problème, si la restriction de la solution trouvée sur les variables de topologie est entière, alors la solution est optimale pour le problème. En général, la solution obtenue n'est pas réalisable pour le problème. Dans ce cas, on doit commencer une phase de génération de coupes. Celle-ci consiste à rechercher et à rajouter des inégalités violées par la solution courante. Le nouveau programme linéaire obtenu est de nouveau résolu. On itère ce processus jusqu'à ce que soit une solution réalisable est trouvée, soit aucune nouvelle contrainte violée ne peut être trouvée. Dans le premier cas, une solution optimale a été trouvée, et dans le deuxième, on commence une phase de branchement.

L'algorithme 6.1 illustre les principales étapes de nos algorithmes de coupes et branchements.

Algorithme 6.1 Algorithme de coupes et branchements

Données : un graphe $G = (V, E)$, une famille $\mathcal{F}$ d'ensembles d'arêtes, un ensemble de demandes K et un vecteur poids $c \in I\!R^{2|E|}$.
Sortie : solution optimale de $\text{Min}\{cx \mid (x, f) \in \text{MCSIPND}_m^{as}(G, \mathcal{F}, K)\}$ ou $\text{Min}\{cx \mid (x, f) \in \text{MCSIPND}_s^{as}(G, \mathcal{F}, K)\}$.

1 : PL $\leftarrow$ PL_{debut}
2 : Résoudre le programme linéaire PL.
Soit (x^*, f^*) la solution optimale de PL.
3 : **Si** (x^*, f^*) est réalisable pour MCSIPND(G,$\mathcal{F}$,K) **alors**
(x^*, f^*) est une solution optimale. STOP
4 : **Si** des contraintes violées par (x^*, f^*) sont trouvées **alors**
Les ajouter à LP.
Aller en 2.
5 : **Sinon**
Brancher sur une variable de topologie fractionnaire.
6 : Prendre la meilleure solution de tous les sous-problèmes.

Lors des phases de coupes, on considère les contraintes décrites dans le chapitre 5. Parmi celles-ci, on compte par exemple les contraintes de coupe avec capacités, les contraintes de capacités résiduelles, les contraintes de saturation, les contraintes de coupe-cycle topologiques, les contraintes de coupe-cycle avec capacités, les contraintes d'étoile-partition topologiques et les contraintes d'étoile-partition avec capacités. L'utilisation de ces contraintes ainsi que les algorithmes de séparation associés seront décrits dans la section 6.1.3.

6.1.2 Algorithmes de coupes, génération de colonnes et branchements

Les formulations arcs-chemins comportent un nombre très important de variables. Il est donc plus judiscieux de considérer une apporche par génération de colonnes pour résoudre ces modèles. Cette approche consiste tout d'abord à déterminer un ensemble de variables formant une base réalisable.

Pour le problème MCSIPND_m, déterminer ces variables de départ consiste à rechercher les plus courts chemins pour chaque panne $F_i \in \mathcal{F}$ et chaque demande $k \in K$. En effet, il suffit de dupliquer les arêtes présentes dans les plus courts chemins pour pouvoir écouler le flot. On obtient donc $|K| \times |\mathcal{F}|$ variables qui sont considérées dans

le programme linéaire initial. Ce dernier contient également $|K| \times |\mathcal{F}|$ contraintes de demande, $2|E| \times |\mathcal{F}|$ contraintes de capacité et $|K| \times |\mathcal{F}| + 2|E|$ contraintes triviales.

Il est plus difficile de déterminer un ensemble de variables formant une solution réalisable pour la relaxation linéaire du problème MCSIPND$_s$. En effet, le nombre de liaisons entre deux sommets étant limité à un, on ne peut pas utiliser simplement les plus courts chemins qui risquent de ne pas être suffisants pour écouler le flot. Comme il a été présenté dans la section 5.3.2.2, pour déterminer cet ensemble de variables, on utilise un algorithme de génération de colonnes pour résoudre le problème suivant :

$$
\begin{aligned}
& \text{Minimiser } \varepsilon \\
(SI_s) \quad & \sum_{P \in \mathcal{P}_k^i} y_k^i(P) = \omega_k && \forall k \in K,\ i = 1, \ldots, t, \\
& \sum_{k \in K} \sum_{P \in \mathcal{P}_k^i \,|\, (u,v) \in P} y_k^i(P) \leq 10 + \varepsilon && \forall (u,v) \in A^1 \backslash \overline{F}_i,\ i = 1, \ldots, t, \\
& y_k^i(P) \geq 0 && \forall k \in K,\ i = 1, \ldots, t,\ \forall P \in \mathcal{P}_k^i, \\
& \varepsilon \geq 0.
\end{aligned}
$$

On suppose que l'on installe toutes les arêtes de capacité 10 et on considère la variable ε ainsi que les variables de flot sur les chemins. Ce programme linéaire est résolu à l'aide de l'algorithme 6.2 présenté ci-après.

Algorithme 6.2 Résolution du programme linéaire (SI_s)

Données : un graphe $G = (V, E)$, une famille $\mathcal{F}$ d'ensembles d'arêtes, un ensemble de demandes K.
Sortie : solution optimale de (SI_s).

1 : Déterminer l'ensemble de variables de départ.
2 : Créer le programme linéaire PL de départ.
3 : Résoudre le programme linéaire PL.
4 : Déterminer les variables duales.
5 : Résoudre le problème auxiliaire.
6 : **Si** tous les coûts réduits sont positifs **alors**
    Terminé, une solution optimale est trouvée.
  Sinon
    Ajouter les variables de coût réduit strictement négatif à PL.
7 : Aller en 3.

Les variables de départ formant une base réalisable sont déterminées de la même façon que celles utilisées dans la solution initiale du problème MCSIPND$_m$. Le problème

auxiliaire, qui consiste à rechercher des variables de coût réduit strictement négatif, se ramène à un problème de plus courts chemins présenté dans la section 5.3.2.2. Comme les poids associés à ce problème sont positifs, on utilise l'algorithme de Dijkstra pour le résoudre. La valeur optimale de ε représente la capacité manquante. Si $\varepsilon = 0$ alors il existe une solution réalisable pour le problème MCSIPND$_s$ et celle-ci peut être formée à l'aide des chemins de la solution optimale du programme linéaire (SI_s). Si par contre $\varepsilon \neq 0$, alors le problème MCSIPND$_s$ n'admet pas de solution réalisable.

En plus des variables formant la base réalisable de départ, nous allons générer des variables tout au long de l'algorithme à l'aide du problème auxiliaire. Pour conserver ces variables, nous avons créé des pools dont la taille augmente dynamiquement. Toutes les variables sont enlevées du programme linéaire courant quand elles ne sont pas actives. Nous évaluons tout d'abord les variables du pool avant de relancer la résolution du problème auxiliaire.

Algorithme 6.3 Algorithme de coupes, génération de colonnes et branchements

Données : un graphe $G = (V, E)$, une famille $\mathcal{F}$ d'ensembles d'arêtes, un ensemble de demandes K et un vecteur poids $c \in I\!R^{2|E|}$.
Sortie : solution optimale de $\text{Min}\{cx \mid (x, y) \in \text{MCSIPND}_m^{ac}(G, \mathcal{F}, K)\}$ ou $\text{Min}\{cx \mid (x, y) \in \text{MCSIPND}_s^{ac}(G, \mathcal{F}, K)\}$.
1 : PL $\leftarrow$ PL$_{debut}$
2 : Résoudre le programme linéaire PL.
Soit (x^*, y^*) la solution optimale de PL.
3 : **Si** des contraintes violées par (x^*, y^*) sont trouvées **alors**
Les ajouter à LP.
Aller en 2.
4 : **Sinon**
Déterminer les variables duales.
Résoudre le problème auxiliaire.
Si tous les coûts réduits sont positifs **alors**
Aller en 5.
Sinon
Ajouter les variables de coût réduit strictement négatif à PL.
Aller en 2.
5 : **Si** (x^*, y^*) est réalisable pour MCSIPND(G,$\mathcal{F}$,K) **alors**
(x^*, y^*) est une solution optimale. STOP
6 : **Sinon**
Brancher sur une variable fractionnaire.
7 : Prendre la meilleure solution de tous les sous-problèmes.

La résolution par la génération de colonnes de la relaxation linéaire du problème nous fournit une solution qui n'est en général pas réalisable pour le problème. Pour palier à ce problème, nous avons combiné à la fois des phases de génération de coupes, des phases de génération de variables ainsi que des phases de branchement pour résoudre le problème. L'articulation de ces différentes étapes est décrite par l'algorithme 6.3. La résolution du problème auxiliaire est présentée dans la section 5.3.2.3. Elle est basée sur une succession de problèmes de plus courts chemins dans des graphes avec des coûts positifs. On utilise donc également l'algorithme de Dijkstra pour résoudre ce problème.

Les quatre algorithmes que nous venons de présenter comportent tous une phase de génération de coupes. La section suivante décrit les différentes procédures de séparation pour les classes de contraintes utilisées ainsi que la manière dont elles sont gérées au cours de l'algorithme.

6.1.3 Séparation des contraintes valides

Les contraintes définies au chapitre 5 ne sont pas forcément valides pour les quatre polytopes associés aux deux formulations pour les deux versions du problème. Par exemple, les contraintes de capacité résiduelle sont valides pour MCSIPND$_m^{as}$ et MCSIPND$_s^{as}$. Les contraintes de saturation sont valides pour MCSIPND$_s^{as}$ et MCSIPND$_s^{ac}$. Par contre, les contraintes de coupe avec capacités sont valides pour les quatre polytopes MCSIPND$_m^{as}$, MCSIPND$_m^{ac}$, MCSIPND$_s^{as}$ et MCSIPND$_s^{ac}$.

La séparation de ces classes d'inégalités valides s'effectue dans l'ordre suivant :

1. contraintes de coupe avec capacités
2. contraintes de capacité résiduelle (uniquement pour les algorithmes basés sur les formulations arcs-sommets)
3. contraintes de saturation (uniquement pour le cas simple)
4. contraintes de coupe-cycle topologiques
5. contraintes de coupe-cycle avec capacités
6. contraintes d'étoile-partition topologiques
7. contraintes d'étoile-partition avec capacités

Plusieurs contraintes peuvent être ajoutées à chaque itération. Nous avons choisi de limiter le nombre de contraintes générées à une même itération à 200. Ces inégalités sont globales. En effet, elles sont valides pour tout l'arbre de branchement. Néanmoins,

nous avons choisi de les gérer de façon dynamique, c'est-à-dire qu'elles sont supprimées du programme linéaire courant quand elles ne sont pas actives. Ces contraintes sont conservées dans des pools dont la taille augmente dynamiquement. Les inégalités du pool sont séparées avant de lancer les procédures de séparation sur les différentes classes de contraintes dans l'ordre donné ci-dessus. Pour les trois premières classes de contraintes, lors d'une même itération, on passe à la classe d'inégalités suivante seulement si l'on ne trouve pas d'inégalités violées. Les quatre dernières classes d'inégalités sont séparées au cours de la même itération, sans résoudre le nouveau programme linéaire entre leur séparation. Généralement, ces dernières contraintes apparaissent en nombre réduit.

Dans la suite, nous présentons les différentes procédures de séparation que nous avons implémentées dans nos quatre algorithmes. Tous nos algorithmes sont appliqués sur le graphe $G_{(\bar{x}^1,\bar{x}^2)} = (V, E_{(\bar{x}^1,\bar{x}^2)})$ où $(\bar{x}^1, \bar{x}^2)$ est la solution courante de la relaxation linéaire et $E_{(\bar{x}^1,\bar{x}^2)}$ contient toutes les arêtes uv de E pour lesquels $\bar{x}^1_{uv} + \bar{x}^2_{uv} \neq 0$. On note également $\bar{f}$ la solution courante de la relaxation linéaire pour les variables de flot en termes d'arcs.

6.1.3.1 Séparation des contraintes de coupe

Pour la séparation des contraintes de coupe avec capacités, nous utilisons tout d'abord, une heuristique qui consiste à considérer dans un premier temps les coupes sur les sommets, puis à contracter successivement les arêtes ayant la plus grande valeur selon $2.5\bar{x}^1_{uv} + 10\bar{x}^2_{uv} - \sum_{k\in\gamma(\{u\})\cup\gamma(\{v\})} \omega_k$ jusqu'à ce que l'on obtienne un graphe sur deux sommets. A chaque itération, on vérifie si la coupe associée au sommet formé par contraction de l'arête est violée par la solution $(\bar{x}^1, \bar{x}^2)$, qu'elle soit de type (5.36), (5.38), (5.39) ou (5.40).

Lorsque cette heuristique ne permet plus de trouver des inégalités violées, nous séparons d'une manière heuristique les contraintes de coupes avec capacités de type (5.38), qui dominent les contraintes de coupe topologiques (5.34). Pour ce faire, on détermine l'arbre de Gomory-Hu [39] du graphe $G_{(\bar{x}^1,\bar{x}^2)}$ en considérant comme poids $\bar{x}^1_{uv} + \bar{x}^2_{uv}$. Pour générer cet arbre, nous utilisons l'implémentation de Gusfield [47], demandant $|V| - 1$ flots maximums. Le calcul de chaque flot maximum est effectué à l'aide de l'algorithme de Goldberg et Tarjan [38]. Il ne reste plus qu'à calculer le membre de droite dépendant des demandes intersectant les coupes et à vérifier si les coupes ainsi trouvées sont violées.

6.1.3.2 Séparation des contraintes de saturation

Lors de la séparation des contraintes de saturation, on considère les coupes sur les sommets $\delta_{G_{(\bar{x}^1,\bar{x}^2)}}(v)$, $v \in V$. On vérifie si celles-ci sont violées et on les rajoute le cas échéant. Nous ne considérons pas les autres coupes du graphe $G_{(\bar{x}^1,\bar{x}^2)}$. En effet, nous avons remarqué en pratique que les contraintes de saturation violées sont pour la plupart des contraintes associées à une coupe sur un seul sommet.

6.1.3.3 Séparation des contraintes de capacité résiduelle

Les contraintes de capacité résiduelle (5.8) sont définies pour toute arête uv et pour toute panne e. On peut ainsi décomposer le problème de séparation sur chaque arête uv pour chaque panne e. Les contraintes de capacité résiduelle sont redondantes ou dominées dans le cas où $2x^2_{uv} \geq \mu_L$ ou $2x^2_{uv} \leq \mu_L - 1$ mais renforcent la relaxation linéaire pour tout $\mu_L - 1 < 2x^2_{uv} < \mu_L$. En utilisant les résultats de Atamtürk et Rajan [5], on montre que la séparation des contraintes de capacité résiduelle revient à définir l'ensemble $T = \{k \in K \mid \bar{f}^{k,e}_{uv} + \bar{f}^{k,e}_{vu} > \frac{1}{2.5}\omega_k(2\bar{x}^2_{uv} - \lfloor 2\bar{x}^2_{uv} \rfloor)\}$ et à vérifier si la contrainte de capacité résiduelle associée à uv est violée pour l'ensemble T. Nous utilisons cette procédure pour séparer les contraintes de capacité résiduelle. Cette séparation a une complexité linéaire.

6.1.3.4 Séparation des contraintes de coupe-cycle

Comme pour la séparation des contraintes de coupe-cycle définies pour le problème de sécurisation multicouche du réseau IP (voir section 4.1.5), nous séparons les contraintes de coupe-cycle topologiques (5.52) et les contraintes de coupe-cycle avec capacités (5.53) avec $q = 1$. En effet, nous avons remarqué que les contraintes de coupe-cycle violées que ce soit des contraintes de coupe-cycle topologiques ou des contraintes de coupe-cycle avec capacités, trouvées lors de notre résolution sont le plus souvent de ce type. La procédure de séparation des contraintes de coupe-cycle topologiques consiste à calculer l'arbre de Gomory-Hu du graphe $G_{(\bar{x}^1,\bar{x}^2)}$ ayant comme poids pour chaque arête uv, la somme $\bar{x}^1_{uv} + \bar{x}^2_{uv}$. Puis pour chaque coupe donnée par l'arbre de Gomory-Hu, ayant une valeur strictement inférieure à 2, de vérifier si la coupe intersecte au moins une demande. Si c'est le cas, cette coupe donne une inégalité de type (5.52) violée par $(\bar{x}^1, \bar{x}^2)$. On détermine alors les ensembles T_1 et T_2 de telle sorte que T_1 soit maximal en utilisant l'algorithme 4.2 de la page 90. Comme l'algorithme de Gomory-Hu a une compexité élevée, on considère tout d'abord les coupes sur les sommets $\delta_{G_{(\bar{x}^1,\bar{x}^2)}}(v)$, $v \in V$.

Pour la séparation des contraintes de coupe-cycle avec capacités, on considère également en premier lieu les coupes sur les sommets $\delta_{G_{(\bar{x}^1,\bar{x}^2)}}(w)$, $w \in V$. On vérifie ensuite si la contrainte associée à la coupe est violée après avoir évalué le membre de droite. Lorsqu'une contrainte violée a été détectée, on détermine les ensembles T_1 et T_2 de la même façon que pour la séparation des contraintes de coupe-cycle topologiques. Dans le but d'améliorer cette procédure, on sélectionne ensuite l'arête uv ayant la plus grande valeur pour $2.5\bar{x}^1_{uv}+10\bar{x}^2_{uv}-\sum_{k\in\gamma(\{u\})\cup\gamma(\{v\})}\omega_k$ et on contracte cette arête. On considère alors le nouveau sommet ainsi formé. On itère ce processus jusqu'à obtenir un graphe sur deux sommets. Les contraintes de coupe-cycle généralisées (5.54) et (5.55) ne sont pas considérées. En effet, les contraintes précédentes nous ont semblé suffisantes et de plus elles sont plus faciles à mettre en œuvre.

6.1.3.5 Séparation des contraintes d'étoile-partition

Pour la séparation des contraintes d'étoile-partition topologiques (5.56) et des contraintes d'étoile-partition avec capacités (5.57), nous commençons par déterminer dans le graphe $G_{(\bar{x}^1,\bar{x}^2)}$ un cycle (impair s'il s'agit de contraintes d'étoile-partition topologiques) formé d'arêtes fractionnaires . Puis pour chaque cycle détecté, on essaye de déterminer les ensembles F_{j_i}, $j_i \in \{1,\ldots,t\}$, $i = 1,\ldots,p$ parmi les arêtes de $\delta_G(v_i, V'\backslash\{v_1,\ldots,v_p\})$, permettant de définir l'ensemble F tels que la contrainte d'étoile-partition induite par $V'\backslash\{v_1,\ldots,v_p\},\{v_1\},\ldots,\{v_p\}$, et F soit violée par $(\bar{x}^1,\bar{x}^2)$.

Lorsque la phase de coupes ne permet plus de trouver des inégalités violées, et dans le cas des algorithmes de coupes, génération de colonnes et branchements, que le problème auxiliaire ne permet plus de rajouter des variables, et si bien sûr, la solution n'est pas réalisable, débute alors la phase de branchement. Dans la section suivante, nous décrivons la stratégie de branchement que l'on a considérée.

6.1.4 Branchement

Soit $(\mathcal{P})$ le programme mixte du nœud courant dans l'arbre de branchement. Supposons que la solution optimale de la relaxation linéaire de $(\mathcal{P})$ est fractionnaire. Soit $(\bar{x}^1,\bar{x}^2)$ cette solution fractionnaire. La phase de branchement consiste à choisir une variable fractionnaire $\bar{x}^l_{uv}$ avec $uv \in E$ et $l \in \{1,2\}$ et à créer deux sous-problèmes $(\mathcal{P}_1)$ et $(\mathcal{P}_2)$ à partir de $(\mathcal{P})$ en ajoutant respectivement les contraintes $x^l_{uv} \leq \lfloor \bar{x}^l_{uv} \rfloor$ et $x^l_{uv} \geq \lceil \bar{x}^l_{uv} \rceil$. Dans le cas de variables binaires (pour le problème MCSIPND$_s$), cela revient à fixer la variable x^l_{uv} soit à 0, soit à 1.

Il existe de nombreuses stratégies pour sélectionner la variable fractionnaire sur laquelle on veut brancher. Nous avons choisi d'utiliser la stratégie définie par Padberg et Rinaldi [69] pour le problème du voyageur de commerce symétrique. Cette stratégie consiste à choisir une variable dont la partie fractionnaire est proche de 0.5 de la manière suivante. Soit $(\bar{x}^1, \bar{x}^2)$ la solution optimale du dernier programme linéaire résolu et soit $(\bar{z}^1, \bar{z}^2)$ tel que $\bar{z}^l = \bar{x}^l \bmod 1$, $l = 1, 2$. Soit $L = \min\{0.5, \max\{\bar{z}^l_{uv} \,|\, \bar{z}^l_{uv} \leq 0.5, uv \in E, l = 1, 2\}\}$ et $H = \max\{0.5, \min\{\bar{z}^l_{uv} \,|\, \bar{z}^l_{uv} \geq 0.5, uv \in E, l = 1, 2\}\}$. Soit $C = \{(uv, l) \in E \times \{1, 2\} \,|\, 0.75L \leq \bar{z}^l_{uv} \leq H + 0.25(1 - H)\}$. Les variables pour lesquelles les indices sont dans C sont dites proches de 0.5. On choisit alors celle de ces variables qui a le coefficient le plus élevé dans la fonction objective.

Lors d'un branchement classique, la phase de branchement sélectionne une variable fractionnaire et crée deux sous-problèmes. Pour le problème MCSIPND$_s$, nous avons remarqué que pour chaque arête $uv \in E$, les variables x^1_{uv} et x^2_{uv} sont très liées. Une idée pourrait être de brancher simultanément sur ces deux variables. On crée alors trois sous-problèmes en ajoutant les contraintes $x^1_{uv} \leq 0$ et $x^2_{uv} \leq 0$, ou $x^1_{uv} \leq 0$ et $x^2_{uv} \geq 1$, ou $x^1_{uv} \geq 1$ et $x^2_{uv} \leq 0$. Comme les variables sont binaires, cela revient à fixer x^1_{uv} à 0 et x^2_{uv} à 0, ou bien x^1_{uv} à 0 et x^2_{uv} à 1, ou bien x^1_{uv} à 1 et x^2_{uv} à 0. On remarque que l'on ne peut pas fixer à 1 à la fois x^1_{uv} et x^2_{uv}. En effet, la contrainte d'unicité sur l'arête uv serait violée. Si l'on considère la taille de l'arbre de branchement dans le pire des cas, lors du branchement sur une ou sur deux variables, le nombre de nœuds dans l'arbre pour le deuxième cas est beaucoup moins élevé. En effet, les arbres sont, dans le pire des cas, soit une 2-arborescence saturée de hauteur $2|E|$ pour le branchement sur une variable, soit une 3-arborescence saturée de hauteur $|E|$ pour le branchement sur deux variables. On a donc respectivement $\frac{2^{2|E|+1}-1}{2-1}$ et $\frac{3^{|E|+1}-1}{3-1}$ nœuds dans l'arbre de branchement. Pour une instance à 5 sommets, on aurait dans le pire des cas, plus de 2 millions de nœuds avec le branchement sur une variable, pour à peine 90 000 nœuds avec le branchement sur deux variables. Si le nombre de nœuds est beaucoup moins important dans le pire des cas avec le branchement sur deux variables, il s'avère qu'en pratique le branchement sur une seule variable génère moins de nœuds en moyenne. Cela s'explique par la relation très grande qui existe entre les variables x^1_{uv} et x^2_{uv}. Nous avons donc utilisé un branchement sur une seule variable.

Un point important dans l'efficacité d'un algorithme de génération de coupes et branchement est le calcul d'une bonne borne supérieure. Nous avons donc développé une heuristique primale dans le but de pouvoir définir puis améliorer au cours du temps une borne supérieure réalisable pour le problème. Cette heuristique est présentée dans la suite.

6.1.5 Heuristique primale

Lors de l'exécution d'une heuristique primale, on essaie de transformer chaque solution de la relaxation linéaire en une solution réalisable (donc entière) pour le problème MCSIPND. Pour le problème MCSIPND$_m$, cette heuristique consiste à arrondir à l'entier supérieur toute valeur fractionnaire. Cette heuristique n'est pas utilisable pour le problème MCSIPND$_s$. En effet, la solution ainsi obtenue peut violer les contraintes d'unicité. Nous avons développé une heuristique primale pour le problème MCSIPND$_s$ qui peut être présentée de la façon suivante. Soit $\bar{x}$ une solution fractionnaire de la relaxation linéaire du problème MCSIPND$_s$ et $\tilde{x}$ la solution proposée par l'heuristique primale. Pour toute arête $uv \in E$, si $\bar{x}^1_{uv} + 4\bar{x}^2_{uv} \leq 1$, on fixe $\tilde{x}^1_{uv}$ à 1 et $\tilde{x}^2_{uv}$ à 0, sinon on fixe $\tilde{x}^1_{uv}$ à 0 et $\tilde{x}^2_{uv}$ à 1.

Nous avons implémenté les quatre algorithmes présentés ci-dessus pour résoudre le problème MCSIPND dans ses deux versions, simple ou multiple. Dans la suite, nous donnons quelques résultats expérimentaux obtenus à l'aide de ces algorithmes sur des instances aléatoires ainsi que sur des instances réelles fournies par France Télécom.

6.2 Résultats expérimentaux

6.2.1 Contexte informatique

Avant de présenter les différents résultats expérimentaux que nous avons obtenus, nous donnons un bref descriptif des logiciels et du matériel informatique que nous avons utilisés.

Les algorithmes de coupes et branchements et de coupes, génération de colonnes et branchements décrits précedemment ont été implémentés en langage C++ en utilisant le logiciel ABACUS (A Branch-And-CUt System), version 2.4 alpha [1, 30, 81] pour la gestion de l'arbre de branchement. Pour ce qui est de la résolution des programmes linéaires, ABACUS fait appel au logiciel CPLEX, version 9.0 [2], qui est une implémentation trés rapide de l'algorithme du simplexe. Ils ont été testés sur un Pentium IV cadencé à 2,4 Ghz avec 1024 Mo de mémoire vive, sous système Linux. Nous avons fixé un temps maximum d'exécution à 5 heures.

Dans la section suivante, nous allons présenter brièvement les instances sur lesquelles nous avons effectué nos tests.

6.2.2 Description des instances traitées

Les résultats qui sont présentés ici ont été obtenus à partir d'instances aléatoires, mais aussi d'instances provenant d'applications réelles. Pour toutes les instances, le graphe G^1, représentant le réseau IP, est considéré complet.

Instances aléatoires

Les instances aléatoires proviennent de la TSP Library [76]. Le graphe G^1 est obtenu en considérant un sous-ensemble de sommets d'instances de le TSP Library. Le graphe G^1 est complet. Le coût d'installation d'une arête est égal à la distance euclidienne entre ses deux extrémités. Les ensembles d'arêtes F_e sont générés aléatoirement ainsi que les demandes. Ces instances sont générées avec 6, 8 ou 10 sommets, $|\mathcal{F}|$ prend les valeurs 10 et 20 et l'ensemble K contient 5, 10 ou 20 demandes. Nous avons testé trois instances pour chaque taille, chaque $|\mathcal{F}|$ et chaque K. Nous considérons les moyennes des résultats obtenus pour ces trois instances.

Instances réelles

Les instances réelles sont extraites de réseaux opérationnels, et ont été fournies par France Télécom. Ces instances ont entre 6 et 18 sommets et $\mathcal{F}$ contient 11 à 32 ensembles d'arêtes. Le nombre de demandes varie entre 5 et 20. En fait, France Télécom fournit la topologie du réseau optique et le routage entre chaque paire de nœuds du réseau. A chaque arête f du réseau IP, on associe le chemin de routage dans le réseau optique entre les brasseurs correspondant aux routeurs IP extrémités de f. En utilisant ces chemins, nous avons généré $\mathcal{F} = \{F_e \subseteq E^1,\, e \in E^2\}$ où F_e est l'ensemble des arêtes f de E^1 telles que e appartient au chemin associé à f.

Pour ces instances, nous avons utilisé différentes fonctions coût qui ont un grand intérêt pour France Télécom. Ces fonctions coût sont similaires aux fonctions $c_1(.)$ et $c_2(.)$ définies pour les instances réelles du problème MSIPND. Le coût associé à chaque lien de la couche cliente est directement lié au chemin de routage correspondant dans le réseau de transport, et par conséquent, il dépend du coût de ce chemin. Le coût $c(f)$ d'un lien f dans le réseau IP est donné par

$$c(f) = c + \omega(f),$$

où c est un coût fixe représentant la construction de f dans le réseau IP et les équipements nécessaires à cette installation, et $\omega(f)$ est un coût dépendant du réseau optique.

L'installation d'un segment optique implique un coût fixe à chaque extrémité du segment. Donc, une première estimation possible du coût optique $\omega(f)$ est la somme

des coûts fixes associés aux segments optiques de P_f. Une bonne approche serait de considérer un coût $\omega(f)$ proportionnel au nombre de segments optiques de P_f. Par conséquent, nous avons défini une première fonction coût $\omega(f)$ qui consiste en le nombre de liens du chemin optique entre les brasseurs correspondant aux extrémités de l'arête f. Le coût $c(f)$ est donc donné dans ce cas par $c_1(f) = c + |P_f|$.

Le coût d'un lien optique dépend également de la longueur du lien. Dans ce cas, $c(f)$ peut être défini par

$$c_2(f) = c + \sum_{e \in P_f} l(e),$$

où $l(e)$ est la longueur de e. Pour nos expérimentations, nous avons considéré $l(e)$ comme étant la distance réelle en kilomètres entre les extrémités de e.

Les extrémités des demandes sont déterminées de façon aléatoires. Le volume de chaque demande est calculé à l'aide d'un modèle gravitaire utilisant la distance et la population des deux villes extrémités. En effet, le modèle gravitaire s'appuit sur les interactions spatiales. La formule générale du modèle gravitaire pour une demande (o_k, d_k, v_k) de K est $v_k = \frac{P_{o_k}^{\alpha} P_{d_k}^{\beta}}{d_{o_k,d_k}}$ où P_{o_k} et P_{d_k} sont les populations des villes d'origine et de destination respectivement et d_{o_k,d_k} représente la distance euclidienne séparant o_k et d_k. Nous avons fixé $\alpha = 1.2$ et $\beta = 0.8$ pour que les volumes de deux demandes entre les mêmes villes mais dans des sens opposés ne soient pas identiques. Nous avons ensuite conservé les 5, 10, 15 ou 20 demandes de plus grand volume pour créer les instances.

Nos résultats expérimentaux sont reportés dans les tables des sections suivantes. Les différentes colonnes de ces tables représentent :

$\|V^1\|$	:	le nombre de sommets de G^1,
$\|\mathcal{F}\|$	:	le nombre d'ensembles F_e,
$\|K\|$	:	le nombre de demandes,
VF	:	le nombre de variables de flot,
NC	:	le nombre de contraintes de coupe générées,
NCR	:	le nombre de contraintes de capacité résiduelle générées (uniquement pour les formulations arcs-sommets),
NS	:	le nombre de contraintes de saturation générées (uniquement pour le problème MCSIPND$_s$),
NCC	:	le nombre de contraintes de coupe-cycle générées,
NEP	:	le nombre de contraintes d'étoile-partition générées,
NT	:	le nombre de nœuds de l'arbre de branchement générés,

o/p	:	le nombre de problèmes résolus à l'optimum sur le nombre d'instances testées (uniquement pour les instances aléatoires),
Copt	:	la valeur de la solution optimale (uniquement pour les instances réelles),
Gap	:	l'erreur relative entre la meilleure borne supérieure (la valeur optimale si le problème à été résolu à l'optimum) et la borne inférieure obtenue avant branchement,
TT	:	le temps CPU total en h :m :s.

6.2.3 Problème MCSIPND$_m$

6.2.3.1 Instances aléatoires

Notre première série d'expérimentations pour le problème MCSIPND$_m$ (les liaisons multiples sont acceptées) concerne les instances aléatoires. Nous avons considéré trois instances pour chaque taille. Les tables 6.1 et 6.2 rapportent la moyenne des résultats obtenus pour ces trois instances avec l'algorithme de coupes et branchements basé sur la formulation arcs-sommets pour la table 6.1 et avec l'algorithme de coupes, génération de colonnes et branchements basé sur la formulation arcs-chemins pour la table 6.2.

On remarque que toutes les instances de la table 6.1 ayant 6 sommets ont été résolues à l'optimum. Cette résolution a de plus été très rapide. En effet, toutes ces instances ont été résolues en moyenne en moins de 5 minutes excepté celles avec $|\mathcal{F}| = 20$ et $|K| = 20$ qui ont été résolue en un peu plus d'une demi-heure. De même, les instances avec $|\mathcal{F}| = 20$ ont toutes été résolues dans le temps imparti sauf trois d'entre-elles ayant 20 demandes. Lorsque le nombre de demandes est fixé à 5, les instances sont toutes résolues en une trentaine de minutes, excepté les instances à 10 sommets et $|\mathcal{F}| = 20$ dont seulement une sur les trois instances a été résolue à l'optimum en moins de 5 heures.

Les instances ayant 10 et 20 demandes et $|\mathcal{F}| = 20$ semblent plus difficiles à résoudre. En effet, aucune des instances testées n'a été résolue à l'optimum lorsque le nombre de sommets dépasse 8. De même, pour les instances à 10 sommets et $|\mathcal{F}| = 20$, seule une instance a été résolue en moins de 5 heures.

Nous pouvons aussi remarquer que pour la plupart des instances de cette table, un nombre significatif de contraintes de coupe et de contraintes de capacité résiduelle ont été générées. Cela implique que ces inégalités sont utiles pour résoudre les instances

$\|V^1\|$	$\|\mathcal{F}\|$	$\|K\|$	VF	NC	NCR	NCC	NEP	NT	o/p	Gap	TT
6	10	5	1500	85.67	78.00	0.67	0.67	132.33	3/3	9.47	0 :00 :13.52
6	10	10	3000	80.00	89.00	0.00	0.00	109.67	3/3	5.86	0 :00 :40.11
6	10	20	6000	71.33	201.00	0.33	0.00	123.00	3/3	6.39	0 :02 :22.54
6	20	5	3000	105.00	226.33	2.00	0.67	220.33	3/3	10.74	0 :01 :38.04
6	20	10	6000	125.67	427.67	2.67	0.00	214.33	3/3	8.41	0 :04 :27.97
6	20	20	12000	153.33	641.00	0.33	0.00	347.00	3/3	7.46	0 :33 :29.84
8	10	5	2800	104.00	84.00	0.67	0.33	115.00	3/3	12.09	0 :00 :38.38
8	10	10	5600	304.33	449.67	2.00	1.00	771.00	3/3	10.68	0 :16 :57.89
8	10	20	11200	381.33	1290.00	0.00	0.00	2585.00	1/3	9.60	4 :13 :51.50
8	20	5	5600	282.67	1102.33	2.00	0.67	1041.00	3/3	14.84	0 :30 :23.18
8	20	10	11200	281.00	1524.67	2.00	0.67	2454.00	0/3	25.13	5 :00 :00.00
8	20	20	22400	283.00	3726.00	1.00	0.00	1266.33	0/3	15.30	5 :00 :00.00
10	10	5	4500	819.67	519.33	0.33	0.33	2282.33	3/3	13.58	0 :34 :04.75
10	10	10	9000	846.33	1188.67	1.00	0.00	1773.67	3/3	9.81	1 :39 :03.70
10	10	20	18000	524.67	1325.33	0.33	0.67	1144.00	2/3	33.42	3 :22 :05.67
10	20	5	9000	199.33	2140.00	0.67	0.00	2653.00	1/3	38.60	4 :05 :09.57
10	20	10	18000	378.33	5020.00	0.67	0.00	1523.67	0/3	33.59	5 :00 :00.00
10	20	20	36000	491.00	4169.33	0.67	0.00	457.00	0/3	45.39	5 :00 :00.00

TAB. 6.1 – Résultats pour les instances aléatoires du problème MCSIPND$_m$ avec la formulation arcs-sommets

aléatoires. Les contraintes de coupe-cycle et d'étoile-partition ne semblent pas jouer un grand rôle pour ce type d'instances. En effet, les arêtes n'appartiennent pas nécessairement à un ensemble F_e, et donc il est assez difficile de trouver des configurations de coupe-cycle et d'étoile-partition.

D'autre part, même pour les instances les plus petites avec 6 sommets, le nombre de nœuds dans l'arbre de branchements est supérieur à 100. Ce nombre peut même atteindre les 2500 pour les instances les plus difficiles. Nos contraintes ne sont donc pas suffisantes pour résoudre le problème dans la phase de coupes. On peut également noter que le gap est assez important pour la plupart des instances.

La table 6.2 présente les mêmes instances que la table 6.1, mais celle-ci contient la moyenne des résultats obtenus pour les 3 instances de chaque taille, à l'aide de l'algorithme de coupes, génération de colonnes et branchements basé sur la formulation arcs-chemins.

Toutes les instances à 6 sommets ont été résolues à l'optimum en moins de 2 minutes en moyenne. De même, pour $|\mathcal{F}| = 10$, toutes les instances ont été résolues en moins de deux heures excepté les trois instances à 10 sommets et 20 demandes qui n'ont pas

$\|V^1\|$	$\|\mathcal{F}\|$	$\|K\|$	VF	NC	NCC	NEP	NT	o/p	Gap	TT
6	10	5	590.67	93.33	0.33	0.67	112.33	3/3	9.40	0 :00 :07.25
6	10	10	899.33	92.67	0.00	0.00	95.00	3/3	5.24	0 :00 :08.38
6	10	20	1363.33	124.67	0.00	0.00	157.00	3/3	5.96	0 :00 :15.40
6	20	5	813.33	118.33	1.67	0.00	237.67	3/3	10.71	0 :00 :33.72
6	20	10	1490.00	156.67	1.00	0.00	165.67	3/3	8.29	0 :00 :39.12
6	20	20	2326.00	156.33	0.33	0.00	297.00	3/3	7.38	0 :01 :45.86
8	10	5	1094.00	91.67	0.33	0.33	121.67	3/3	11.35	0 :00 :22.58
8	10	10	2769.33	457.00	0.67	0.00	876.33	3/3	10.31	0 :04 :58.44
8	10	20	4100.00	2083.00	1.33	0.00	3334.33	3/3	9.48	0 :22 :59.97
8	20	5	3324.67	246.33	2.67	1.00	1223.67	3/3	11.89	0 :21 :40.39
8	20	10	5937.33	5472.00	1.67	2.00	9515.67	2/3	13.63	3 :11 :59.93
8	20	20	5619.00	1733.33	0.67	0.00	5004.67	2/3	10.64	1 :57 :35.10
10	10	5	7192.67	1271.33	0.33	0.00	2159.00	3/3	11.49	0 :52 :43.15
10	10	10	8197.33	2775.00	0.67	0.00	3378.33	3/3	9.98	1 :45 :52.21
10	10	20	12500.00	2773.67	1.00	0.00	11246.67	0/3	12.38	5 :00 :00.00
10	20	5	6991.00	1249.33	3.67	5.00	3978.33	1/3	33.94	3 :59 :39.57
10	20	10	11784.00	5659.67	2.67	2.33	3953.00	1/3	26.08	4 :33 :09.10
10	20	20	14427.33	3986.67	1.00	0.67	3665.00	0/3	33.98	5 :00 :00.00

TAB. 6.2 – Résultats pour les instances aléatoires du problème MCSIPND$_m$ avec la formulation arcs-chemins

été résolues dans le temps imparti. Pour les instances ayant un nombre plus petit de demandes, le problème semble plus facile à résoudre. En effet, pour les instances ayant 5 demandes, seules 2 d'entre-elles (à 10 sommets et $|\mathcal{F}| = 20$) n'ont pas été résolues.

Parmi les instances à 10 sommets et $|\mathcal{F}| = 20$, seules deux ont été résolues. De plus, le nombre de nœuds dans l'abre de branchements est encore très grand et le gap peut dépasser les 30% pour les instances les plus difficiles.

Sur les 54 instances testées, 17 n'ont pas été résolues en moins de 5 heures dans la table 6.1 contre 12 dans la table 6.2. Il semble donc que l'algorithme basé sur la formulation arcs-chemins est plus efficace que celui basé sur la formulation arcs-sommets. Neanmoins, les instances résolues avec le deuxième n'ont pas toutes été résolues avec le premier. En effet, pour $|V^1| = 10$, $|\mathcal{F}| = 10$ et $|K| = 20$, la formulation arcs-sommets a permis de résoudre deux instances de plus que la formulation arcs-chemins. Par contre, la formulation arcs-chemins a obtenu de bien meilleurs résultats pour les instances à 8 sommets avec 6 instances supplémentaires résolues. De plus, elle a permis de résoudre une instance difficile à 10 sommets, $|\mathcal{F}| = 20$ et 10 demandes qui n'avait pas été résolue avec la formulation arcs-sommets.

Le nombre de variables de flot pour la formulation arcs-sommets est fixe et dépend

uniquement de $|V^1|$, $|\mathcal{F}|$ et $|K|$. Ce n'est pas le cas des variables de flot de la formulation arcs-chemins. On remarque que le nombre de ces variables de flot sur les chemins est inférieur en moyenne au nombre de variables de flots sur les arêtes de la première formulation, excepté dans le cas des instances à 10 sommets, $|\mathcal{F}| = 10$ et 5 demandes.

On remarque également que les temps CPU de la table 6.2 sont toujours inférieurs à ceux de la table 6.1, excepté pour les instances à 10 sommets et $|\mathcal{F}| = 10$. Par exemple, pour les instances à 6 sommets, l'algorithme de coupes, génération de colonnes et branchements permet d'obtenir la solution optimale en moins de 2 minutes contre plus de 30 minutes avec le deuxième algorithme. La formulation arcs-chemins semble donc donner de meilleurs résultats pour les instance aléatoires. Néanmoins, elle génère plus de nœuds dans l'arbre de branchements.

6.2.3.2 Instances réelles

Les tables 6.3 à 6.6 présentent les résultats pour le problème MCSIPND$_m$ pour les instances réelles à l'aide des deux algorithmes. Dans un premier temps, nous nous intéressons aux instances associées à la fonction coût $c_1(.)$.

Fonction coût $c_1(.)$

On rappelle que la fonction coût $c_1(.)$ représente pour chaque liaison de la couche cliente, le nombre de liaisons de la couche tranport utilisées dans le chemin correspondant. Cette fonction coût est donc entière.

La table 6.3 rapporte les résultats obtenus à l'aide de l'algorithme de coupes et branchements basé sur la formulation arcs-sommets pour les instances où le coût d'une arête f de la couche IP est donné par $c_1(f)$.

La difficulté augmente avec le nombre de sommets et le nombre de demandes. Toutes les instances comportant 5 demandes et ayant jusqu'à 15 sommets ont été résolues dans le temps imparti et la résolution ayant demandé le plus de temps est celle de l'instance à 15 sommets qui a duré un peu moins d'une heure et vingt minutes. De même, pour les instances à 10 demandes, on a obtenu une solution optimale pour toutes celles de moins de 10 sommets et ce en moins de trois heures. De même les instances à 14 et 16 sommets ont également été résolues. Pour les instances à 15 (resp. 20) demandes, seules les instances ayant moins de 8 (resp. 7) sommets ont pu être résolues.

Pour toutes les instances qui n'ont pas pu être résolues en moins de 5 heures, nous avons obtenu une solution réalisable pour le problème (donnée en italique). De plus, on

$\|V^1\|$	$\|\mathcal{F}\|$	$\|K\|$	VF	NC	NCR	NCC	NEP	NT	Copt	Gap	TT
6	11	5	1650	10	10	0	1	3	57	0.00	0 :00 :00.77
6	11	10	3300	56	111	2	3	65	99	8.08	0 :00 :27.28
6	11	15	4950	99	779	1	0	157	105	7.62	0 :02 :41.68
6	11	20	6600	199	1194	1	0	375	115	9.57	0 :13 :34.46
7	14	5	2940	11	29	1	0	1	57	0.00	0 :00 :01.13
7	14	10	5880	233	571	5	5	387	110	19.09	0 :13 :52.34
7	14	15	8820	173	1590	3	2	321	111	14.41	0 :30 :25.41
7	14	20	11760	331	2706	4	3	367	117	11.97	0 :40 :30.21
8	17	5	4760	173	114	2	0	115	81	11.11	0 :02 :36.71
8	17	10	9520	191	717	2	7	567	109	15.60	0 :49 :59.51
8	17	15	14280	512	815	4	2	621	123	13.01	2 :12 :27.03
8	17	20	19040	666	9743	7	5	709	*160*	33.75	5 :00 :00.00
9	22	5	7920	105	111	4	0	141	129	7.75	0 :11 :44.37
9	22	10	15840	208	234	4	0	69	162	3.70	0 :10 :43.66
9	22	15	23760	755	2275	9	0	895	*214*	14.49	5 :00 :00.00
9	22	20	31680	134	3555	12	0	481	*253*	25.30	5 :00 :00.00
10	25	5	11250	71	157	4	0	113	129	7.75	0 :16 :27.52
10	25	10	23400	262	279	4	2	369	156	3.21	2 :45 :10.77
10	25	15	33750	772	1616	6	0	473	*231*	18.18	5 :00 :00.00
10	25	20	45000	447	2754	6	1	247	*297*	34.34	5 :00 :00.00
11	27	5	14850	60	144	4	0	89	129	8.53	0 :24 :27.79
11	27	10	29700	246	411	4	1	429	*157*	3.82	5 :00 :00.00
11	27	15	44550	546	1642	7	0	275	*257*	26.46	5 :00 :00.00
11	27	20	59400	358	2289	6	0	131	*313*	35.14	5 :00 :00.00
12	32	5	21120	85	180	5	0	99	129	8.53	0 :54 :33.54
12	32	10	42240	333	777	5	1	303	*194*	12.37	5 :00 :00.00
12	32	15	63360	251	1129	8	0	127	*300*	32.67	5 :00 :00.00
12	32	20	84480	357	2001	7	0	89	*392*	41.58	5 :00 :00.00
13	26	5	20280	54	117	4	0	59	129	9.30	0 :30 :16.22
13	26	10	40560	270	443	4	0	341	*221*	23.53	5 :00 :00.00
13	26	15	60840	226	895	2	5	153	*297*	31.65	5 :00 :00.00
14	27	5	24570	55	22	0	0	73	150	8.00	0 :34 :17.06
14	27	10	49140	342	298	2	0	175	198	6.06	4 :56 :12.04
15	28	5	29400	68	212	4	0	73	129	10.08	1 :16 :25.47
15	28	10	58800	305	407	2	1	151	*195*	7.69	5 :00 :00.00
16	29	5	34800	315	208	3	0	323	*153*	11.76	5 :00 :00.00
16	29	10	69600	88	206	1	0	67	185	4.32	4 :21 :11.28
17	29	5	39440	213	325	1	0	303	*180*	16.11	5 :00 :00.00
18	30	5	45900	87	234	2	0	229	*218*	17.43	5 :00 :00.00

TAB. 6.3 – Résultats pour les instances réelles du problème MCSIPND$_m$ avec la fonction coût $c_1(.)$ et la formulation arcs-sommets

peut remarquer que pour certaines de ces instances, le gap obtenu est relativement petit. Ce gap est par exemple inférieur à 4% pour l'instance à 11 sommets et 10 demandes. Et pour les plus grandes instances à 17 et 18 sommets, le gap est inférieur à 18%.

La table 6.4 rapporte les résultats obtenus à l'aide de l'algorithme basé sur la formulation arcs-chemins pour les mêmes instances que la table 6.3. Toutes les instances résolues dans la table 6.3 ont également abouti à une solution optimale excepté l'instance à 14 sommets et 10 demandes mais la solution réalisable trouvée est en fait optimale. De plus, le temps mis pour cette résolution est bien inférieur excepté pour l'instance à 15 sommets et 5 demandes. En effet, pour l'instance à 7 sommets et 20 demandes, le temps CPU est passé de 40 minutes à 3 minutes. Pour celle à 9 sommets et 10 demandes, on est passé de 10 minutes à moins de 2 et pour celle à 8 sommets et 15 demandes, le programme a gagné 2 heures. 4 nouvelles instances ont été résolues à l'optimum dont 2 en moins de 1 heure 20 minutes. On peut remarquer que la solution réalisable obtenue avec la formulation arcs-sommets pour l'instance à 11 sommets et 10 demandes était de 157 alors que la solution optimale est de 156. Les solutions réalisables trouvées pour les instances à 8 sommets et 20 demandes, à 9 sommets et 20 demandes et à 13 sommets et 10 demandes sont beaucoup plus éloignées de la solution optimale pour le problème.

En ce qui concerne les instances non résolues par les deux algorithmes dans le temps imparti, on peut remarquer que la meilleure solution réalisable trouvée avec la formulation arcs-chemins est inférieure à celle trouvée à l'aide de la formulation arcs-sommets pour la majorité des instances.

Comme pour les instances aléatoires, pour la plupart des instances, un nombre significatif de contraintes de coupes a été généré. On remarque que les contraintes de coupe-cycle et d'étoile-partition semblent jouer un rôle plus important pour les instances réelles que pour les instances aléatoires. On peut également remarquer que plus le nombre de demandes augmente, plus nombreuses sont ces contraintes. En effet, il est plus facile de trouver des configurations d'étoile-partition et de coupe-cycle. Il semble que la formulation arcs-sommets permette de générer plus de contraintes de coupe-cycle (jusqu'à 12 contraintes générées pour l'instance à 9 sommets et 20 demandes) et la formulation arcs-chemins privilégie plutot les contraintes d'étoile-partition (jusqu'à 19 contraintes générées pour l'instance à 8 sommets et 20 demandes).

Pour ces instances réelles, on peut également remarquer que le nombre de variables de flot utilisé pour la formulation arcs-chemins est inférieur à celui de la formulation arcs-sommets. Par exemple, pour l'instance à 16 sommets et 10 demandes qui utilise 69600 variables de flot pour la première, on n'a généré uniquement 19934 variables de

$\|V^1\|$	$\|\mathcal{F}\|$	$\|K\|$	VF	NC	NCC	NEP	NT	Copt	Gap	TT
6	11	5	137	15	0	0	3	57	0.00	0 :00 :00.32
6	11	10	628	59	2	1	81	99	8.08	0 :00 :07.12
6	11	15	1071	88	1	0	147	105	7.62	0 :00 :17.86
6	11	20	1778	233	1	0	451	115	9.57	0 :01 :37.51
7	14	5	200	10	0	1	3	57	0.00	0 :00 :00.51
7	14	10	2124	415	4	2	1077	110	15.45	0 :06 :31.10
7	14	15	2580	181	4	3	301	111	13.51	0 :02 :26.90
7	14	20	2814	332	2	4	415	117	11.97	0 :02 :59.06
8	17	5	1693	178	1	0	93	81	11.11	0 :00 :55.39
8	17	10	3452	180	2	5	449	109	15.60	0 :07 :13.23
8	17	15	4717	511	4	2	1155	123	14.63	0 :18 :48.43
8	17	20	8230	2570	5	19	5655	131	19.08	3 :34 :26.29
9	22	5	2846	71	1	0	97	129	8.53	0 :03 :07.79
9	22	10	2400	282	2	0	65	162	1.85	0 :01 :32.51
9	22	15	6964	5288	6	2	8101	*197*	7.11	5 :00 :00.00
9	22	20	7230	1649	5	4	1651	199	5.03	1 :16 :52.77
10	25	5	4011	99	2	0	151	129	8.53	0 :08 :16.44
10	25	10	5594	318	2	4	373	156	3.21	0 :43 :35.76
10	25	15	8673	5497	5	0	3993	*213*	11.27	5 :00 :00.00
10	25	20	8966	4038	9	3	3317	*261*	24.90	5 :00 :00.00
11	27	5	5155	123	2	0	173	129	8.53	0 :16 :25.61
11	27	10	7598	351	3	5	303	156	3.21	1 :09 :01.47
11	27	15	11139	2807	5	0	1959	*220*	14.09	5 :00 :00.00
11	27	20	11674	2598	9	8	1739	*284*	28.17	5 :00 :00.00
12	32	5	7188	100	2	0	153	129	8.53	0 :34 :49.36
12	32	10	15120	572	4	1	559	*182*	6.59	5 :00 :00.00
12	32	15	8852	262	5	2	127	*285*	28.77	5 :00 :00.00
12	32	20	13988	1109	7	0	681	*378*	39.15	5 :00 :00.00
13	26	5	6514	55	2	0	31	129	6.98	0 :17 :16.66
13	26	10	17494	616	4	0	403	179	4.47	4 :39 :40.15
13	26	15	18062	468	2	6	679	*235*	13.62	5 :00 :00.00
14	27	5	4081	44	0	0	49	150	8.00	0 :10 :56.41
14	27	10	17610	608	1	0	205	*198*	6.06	5 :00 :00.00
15	28	5	14900	180	2	0	177	129	8.53	3 :33 :58.80
15	28	10	19266	403	0	2	223	*219*	17.81	5 :00 :00.00
16	29	5	15910	294	3	0	177	*159*	15.09	5 :00 :00.00
16	29	10	19934	114	2	0	77	185	4.32	3 :26 :08.28
17	29	5	21843	206	0	0	87	*182*	17.03	5 :00 :00.00
18	30	5	21494	47	0	0	63	*220*	18.64	5 :00 :00.00

TAB. 6.4 – Résultats pour les instances réelles du problème MCSIPND$_m$ avec la fonction coût $c_1(.)$ et la formulation arcs-chemins

flot sur les chemins.

Fonction coût $c_2(.)$

Les tables 6.5 et 6.6 présentent les résultats pour les instances utilisant la fonction coût $c_2(.)$. Contrairement à $c_1(.)$ cette fonction n'est pas entière. Elle est basée sur la distance euclidienne entre les extrémintés des arêtes composant les chemins de la couche transport.

La table 6.5 rapporte les résultats de l'algorithme de coupes et branchements basé sur la formulation arcs-sommets pour les instances avec la fonction coût $c_2(.)$. Ces instances semblent plus difficiles à résoudre que leurs homologues utilisant la fonction coût $c_1(.)$. En effet, 4 instances de moins ont été résolues et les instances résolues l'ont été en beaucoup plus de temps. Par exemple, l'instance à 13 sommets et 5 demandes a demandé plus de 2 heures au lieu de 30 minutes. De même, les gap sont plus grand mais le nombre de nœuds générés dans l'arbre de branchement est moins important. On remarque également que le nombre de contraintes de coupe-cycle et d'étoile-partition générées est similaire à celui présent dans la table 6.3.

Comme pour la résolution basée sur la formulation arcs-sommets, les instances réelles utilisant la fonction coût $c_2(.)$ sont plus difficiles à résoudre que celles utilisant la fonction coût $c_1(.)$ à l'aide de la formulation arcs-chemins. La table 6.6 présente les résultats pour cette formulation. Aucune instance de plus de 12 sommets n'a été résolue et celles qui l'on été ont demandé un temps assez long.

Lorsque les instances ont été résolues avec les deux algorithmes en moins de 5 heures, on remarque que celui basé sur la formulation arcs-chemins a demandé moins de temps. De nombreuses instances pour lesquelles on a obtenu la solution optimale avec l'une des formulations n'ont pu être résolues à l'aide de la deuxième. En effets, les instances à 9 sommets et 10, 15 ou 20 demandes ont été résolues à l'aide de la formulation arcs-chemins et ne l'ont pas été avec la formulation arcs-sommets, tout comme l'instance à 10 sommets et 10 demandes. Par contre la remarque inverse peut être faite à propos des instances à 12, 13 et 14 sommets et 5 demandes.

Pour ce dernier type d'instances, on peut remarquer que, comme pour les autres, le nombre de variables de flot sur les chemins générées est inférieur à celui des variables de flot sur les arêtes, quelque soit l'instance considérée.

$\|V^1\|$	$\|\mathcal{F}\|$	$\|K\|$	VF	NC	NCR	NCC	NEP	NT	Copt	Gap	TT
6	11	5	1650	10	7	1	0	7	4401.00	9.67	0 :00 :01.35
6	11	10	3300	64	177	1	2	85	7115.00	7.43	0 :00 :51.25
6	11	15	4950	68	733	1	0	259	7472.00	8.55	0 :06 :35.93
6	11	20	6600	84	191	0	0	97	7983.00	6.02	0 :04 :35.07
7	14	5	2940	19	17	2	1	13	4401.00	9.67	0 :00 :05.24
7	14	10	5880	130	321	5	0	137	7164.00	11.59	0 :05 :13.47
7	14	15	8820	111	762	5	2	145	7164.00	14.17	0 :12 :07.88
7	14	20	11760	164	605	4	0	99	7810.00	9.42	0 :11 :02.72
8	17	5	4760	165	104	1	0	207	5271.00	13.30	0 :05 :53.31
8	17	10	9520	108	394	3	3	73	6630.00	13.48	0 :04 :54.51
8	17	15	14280	179	1648	6	0	215	7611.00	15.76	0 :32 :24.44
8	17	20	19040	160	2045	7	2	183	7776.00	15.64	1 :08 :53.41
9	22	5	7920	71	227	5	0	267	8334.00	15.67	0 :22 :46.10
9	22	10	15840	727	727	6	2	1325	*10231.00*	8.64	5 :00 :00.00
9	22	15	23760	510	2023	6	4	927	*12998.00*	15.06	5 :00 :00.00
9	22	20	31680	289	1852	5	0	483	*12124.00*	6.46	5 :00 :00.00
10	25	5	11250	89	211	5	0	301	8334.00	15.67	0 :49 :53.91
10	25	10	22500	359	756	5	1	959	*10370.00*	8.61	5 :00 :00.00
10	25	15	33750	501	1489	7	0	487	*12987.00*	11.06	5 :00 :00.00
10	25	20	45000	339	2389	9	1	271	*15567.00*	25.10	5 :00 :00.00
11	27	5	14850	105	312	5	0	319	8334.00	15.52	1 :24 :06.57
11	27	10	29700	299	813	6	0	607	*11529.00*	13.59	5 :00 :00.00
11	27	15	44550	206	1412	6	0	291	*15552.00*	22.59	5 :00 :00.00
11	27	20	59400	281	2080	9	1	157	*17187.00*	25.35	5 :00 :00.00
12	32	5	21120	120	416	5	0	445	8334.00	15.67	4 :11 :39.43
12	32	10	42240	291	750	6	3	301	*12597.00*	17.80	5 :00 :00.00
12	32	15	63360	218	1662	10	1	135	*19557.00*	36.34	5 :00 :00.00
12	32	20	84480	179	2060	8	0	103	*19605.00*	30.34	5 :00 :00.00
13	26	5	20280	246	473	5	0	409	8337.00	13.85	2 :15 :35.27
13	26	10	40560	312	487	7	0	345	*11775.00*	11.90	5 :00 :00.00
13	26	15	60840	206	1215	7	2	151	*17772.00*	29.62	5 :00 :00.00
14	27	5	24570	49	79	0	0	149	9153.00	4.31	1 :05 :19.59
14	27	10	49140	302	355	3	0	213	*12303.00*	3.92	5 :00 :00.00
15	28	5	29400	300	693	3	0	533	*9753.00*	26.29	5 :00 :00.00
15	28	10	58800	152	544	5	0	157	*15635.00*	30.10	5 :00 :00.00
16	29	5	34800	292	444	2	0	365	*9511.00*	20.49	5 :00 :00.00
16	29	10	69600	208	403	5	0	141	*15880.00*	34.76	5 :00 :00.00
17	29	5	39440	122	283	1	0	301	*8904.00*	14.80	5 :00 :00.00
18	30	5	45900	55	216	1	0	209	*9162.00*	15.07	5 :00 :00.00

TAB. 6.5 – Résultats pour les instances réelles du problème MCSIPND$_m$ avec la fonction coût $c_2(.)$ et la formulation arcs-sommets

$\|V^1\|$	$\|\mathcal{F}\|$	$\|K\|$	VF	NC	NCC	NEP	NT	Copt	Gap	TT
6	11	5	299	16	0	0	7	4401.00	10.21	0 :00 :01.16
6	11	10	528	67	2	2	41	7115.00	7.32	0 :00 :03.13
6	11	15	1141	106	1	1	229	7472.00	8.23	0 :00 :26.48
6	11	20	1352	92	1	0	101	7983.00	6.02	0 :00 :17.56
7	14	5	636	25	0	1	11	4401.00	10.21	0 :00 :02.98
7	14	10	1424	147	4	5	187	7164.00	11.78	0 :00 :41.19
7	14	15	7192	131	4	5	159	7164.00	11.52	0 :01 :15.83
7	14	20	2130	149	2	4	95	7810.00	9.56	0 :00 :43.78
8	17	5	2403	147	1	0	185	5271.00	13.30	0 :02 :28.12
8	17	10	1872	64	2	6	73	6630.00	13.95	0 :00 :40.42
8	17	15	3271	145	3	2	211	7611.00	13.41	0 :02 :53.17
8	17	20	4224	203	5	3	181	7776.00	14.10	0 :05 :24.05
9	22	5	4030	74	2	0	197	8334.00	12.57	0 :08 :14.27
9	22	10	5214	571	5	0	717	10231.00	7.26	0 :30 :17.45
9	22	15	5756	680	5	2	765	11851.00	6.66	0 :28 :36.74
9	22	20	6304	519	4	2	649	12121.00	5.74	0 :33 :02.45
10	25	5	6601	119	2	0	287	8334.00	14.33	0 :28 :15.22
10	25	10	8864	274	4	2	677	10370.00	8.36	1 :21 :48.66
10	25	15	10453	2632	5	0	2693	*12460.00*	6.36	5 :00 :00.00
10	25	20	11490	1634	7	2	2413	*12654.00*	8.03	5 :00 :00.00
11	27	5	9445	113	2	0	273	8334.00	12.57	1 :12 :01.40
11	27	10	14542	414	4	1	1265	*11052.00*	9.51	5 :00 :00.00
11	27	15	14143	1254	5	0	1489	*13999.00*	13.03	5 :00 :00.00
11	27	20	14438	1937	9	3	1509	*15900.00*	19.03	5 :00 :00.00
12	32	5	17748	165	2	0	451	*8334.00*	14.72	5 :00 :00.00
12	32	10	20234	657	5	2	661	*12035.00*	13.20	5 :00 :00.00
12	32	15	20488	549	7	5	723	*14041.00*	11.06	5 :00 :00.00
12	32	20	18522	940	8	0	373	14506.00	4.69	3 :24 :00.65
13	26	5	20500	261	1	0	337	*8358.00*	13.25	5 :00 :00.00
13	26	10	21856	348	5	0	457	*10980.00*	5.43	5 :00 :00.00
13	26	15	22412	381	6	5	409	*13461.00*	7.16	5 :00 :00.00
14	27	5	12363	53	0	0	101	*9153.00*	3.91	5 :00 :00.00
14	27	10	21550	580	4	0	309	*12942.00*	8.46	5 :00 :00.00
15	28	5	24986	185	1	0	123	*9426.00*	23.57	5 :00 :00.00
15	28	10	24244	228	2	0	129	*15588.00*	30.11	5 :00 :00.00
16	29	5	29922	159	0	0	75	*8810.00*	14.81	5 :00 :00.00
16	29	10	28638	183	4	0	113	*15691.00*	33.29	5 :00 :00.00
17	29	5	29695	93	0	0	45	*9813.00*	22.70	5 :00 :00.00
18	30	5	31240	23	0	0	47	*9936.00*	21.68	5 :00 :00.00

TAB. 6.6 – Résultats pour les instances réelles du problème MCSIPND$_m$ avec la fonction coût $c_2(.)$ et la formulation arcs-chemins

6.2.4 Problème MCSIPND$_s$

Les tables 6.7 à 6.12 contiennent les résultats relatifs aux expérimentations pour le problème MCSIPND$_s$. Comme pour la section précédente portant sur le problème MCSIPND$_m$, nous avons considéré des instances aléatoires ainsi que des instances réelles. Nous avons testé nos deux algorithmes sur ces différentes instances.

6.2.4.1 Instances aléatoires

$\|V^1\|$	$\|\mathcal{F}\|$	$\|K\|$	VF	NC	NCR	NS	NCC	NEP	NT	o/p	Gap	TT
6	10	5	1500	58.67	49.67	3.67	0.33	0.00	46.33	3/3	14.04	0 :00 :07.70
6	10	10	3000	43.67	47.33	4.67	0.00	0.00	23.67	3/3	4.48	0 :00 :15.00
6	10	20	6000	48.67	96.67	3.33	0.00	0.00	36.33	3/3	6.63	0 :01 :17.19
6	20	5	3000	75.67	151.00	3.00	1.67	0.33	106.33	3/3	16.16	0 :01 :04.65
6	20	10	6000	70.67	187.33	9.67	1.00	0.00	41.00	3/3	9.93	0 :01 :29.15
6	20	20	12000	44.67	94.33	2.67	0.00	0.00	24.33	3/3	6.22	0 :03 :11.87
8	10	5	2800	81.67	97.33	0.67	0.33	0.33	85.00	3/3	13.16	0 :00 :45.93
8	10	10	5600	215.33	293.00	2.33	0.67	0.00	296.33	3/3	10.02	0 :06 :34.95
8	10	20	11200	277.00	773.33	4.33	0.00	0.00	373.67	3/3	9.02	0 :30 :53.21
8	20	5	5600	177.33	570.33	1.00	1.67	0.67	118.33	3/3	13.43	0 :05 :30.62
8	20	10	11200	381.00	1140.00	11.00	0.67	0.67	772.33	3/3	14.34	1 :38 :42.90
8	20	20	22400	305.67	1988.00	13.00	0.33	0.00	639.67	1/3	12.44	3 :50 :48.94
10	10	5	4500	189.67	510.00	0.00	0.00	0.00	412.00	3/3	18.06	0 :06 :52.79
10	10	10	9000	171.33	610.00	2.00	0.00	0.00	1379.00	3/3	11.49	1 :39 :00.75
10	10	20	18000	176.67	823.33	0.00	1.00	0.33	1214.00	2/3	15.42	2 :54 :21.12
10	20	5	9000	314.00	1293.00	0.00	3.00	0.00	507.00	1/3	22.14	3 :29 :34.19
10	20	10	18000	740.00	5198.00	0.00	1.00	0.00	1463.00	0/3	34.28	5 :00 :00.00
10	20	20	36000	448.67	4156.33	1.67	0.67	0.00	464.33	0/3	27.72	5 :00 :00.00

TAB. 6.7 – Résultats pour les instances aléatoires du problème MCSIPND$_s$ avec la formulation arcs-sommets

La table 6.7 regroupe les différents résultats de l'algorithme de coupes et branchements basé sur la formulation arcs-sommets pour des instances aléatoires.

Toutes les instances à 6 sommets ont été résolues à l'optimum en moins de 4 minutes. Il en est de même pour toutes les instances avec $|\mathcal{F}| = 10$ excepté l'une des plus grandes contenant 10 sommets et 20 demandes. De même, on a obtenu une solution optimale pour toutes les instances ayant 5 demandes, excepté 2 d'entre-elles à 10 sommets et $|\mathcal{F}| = 20$.

Parmi les 9 instances à 10 sommets et $|\mathcal{F}| = 20$, une seule a été résolue avant le temps limite de 5 heures. Le gap obtenu pour les instances aléatoires n'est pas très élevé excepté pour ces dernières instances non résolues, même si l'on a tout de même obtenu des solutions réalisables.

Il semble que le problème MCSIPND$_s$ dans sa version sans liaisons multiples soit plus facile à résoudre pour les instances aléatoires et la formulation arcs-sommets. En effet, seulement 11 des 54 instances n'ont pas été résolues pour le problème MCSIPND$_s$, pour 17 pour le problème MCISPND$_m$ dans les mêmes conditions (voir table 6.1).

Les contraintes de saturation qui ne sont valides que pour le problème MCSIPND$_m$, bien qu'étant présentes en nombre moins important que les contraintes de coupes et les contraintes de capacité résiduelle, sont prépondérantes dans la résolution de ces instances aléatoires. Comme pour la version avec liaisons multiples, le nombre de contraintes de coupe-cycle et d'étoile-partition générées n'est pas significatif pour les instances aléatoires.

$\|V^1\|$	$\|\mathcal{F}\|$	$\|K\|$	VF	NC	NS	NCC	NEP	NT	o/p	Gap	TT
6	10	5	550.67	85.00	3.33	0.33	0.33	52.33	3/3	13.93	0 :00 :04.58
6	10	10	776.67	39.57	4.67	0.33	0.00	29.67	3/3	4.54	0 :00 :02.95
6	10	20	836.67	21.00	1.33	0.00	0.00	17.00	3/3	4.93	0 :00 :02.39
6	20	5	812.00	55.67	3.00	2.00	0.00	93.67	3/3	15.82	0 :00 :16.27
6	20	10	1194.33	49.33	8.67	2.00	0.00	50.33	3/3	9.02	0 :00 :10.58
6	20	20	1532.33	44.33	3.67	0.00	0.00	24.33	3/3	7.58	0 :00 :08.16
8	10	5	1225.33	90.67	1.00	1.00	0.67	81.67	3/3	12.42	0 :00 :20.48
8	10	10	1910.67	259.00	2.33	2.33	0.00	270.33	3/3	9.16	0 :01 :15.36
8	10	20	2912.00	328.67	4.33	0.00	0.00	309.67	3/3	8.76	0 :02 :02.43
8	20	5	2514.67	164.67	1.33	3.00	0.33	135.67	3/3	12.71	0 :02 :30.74
8	20	10	4944.00	623.67	10.00	1.00	0.33	1890.33	3/3	14.12	0 :53 :31.81
8	20	20	5373.00	425.00	17.33	0.33	0.00	1000.33	3/3	10.79	0 :32 :17.99
10	10	5	7362.00	1036.33	0.00	0.33	0.00	2625.00	2/3	15.75	1 :46 :59.66
10	10	10	8874.00	873.00	0.33	2.00	0.00	2756.33	3/3	12.60	2 :02 :09.73
10	10	20	12624.00	1534.33	0.67	1.33	0.00	7562.67	1/3	11.87	2 :41 :08.19
10	20	5	10438.67	722.67	1.33	6.67	1.33	2799.67	1/3	22.90	3 :29 :28.52
10	20	10	13042.00	2816.67	0.00	4.33	2.00	*3395.67*	0/3	15.92	5 :00 :00.00
10	20	20	14828.00	3292.33	3.33	1.33	0.00	*3321.00*	0/3	17.82	5 :00 :00.00

TAB. 6.8 – Résultats pour les instances aléatoires du problème MCSIPND$_s$ avec la formulation arcs-chemins

La table 6.8 rapporte les différents résultats de l'algorithme de coupes, génération de colonnes et branchements basé sur la formulation arcs-chemins pour des instances aléatoires.

La différence entre les résultats pour le problème MCSIPND$_m$ et ceux pour le problème MCSIPND$_s$ pour la formulation arcs-chemins est moins significative que pour la formulation arcs-sommets. En effet, le nombre d'instances résolues est à peu près identique, même si le temps nécessaire à la résolution est dans l'ensemble moins important pour le problème MCSIPND$_s$. Le gap est également plus réduit pour les plus grandes instances.

Le nombre de contraintes de coupes générées est moins important pour le problème MCSIPND$_s$ et celui des contraintes de coupe-cycle et d'étoile-partition générées est presque identique.

Aucun des algorithmes ne semble vraiment être la plus efficace pour ces instances aléatoires. Certaines instances résolues avec l'un ne le sont pas avec l'autre et vice versa. C'est par exemple le cas de deux instances à 8 sommets, $|\mathcal{F}| = 20$ et 20 demandes ou d'une instance à 10 sommets, $|\mathcal{F}| = 10$ et 5 demandes ou d'une à 10 sommets, $|\mathcal{F}| = 10$ et 20 demandes. Néanmoins, on peut remarquer qu'on a obtenu une solution optimale pour toutes les instances à 6 et 8 sommets avec la formulation arcs-chemins alors que ce n'était pas le cas pour la formulation arcs-sommets.

6.2.4.2 Instances réelles

Les tables 6.9 à 6.12 correspondent aux résultats obtenus à l'aide des deux algorithmes pour le problème MCSIPND$_s$ pour des instances réelles avec les fonctions coûts $c_1(.)$ et $c_2(.)$.

Fonction coût $c_1(.)$

Les résultats pour la fonction coût $c_1(.)$ et la formulation arcs-sommets sont contenus dans la table 6.9. Comme pour les instances aléatoires, il semble que le problème MCSIPND$_s$ soit plus facile à résoudre que le problème MCSIPND$_m$. Ce phénomène peut être dû aux contraintes de saturation qui ne sont valides que pour le problème MCSIPND$_s$. En effet, le temps nécessaire à la résolution des instances est en moyenne moins grand. On peut remarquer que l'instance à 11 sommets et 10 demandes qui n'avait pas pu être résolue pour le problème MCSIPND$_m$ et la formulation arcs-sommets, l'a été pour le problème MCSIPND$_s$ en moins de 8 minutes. Pour toutes les instances qui n'ont pas été résolues, on dispose également d'une solution réalisable. Le gap est assez réduit, il ne dépasse que rarement 15%.

On peut également remarquer qu'une solution optimale pour le problème MCSIPND$_s$

$\|V^1\|$	$\|\mathcal{F}\|$	$\|K\|$	VF	NC	NCR	NS	NCC	NEP	NT	Copt	Gap	TT
6	11	5	1650	12	21	0	0	1	3	57	0.00	0 :00 :06.34
6	11	10	3300	46	137	1	2	0	79	104	10.58	0 :00 :58.33
6	11	15	4950	71	389	1	1	0	75	109	11.01	0 :01 :39.03
6	11	20	6600	72	705	1	1	0	191	122	9.84	0 :09 :39.32
7	14	5	2940	12	46	0	1	0	1	57	0.00	0 :00 :01.54
7	14	10	5880	144	546	3	8	1	705	116	20.69	0 :27 :57.15
7	14	15	8820	110	1660	3	7	2	303	117	17.95	0 :26 :59.79
7	14	20	11760	193	2129	3	5	0	103	122	15.57	0 :24 :11.47
8	17	5	4760	187	101	1	1	0	97	82	12.20	0 :01 :45.53
8	17	10	9520	165	955	4	5	4	777	115	20.00	1 :23 :37.11
8	17	15	14280	320	3983	3	8	0	981	129	22.48	4 :00 :51.52
8	17	20	19040	550	8849	3	10	3	771	*143*	25.87	5 :00 :00.00
9	22	5	7920	35	64	2	3	0	21	129	8.53	0 :01 :44.97
9	22	10	15840	107	111	1	1	0	19	163	1.84	0 :03 :14.06
9	22	15	23760	421	2168	2	3	0	783	*204*	7.84	5 :00 :00.00
9	22	20	31680	206	1972	1	5	0	389	*206*	6.80	5 :00 :00.00
10	25	5	11250	44	135	2	4	0	31	129	8.53	0 :04 :04.37
10	25	10	23400	82	47	1	0	0	17	157	3.82	0 :05 :29.97
10	25	15	33750	553	1743	2	4	0	485	*223*	13.90	5 :00 :00.00
10	25	20	4500	336	3346	2	6	0	227	*261*	24.52	5 :00 :00.00
11	27	5	14850	36	114	2	3	0	45	129	8.53	0 :07 :42.76
11	27	10	29700	86	42	1	0	0	13	157	3.82	0 :07 :24.09
11	27	15	44550	477	1717	1	4	0	285	*223*	13.90	5 :00 :00.00
11	27	20	59400	371	1872	1	8	0	141	*282*	27.30	5 :00 :00.00
12	32	5	21120	49	215	1	4	0	37	129	7.75	0 :21 :12.29
12	32	10	42240	233	485	1	4	0	291	*186*	8.60	5 :00 :00.00
12	32	15	63360	211	1125	1	4	0	139	*247*	18.22	5 :00 :00.00
12	32	20	84480	224	1087	1	7	0	79	*317*	27.44	5 :00 :00.00
13	26	5	20280	43	155	0	4	0	51	129	9.30	0 :18 :06.50
13	26	10	40560	187	421	0	1	0	301	*186*	9.14	5 :00 :00.00
13	26	15	60840	219	927	1	1	1	155	*242*	16.12	5 :00 :00.00
14	27	5	24570	57	33	0	0	0	41	150	4.67	0 :19 :08.56
14	27	10	49140	512	387	0	0	0	231	*226*	17.26	5 :00 :00.00
15	28	5	29400	52	217	0	3	0	51	129	10.08	0 :29 :51.69
15	28	10	58800	205	442	0	2	0	153	*215*	16.28	5 :00 :00.00
16	29	5	34800	343	316	0	1	0	265	*158*	13.92	5 :00 :00.00
16	29	10	69600	85	171	0	1	0	45	191	5.76	2 :16 :08.88
17	29	5	39440	340	312	1	1	0	305	*190*	19.47	5 :00 :00.00
18	30	5	45900	68	262	1	3	0	229	*211*	14.69	5 :00 :00.00

TAB. 6.9 – Résultats pour les instances réelles du problème MCSIPND$_s$ avec la fonction coût $c_1(.)$ et la formulation arcs-sommets

$\|V^1\|$	$\|\mathcal{F}\|$	$\|K\|$	VF	NC	NS	NCC	NEP	NT	Copt	Gap	TT
6	11	5	103	4	2	4	0	1	57	0.00	0 :00 :00.24
6	11	10	618	31	1	3	0	89	104	10.58	0 :00 :06.05
6	11	15	931	48	1	2	0	57	109	10.09	0 :00 :07.15
6	11	20	1496	47	1	3	0	173	122	9.84	0 :00 :27.73
7	14	5	190	5	2	5	0	1	57	0.00	0 :00 :00.39
7	14	10	2026	134	2	11	1	999	116	18.97	0 :04 :36.40
7	14	15	2284	94	2	12	3	257	117	17.09	0 :01 :50.83
7	14	20	3226	165	3	9	1	255	122	15.57	0 :03 :21.53
8	17	5	1549	109	1	2	0	61	82	12.20	0 :00 :27.62
8	17	10	2950	118	2	6	6	1069	115	20.00	0 :12 :13.75
8	17	15	4267	226	3	12	1	777	129	19.38	0 :17 :21.46
8	17	20	5856	391	3	12	7	731	135	20.74	0 :25 :56.44
9	22	5	1600	38	2	6	0	41	129	6.98	0 :00 :28.35
9	22	10	2059	120	1	4	0	33	163	0.61	0 :00 :30.78
9	22	15	5701	818	2	7	0	1735	203	7.39	1 :20 :00.15
9	22	20	3193	305	1	6	1	411	206	6.80	0 :27 :17.81
10	25	5	3043	37	2	5	0	73	129	6.98	0 :02 :59.43
10	25	10	1316	69	1	1	0	15	157	3.82	0 :00 :08.57
10	25	15	8220	1484	2	8	0	2015	207	7.25	2 :42 :47.08
10	25	20	10380	1105	2	8	0	1457	*243*	18.93	5 :00 :00.00
11	27	5	3977	48	2	6	0	83	129	6.98	0 :06 :15.44
11	27	10	2905	72	1	1	0	11	157	3.82	0 :01 :07.24
11	27	15	10351	1920	1	9	0	2017	*213*	9.86	5 :00 :00.00
11	27	20	13097	1602	1	12	0	1612	*273*	24.54	5 :00 :00.00
12	32	5	6497	84	1	4	0	97	129	8.53	0 :17 :26.27
12	32	10	13723	421	1	4	0	851	*182*	6.59	5 :00 :00.00
12	32	15	15291	512	1	10	6	813	*224*	9.82	5 :00 :00.00
12	32	20	12114	615	1	12	0	823	*292*	20.55	5 :00 :00.00
13	26	5	2229	15	0	4	0	27	129	6.98	0 :01 :40.33
13	26	10	12830	143	1	4	0	241	181	5.52	2 :13 :10.01
13	26	15	17401	321	0	2	4	495	*234*	13.25	5 :00 :00.00
14	27	5	4326	53	0	0	0	27	150	5.33	0 :07 :44.68
14	27	10	18212	652	1	4	0	281	*208*	10.10	5 :00 :00.00
15	28	5	9786	54	0	4	0	39	129	7.75	0 :15 :31.75
15	28	10	13896	121	0	1	0	87	191	5.24	1 :46 :42.95
16	29	5	18949	338	0	3	0	119	*158*	13.92	5 :00 :00.00
16	29	10	13563	80	0	3	0	23	191	5.76	0 :17 :14.14
17	29	5	22872	259	1	2	0	99	*192*	20.31	5 :00 :00.00
18	30	5	23332	43	1	3	0	77	*231*	22.08	5 :00 :00.00

TAB. 6.10 – Résultats pour les instances réelles du problème MCSIPND$_s$ avec la fonction coût $c_1(.)$ et la formulation arcs-chemins

est obligatoirement supérieure ou égale à la solution optimale pour la même instance mais pour le problème MCSIPND$_m$. Ceci vient du fait que toute solution réalisable pour le problème MCSIPND$_s$ est réalisable pour MCSIPND$_m$. Par contre, pour certaines instances, la solution optimale pour MCSIPND$_s$ n'est pas égale à celle pour le problème MCSIPND$_m$. C'est par exemple le cas pour l'instance à 10 sommets et 10 demandes ou celle à 29 sommets et 10 demandes.

Les résultats regroupés dans la table 6.10 correspondant à l'algorithme de coupes, génération de colonnes et branchements basé sur la formulation arcs-chemins du problème MCSIPND$_s$ sont bien meilleurs que ceux de la table 6.9 portant sur la formulation arcs-sommets. En effet, de nombreuses instances qui n'avaient pas été résolues le sont à présent et en moins de 30 minutes pour certaines d'entre-elles comme celle à 8 sommets et 20 demandes ou celle à 9 sommets et 20 demandes. De plus, le temps CPU nécessaire est moins grand pour toutes les instances. Pour les instances non résolues excepté celles à 17 et 18 sommets, la meilleure solution réalisable trouvée est également bien meilleure, passant par exemple de 317 à 292 pour l'instance à 12 sommets et 20 demandes. Le nombre de variables de flot générées est moins grand et le nombre de contraintes de tous types est similaire entre les deux tables.

Fonction coût $c_2(.)$

La table 6.11 regroupe les résultats pour le problème MCSIPND$_s$, la formulation arcs-sommets et les instances réelles utilisant cette fois la fonction coût $c_2(.)$.

Comme pour le problème MCSIPND$_m$, pour le problème MCSIPND$_s$, les instances utilisant la fonction coût $c_2(.)$ semblent plus difficiles à résoudre que celles utilisant la fonction coût c_1. De même le problème MCSIPND$_s$ semble plus facile à résoudre que le problème MCSIPND$_m$ pour ces instances.

Aucune instance ayant plus de 15 sommets n'a été résolue et aucune non plus au dessus de 11 sommets pour plus de 10 demandes. Néanmoins pour toutes ces instances ont dispose d'une solution réalisable mais avec un gap entre 15 et 25%.

Les résultats pour le problème MCSIPND$_s$, la fonction coût $c_2(.)$ et la formulation arcs-chemins sont contenus dans la table 6.12. Comme pour la fonction coût $c_1(.)$ les résultats obtenus à l'aide de la formulation arcs-chemins sont bien meilleurs que ceux obtenus avec la formulation arcs-sommets. 6 instances qui n'avaient pas été résolues à l'optimum le sont à présent et les temps d'exécution sont également plus petits la plupart du temps.

$\|V^1\|$	$\|\mathcal{F}\|$	$\|K\|$	VF	NC	NCR	NS	NCC	NEP	NT	Copt	Gap	TT
6	11	5	1650	10	31	1	1	0	5	4401.00	9.67	0 :00 :01.10
6	11	10	3300	35	120	1	2	0	23	7399.00	7.88	0 :00 :17.76
6	11	15	4950	53	156	1	1	0	15	7537.00	7.37	0 :00 :21.16
6	11	20	6600	63	187	1	1	0	19	8330.00	2.69	0 :00 :35.38
7	14	5	2940	19	23	2	3	1	9	4401.00	9.67	0 :00 :04.48
7	14	10	5880	111	271	2	8	1	43	7372.00	12.75	0 :01 :25.05
7	14	15	8820	81	546	3	9	4	39	7172.00	9.99	0 :02 :05.13
7	14	20	11760	152	698	3	5	2	85	8236.00	12.96	0 :09 :48.06
8	17	5	4760	143	228	1	0	0	185	5645.00	18.77	0 :04 :43.34
8	17	10	9520	99	412	2	4	3	69	6838.00	18.01	0 :04 :11.89
8	17	15	14280	190	1535	3	10	0	411	8047.00	19.94	1 :31 :07.89
8	17	20	19040	150	1996	3	9	1	265	8080.00	18.41	1 :44 :53.21
9	22	5	7620	48	172	2	3	0	73	8334.00	15.67	0 :06 :13.92
9	22	10	15840	253	326	1	2	0	199	10471.00	8.46	0 :39 :16.62
9	22	15	23760	169	1018	2	2	0	565	*12417.00*	7.63	5 :00 :00.00
9	22	20	31680	128	795	1	1	0	227	12685.00	7.35	3 :43 :48.55
10	25	5	11250	57	210	2	2	0	129	8334.00	15.47	0 :17 :10.14
10	25	10	23400	255	492	2	1	0	371	10742.00	8.99	2 :17 :08.21
10	25	15	33750	251	1336	2	3	0	481	*13071.00*	10.06	5 :00 :00.00
10	25	20	45000	233	2046	1	5	1	277	*13599.00*	12.56	5 :00 :00.00
11	27	5	14850	56	282	2	3	0	107	8334.00	15.31	0 :25 :55.79
11	27	10	29700	179	721	1	2	0	463	*11220.00*	8.77	5 :00 :00.00
11	27	15	445500	131	1028	1	4	0	275	*13593.00*	9.49	5 :00 :00.00
11	27	20	59400	247	1284	1	6	0	361	*15362.00*	15.47	5 :00 :00.00
12	32	5	21120	54	457	1	4	0	183	8334.00	15.67	1 :34 :35.69
12	32	10	42240	292	678	1	3	2	297	*11917.00*	12.88	5 :00 :00.00
12	32	15	63360	258	1313	1	5	1	135	*16537.00*	24.35	5 :00 :00.00
12	32	20	84480	175	1585	1	3	0	79	*17135.00*	18.86	5 :00 :00.00
13	26	5	20280	152	502	0	5	0	419	8553.00	15.96	3 :04 :25.24
13	26	10	40560	274	407	0	1	0	225	*11225.00*	6.83	5 :00 :00.00
13	26	15	60840	163	1195	1	3	1	151	*16439.00*	23.11	5 :00 :00.00
14	27	5	24570	21	34	0	0	0	35	9153.00	2.60	0 :26 :25.84
14	27	10	49140	292	294	0	1	0	147	12293.00	3.57	3 :35 :35.94
15	28	5	29400	182	679	0	4	0	549	*9321.00*	22.88	5 :00 :00.00
15	28	10	58800	192	610	0	2	0	155	*12950.00*	15.86	5 :00 :00.00
16	29	5	34800	469	469	0	3	0	361	*9796.00*	22.19	5 :00 :00.00
16	29	10	69600	232	374	0	4	0	105	*14168.00*	26.27	5 :00 :00.00
17	29	5	39440	75	258	1	2	0	295	*8958.00*	12.64	5 :00 :00.00
18	30	5	45900	60	348	0	2	0	229	*11266.00*	26.25	5 :00 :00.00

TAB. 6.11 – Résultats pour les instances réelles du problème MCSIPND$_s$ avec la fonction coût $c_2(.)$ et la formulation arcs-sommets

$\|V^1\|$	$\|\mathcal{F}\|$	$\|K\|$	VF	NC	NS	NCC	NEP	NT	Copt	Gap	TT
6	11	5	287	9	1	2	0	5	4401.00	8.40	0 :00 :00.85
6	11	10	613	46	1	2	0	23	7399.00	7.88	0 :00 :02.75
6	11	15	934	52	1	2	1	15	7537.00	7.00	0 :00 :03.29
6	11	20	1161	57	1	2	0	19	8330.00	2.69	0 :00 :03.44
7	14	5	618	15	2	3	1	9	4401.00	9.67	0 :00 :02.34
7	14	10	1126	84	2	9	3	41	7372.00	12.55	0 :00 :09.17
7	14	15	1664	90	2	9	2	31	7172.00	9.97	0 :00 :16.36
7	14	20	2702	138	2	5	2	87	8236.00	12.96	0 :01 :08.93
8	17	5	2737	106	1	1	0	219	5645.00	18.77	0 :02 :20.82
8	17	10	1788	80	2	4	6	53	6838.00	16.04	0 :00 :23.81
8	17	15	3753	151	3	9	1	155	8047.00	16.66	0 :03 :54.33
8	17	20	4378	133	3	7	1	101	8080.00	16.14	0 :03 :21.22
9	22	5	3630	61	2	2	0	121	8334.00	14.22	0 :04 :25.38
9	22	10	3519	240	1	3	0	119	10471.00	7.59	0 :05 :49.41
9	22	15	5271	186	2	5	0	429	12417.00	7.61	0 :20 :23.02
9	22	20	4869	151	1	3	0	197	12685.00	7.06	0 :08 :33.17
10	25	5	6199	61	2	3	0	157	8334.00	13.92	0 :15 :33.31
10	25	10	8166	187	2	3	0	291	10742.00	8.66	0 :49 :12.70
10	25	15	9514	527	2	7	0	1049	12901.00	8.76	2 :03 :58.82
10	25	20	10339	163	2	7	0	361	13548.00	12.23	1 :00 :53.99
11	27	5	8615	53	2	3	0	107	8334.00	14.28	0 :28 :23.10
11	27	10	12476	127	1	1	0	457	11205.00	8.63	2 :21 :31.47
11	27	15	12406	180	1	3	0	1507	*13314.00*	6.98	5 :00 :00.00
11	27	20	12209	417	1	6	2	1473	14323.00	8.70	3 :41 :31.88
12	32	5	12840	62	1	3	0	143	8334.00	14.63	1 :49 :33.19
12	32	10	21116	408	1	7	3	491	*11741.00*	11.58	5 :00 :00.00
12	32	15	20435	632	1	10	3	611	*14630.00*	14.43	5 :00 :00.00
12	32	20	18207	423	1	8	0	635	*15830.00*	11.54	5 :00 :00.00
13	26	5	18616	145	0	5	0	341	*8553.00*	14.41	5 :00 :00.00
13	26	10	19364	202	0	5	0	209	11221.00	6.46	3 :20 :44.92
13	26	15	22133	550	1	7	2	497	*14024.00*	10.10	5 :00 :00.00
14	27	5	4625	22	0	0	0	23	9153.00	1.37	0 :01 :58.09
14	27	10	18057	299	0	2	0	139	12293.00	3.57	2 :35 :41.81
15	28	5	22522	103	0	5	0	133	*10101.00*	28.10	5 :00 :00.00
15	28	10	26227	292	0	4	2	159	*13111.00*	15.93	5 :00 :00.00
16	29	5	27822	140	0	3	0	63	*10046.00*	23.81	5 :00 :00.00
16	29	10	27847	375	0	3	0	103	*12872.00*	18.20	5 :00 :00.00
17	29	5	28478	51	1	1	0	42	*10072.00*	21.78	5 :00 :00.00
18	30	5	26380	49	0	2	0	63	*11529.00*	26.56	5 :00 :00.00

TAB. 6.12 – Résultats pour les instances réelles du problème MCSIPND$_s$ avec la fonction coût $c_2(.)$ et la formulation arcs-chemins

6.2.5 Instances non résolues au bout de 5 heures

Pour certaines instances présentées dans les sections précédentes, nous n'avons pas obtenu de solution optimale. Nous avons considéré l'évolution de la meilleure solution réalisable pour les instances non résolues au bout de 5 heures. Les tables 6.13 et 6.14 regroupent des résultats permettant de suivre cette évolution. Les différentes colonnes de ces tables représentent :

$\|V^1\|$	:	le nombre de sommets de G^1,
$\|\mathcal{F}\|$	:	le nombre d'ensembles F_e,
$\|K\|$	:	le nombre de demandes,
Algo	:	l'algorithme considéré (as représente celui basé sur la formulation arcs-sommets et ac celui basé sur la formulation arcs-chemins),
Coût	:	la fonction coût considérée,
MS	:	la meilleure solution trouvée,
Gap2	:	l'erreur relative entre la meilleure solution réalisable et la meilleure borne inférieure obtenues.

Les données MS et Gap2 sont données toutes les heures, entre une heure d'exécution et cinq heures.

La table 6.13 regroupe les résultats obtenus pour le problème MCSIPND$_m$. Les lignes sont groupées par deux, une ligne pour la formulation arcs-sommets et une ligne pour la formulation arcs-chemins. Lorsqu'il n'y a qu'une ligne, c'est que la deuxième formulation a permis de trouver une solution optimale en moins de 5 heures. Le haut de la table concerne les instances avec la fonction coût $c_1(.)$ et le bas la fonction coût $c_2(.)$. Le gap2 représente la qualité de la meilleure solution trouvée.

On remarque que même si une instance n'a pas été résolue en moins de 5 heures, la meilleure solution trouvée est souvent de bonne qualité. Par exemple, pour l'instance à 14 sommets et la fonction coût $c_1(.)$, le gap est de 0.51%. Pour l'instance à 16 sommets et la fonction $c_1(.)$, la formulation arcs-sommets a permis de trouver une solution réalisable avec un gap de 1.32%. On peut faire la même remarque pour les instances avec la fonction coût $c_2(.)$. En effet, de nombreuses instances donnent des solutions réalisables avec un gap inférieur à 10%. On distingue par exemple l'instance à 13 sommets ou celle à 10 sommets qui ont un gap inférieur à 0.5%.

Néanmoins, pour certaines instances plus difficiles, la meilleure solution trouvée est peut être encore assez éloignée de la solution optimale. En effet, l'instance à 15 sommets et la fonction coût $c_2(.)$ a un gap de 38% au bout de 5 heures, que ce soit avec la formulation arcs-sommets qu'avec la formulation arcs-chemins.

					Après 1h		Après 2h		Après 3h		Après 4h		Après 5h	
$\|V^1\|$	$\|\mathcal{F}\|$	$\|K\|$	Algo	Coût	MS	Gap2	MS	Gap2	MS	Gap2	MS	Gap2	MS	Gap2
10	25	15	as	$c_1(.)$	243	25,91	231	19,07	231	18,46	231	18,46	231	18,46
10	25	15	ac	$c_1(.)$	218	10,66	218	10,10	216	8,54	213	7,04	213	7,04
11	27	10	as	$c_1(.)$	157	2,61	157	1,95	157	1,95	157	1,29	157	1,29
12	32	10	as	$c_1(.)$	272	57,23	207	18,29	207	17,61	194	10,23	194	9,60
12	32	10	ac	$c_1(.)$	189	8,00	182	2,82	182	2,82	182	2,25	182	2,25
13	26	10	as	$c_1(.)$	270	56,07	234	33,71	234	33,71	221	25,57	221	25,57
14	27	10	ac	$c_1(.)$	246	26,15	198	1,02	198	1,02	198	1,02	198	0,51
15	28	10	as	$c_1(.)$	251	37,91	239	29,89	195	5,41	195	5,41	195	5,41
15	28	10	ac	$c_1(.)$	247	35,71	245	33,15	225	21,62	223	20,54	219	18,38
16	29	5	as	$c_1(.)$	202	36,49	159	7,43	159	6,00	153	2,00	153	1,32
16	29	5	ac	$c_1(.)$	202	36,49	167	12,84	167	12,84	167	12,08	159	6,71
17	29	5	as	$c_1(.)$	248	55,97	232	43,21	189	15,24	180	9,09	180	8,43
17	29	5	ac	$c_1(.)$	201	29,68	201	27,22	201	26,42	201	25,63	201	24,84
18	30	5	as	$c_1(.)$	323	73,66	264	39,68	239	25,13	218	13,54	218	12,95
18	30	5	ac	$c_1(.)$	222	21,98	222	20,65	222	19,35	222	17,46	222	17,46
10	25	15	as	$c_2(.)$	15532,00	30,63	14018,00	16,94	13981,00	16,24	13981,00	15,76	12987,00	7,35
10	25	15	ac	$c_2(.)$	13072,00	6,57	12552,00	1,73	12552,00	1,45	12460,00	0,47	12460,00	0,33
11	27	10	as	$c_2(.)$	11650,00	10,71	11650,00	9,63	11650,00	8,97	11529,00	7,50	11529,00	7,09
11	27	10	ac	$c_2(.)$	11650,00	8,48	11322,00	4,73	11322,00	4,28	11052,00	1,45	11052,00	1,31
12	32	10	as	$c_2(.)$	14601,00	34,77	14280,00	30,43	13311,00	20,58	12897,00	16,51	12597,00	13,54
13	26	10	as	$c_2(.)$	14088,00	32,19	13374,00	23,64	13098,00	20,57	11775,00	8,18	11775,00	8,06
13	26	10	ac	$c_2(.)$	13310,00	22,76	12287,00	12,95	12030,00	10,27	11019,00	0,80	10980,00	0,24
14	27	10	as	$c_2(.)$	19854,00	66,57	13980,00	16,47	13486,00	11,98	12303,00	1,76	12303,00	1,43
14	27	10	ac	$c_2(.)$	16095,00	34,16	15804,00	30,97	15352,00	26,55	12942,00	6,29	12942,00	6,12
15	28	10	as	$c_2(.)$	16127,00	45,85	16127,00	44,97	16127,00	44,36	15635,00	39,12	15635,00	38,77
15	28	10	ac	$c_2(.)$	17762,00	59,43	16362,00	46,41	16362,00	46,08	16362,00	45,78	15588,00	38,55
16	29	5	as	$c_2(.)$	10650,00	33,21	9722,00	18,72	9511,00	15,48	9511,00	14,75	9511,00	13,86
16	29	5	ac	$c_2(.)$	9312,00	21,27	9312,00	16,87	8869,00	10,93	8869,00	10,65	8810,00	9,83
17	29	5	as	$c_2(.)$	11469,00	41,99	11469,00	39,86	11469,00	38,25	9411,00	12,88	8904,00	6,30
17	29	5	ac	$c_2(.)$	9813,00	26,03	9813,00	25,71	9813,00	24,55	9813,00	24,45	9813,00	23,79
18	30	5	as	$c_2(.)$	13683,00	69,01	11325,00	36,12	11325,00	34,65	9162,00	8,06	9162,00	7,75
18	30	5	ac	$c_2(.)$	9936,00	27,56	9936,00	23,79	9936,00	23,79	9936,00	23,31	9936,00	21,79

TAB. 6.13 – Instances non résolues pour le problème MCSIPND$_m$

					Après 1h		Après 2h		Après 3h		Après 4h		Après 5h	
$\|V^1\|$	$\|\mathcal{F}\|$	$\|K\|$	Algo	Coût	MS	Gap2	MS	Gap2	MS	Gap2	MS	Gap2	MS	Gap2
10	25	15	as	$c_1(.)$	240	21,21	227	13,50	227	12,94	227	12,94	223	10,95
11	27	15	as	$c_1(.)$	223	13,20	223	12,63	223	12,06	223	12,06	223	12,06
11	27	15	ac	$c_1(.)$	217	7,96	217	7,96	213	4,93	213	4,41	213	4,41
12	32	10	as	$c_1(.)$	217	24,71	201	14,20	193	8,43	186	3,91	186	3,91
12	32	10	ac	$c_1(.)$	188	5,62	182	1,68	182	1,11	182	1,11	182	1,11
13	26	15	as	$c_1(.)$	294	40,67	269	27,49	242	14,69	242	14,15	242	14,15
13	26	15	ac	$c_1(.)$	249	17,45	243	14,08	234	9,35	234	8,84	234	8,84
14	27	10	as	$c_1(.)$	226	17,10	226	15,90	226	13,57	226	13,00	226	12,44
14	27	10	ac	$c_1(.)$	225	12,50	225	11,94	214	5,94	208	2,46	208	1,96
15	28	10	as	$c_1(.)$	218	19,13	218	17,20	215	14,36	215	13,76	215	13,16
16	29	5	as	$c_1(.)$	163	7,95	163	5,84	163	5,16	158	1,28	158	0,64
16	29	5	ac	$c_1(.)$	169	14,19	169	11,92	163	7,24	163	6,54	163	5,84
17	29	5	as	$c_1(.)$	214	29,70	214	28,14	190	13,10	190	12,43	190	11,76
17	29	5	ac	$c_1(.)$	192	19,25	192	17,79	192	16,36	192	15,66	192	15,66
18	30	5	as	$c_1(.)$	241	25,52	211	8,21	211	7,11	211	6,57	211	6,03
18	30	5	ac	$c_1(.)$	231	20,94	231	20,31	231	19,69	231	19,69	231	19,07
10	25	15	as	$c_2(.)$	13910,00	12,92	13579,00	9,44	13431,00	7,77	13431,00	7,46	13071,00	4,36
11	27	15	as	$c_2(.)$	13850,00	8,29	13636,00	6,14	13636,00	5,85	13636,00	5,64	13593,00	5,16
11	27	15	ac	$c_2(.)$	13526,00	3,82	13314,00	1,44	13314,00	0,87	13314,00	0,39	13314,00	0,04
12	32	10	as	$c_2(.)$	13648,00	25,04	11917,00	8,12	11917,00	7,27	11917,00	6,66	11917,00	6,36
12	32	10	ac	$c_2(.)$	12426,00	11,45	12426,00	10,14	12426,00	9,00	11741,00	2,57	11741,00	2,29
13	26	15	as	$c_2(.)$	16655,00	28,70	16655,00	27,41	16655,00	26,49	16439,00	24,47	16439,00	24,02
13	26	15	ac	$c_2(.)$	16378,00	24,81	15296,00	14,79	14786,00	9,90	14786,00	9,60	14024,00	3,77
15	28	10	as	$c_2(.)$	14770,00	33,76	14714,00	31,37	14018,00	24,58	14018,00	24,22	12950,00	14,34
15	28	10	ac	$c_2(.)$	14142,00	24,45	13190,00	15,82	13190,00	15,49	13190,00	14,98	13111,00	14,04
16	29	5	as	$c_2(.)$	10941,00	34,26	9796,00	18,53	9796,00	17,26	9796,00	16,60	9796,00	16,02
16	29	5	ac	$c_2(.)$	10046,00	27,25	10046,00	24,76	10046,00	24,66	10046,00	24,34	10046,00	23,65
17	29	5	as	$c_2(.)$	9296,00	11,75	9296,00	10,17	9296,00	8,95	8958,00	4,50	8958,00	4,16
17	29	5	ac	$c_2(.)$	10227,00	26,40	10227,00	25,29	10072,00	23,08	10072,00	23,08	10072,00	23,01
18	30	5	as	$c_2(.)$	12405,00	43,22	11507,00	30,41	11507,00	27,55	11266,00	23,14	11266,00	22,67
18	30	5	ac	$c_2(.)$	11529,00	34,97	11529,00	33,21	11529,00	31,75	11529,00	31,13	11529,00	30,93

TAB. 6.14 – Instances non résolues pour le problème MCSIPND$_s$

Le gap diminue au cours du temps. Cela vient de la diminution de la meilleure solution réalisable, mais aussi de l'augmentation de la borne inférieure globale. Cette diminution du gap peut être lente et régulière ou très rapide selon les instances. Par exemple, l'instance à 15 sommets et la fonction coût $c_1(.)$ voit son gap diminuer brutalement entre 2 et 3 heures d'exécution avec la formulation arcs-sommets. Ceci est dû à une amélioration significative de la meilleure solution trouvée qui passe de 239 à 195. On remarque ce genre d'amélioration brusque également pour les instances avec le coût $c_2(.)$, par exemple pour l'instance à 14 sommets qui voit son gap passer de 66.57% à 16.47% entre 1 et 2 heures.

Une dernière remarque que l'on peut faire à propos de ces résultats est qu'une très bonne solution réalisable a déjà pu être trouvée au bout d'un temps bien inférieur à 5 heures. Par exemple, pour l'instance à 11 sommets et la fonction coût $c_1(.)$, une solution avec un gap de 2.61% a été trouvée au bout d'une heure seulement et pour celle à 14 sommets, deux heures ont suffi pour obtenir une solution avec un gap de 1.02%. De même, pour la fonction coût $c_2(.)$, on a obtenu un gap à 1.73% au bout de 2 heures pour l'instance à 10 sommets.

La table 6.14 est formée sur le même principe que la table 6.13 pour les instances non résolues du problème MCSIPND$_s$. Les mêmes remarques que celles sur les résultats de la table 6.13 peuvent être faites. De nombreuses instances ont un gap au bout de 5 heures inférieur à 10%. On dispose également d'instances comme l'instance à 18 sommets et la fonction coût $c_1(.)$ dont le gap a diminué brusquement entre 1 et 2 heures passant de 25.52% à 8.21%. De même pour certaines instances un gap très faible a été trouvé après seulement 2 heures de calcul. C'est par exemple le cas de l'instance à 12 sommets et la fonction coût $c_1(.)$ ou de l'instance à 11 sommets et la fonction coût $c_2(.)$.

6.2.6 Une instance française

Nous présentons à présent plus en détail, une instance composée de 10 sommets représentant des villes françaises comme Paris, Lyon, Marseille ou Bordeaux. La couche transport comprend 25 liaisons entre ces villes. On dispose également d'un ensemble de 10 demandes. La figure 6.1 représente la couche transport et la figure 6.2 représente les différentes demandes. La valuation sur les arcs représente le volume des demandes.

Les figures 6.3 et 6.4 donnent les solutions trouvées pour les deux versions du problème pour les fonctions coûts $c_1(.)$ et $c_2(.)$ respectivement. Les traits pleins représentent les liaisons de capacité 10 Gbit/s installées, les traits en pointillés représentent

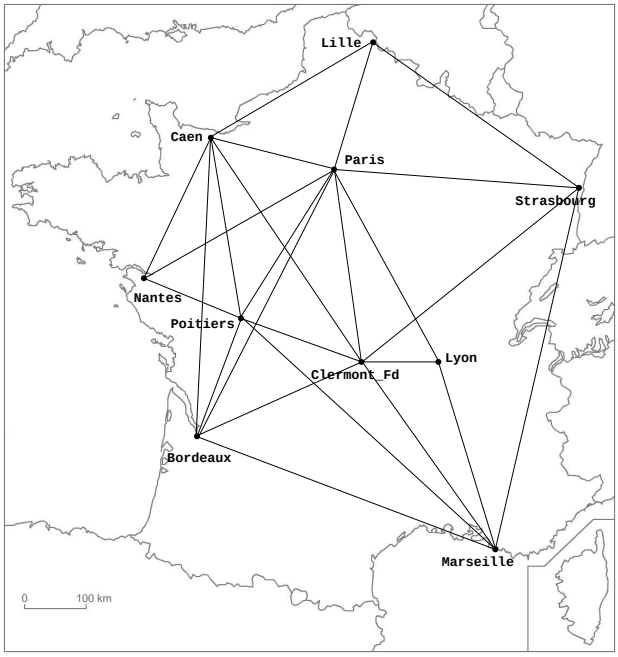


FIG. 6.1 – Couche transport pour l'instance réelle à 20 sommets, $|\mathcal{F}| = 25$ et 10 demandes

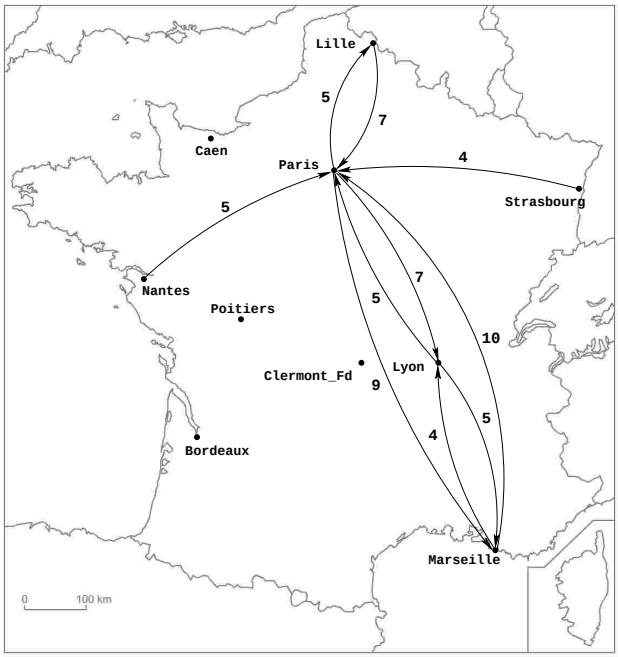
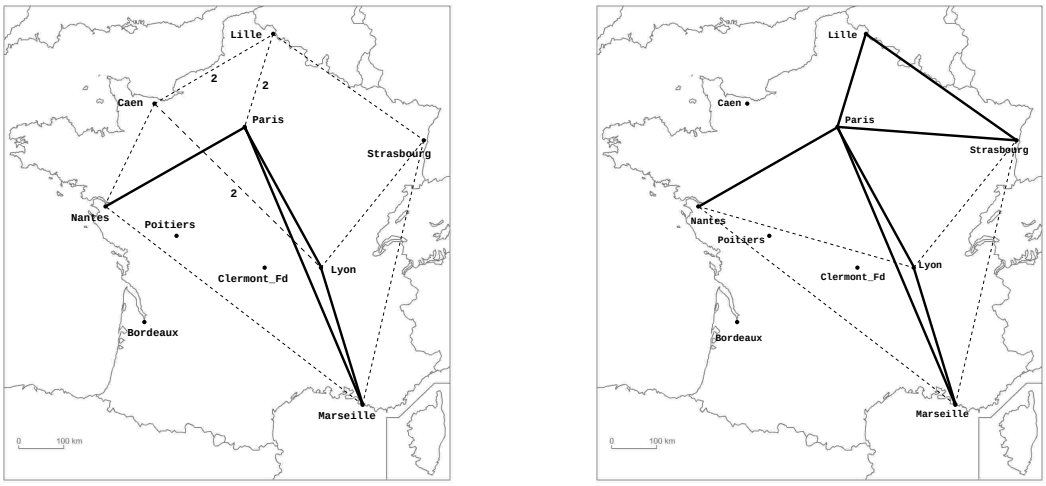


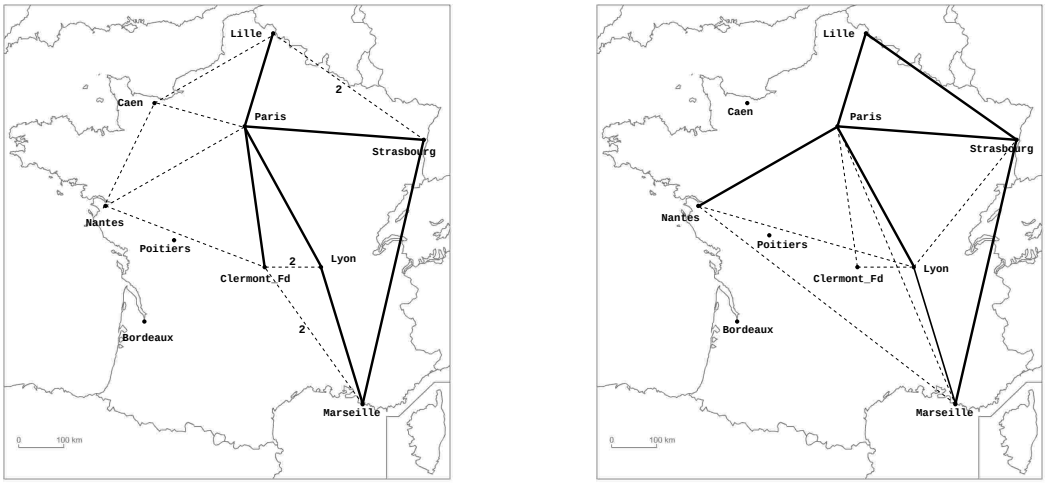
FIG. 6.2 – Demandes pour l'instance réelle à 20 sommets, $|\mathcal{F}| = 25$ et 10 demandes

celles de capacité 2.5 Gbit/s. Pour les solutions du problème MCSIPND$_m$, lorsque plusieurs liaisons de même capacités sont installées entre deux sommets, leur nombre est inscrit juste à côté.



problème MCSIPND$_m$ problème MCSIPND$_s$

FIG. 6.3 – Solutions pour la couche cliente avec la fonction coût $c_1(.)$



problème MCSIPND$_m$ problème MCSIPND$_s$

FIG. 6.4 – Solutions pour la couche cliente avec la fonction coût $c_2(.)$

Toutes les solutions optimales représentées par les figures 6.3 et 6.4 ont été obtenues à l'aide des algorithmes de coupes et branchements basés sur la formulation arcs-sommets, comme avec les algorithmes de coupes, génération de colonnes et branchements basés sur la formulation arcs-chemins. En effet, pour cette instance, la solution est identique. Ceci n'est pas toujours le cas.

6.3 Conclusion

Dans ce chapitre, nous avons décrit pour chaque version du problème MCSIPND (avec ou sans liaisons multiples), un algorithme de coupes et branchements basé sur la formulation arcs-sommets du problème et un algorithme de coupes, génération de colonnes et branchement basé sur sa formulation arcs-chemins. Nous avons appliqué ces algorithmes à des instances de la TSP Library [76] et nous avons également pu résoudre des instances réelles fournies par France Télécom. Il semble, d'après nos résultats expérimentaux que les instances réelles soient plus faciles à résoudre que les instances aléatoires. De même, le problème MCSIPND$_s$ est plus facile à résoudre que le problème MCSIPND$_m$. Les contraintes de saturation qui ne sont valides que pour le premier permettent en effet d'accélérer la résolution. Dans la plupart des cas, l'algorithme basé sur la formulation arcs-chemins est plus performant que celui basé sur la formulation arcs-sommets. C'est d'autant plus significatif pour le problème MCSIPND$_s$ pour lequel la formulation arcs-chemins demande moins de temps CPU, permet de résoudre plus d'instances et pour celles qui n'ont pas été résolues, donne une très bonne solution réalisable. Lorsque nos algorithmes n'ont pas pu résoudre certaines instances dans le temps qui leur était imparti, une très bonne solution heuristique a été trouvée avec un gap entre cette solution et la borne inférieure souvent très faible. Cette solution a parfois été obtenue après seulement 1 ou 2 heures d'exécution. En approfondissant cette étude, il sera possible de résoudre des instances de plus grande taille et de diminiuer le temps de calcul.

Chapitre 7

Sécurisation multicouche du réseau optique et sécurisation simultanée

Dans ce chapitre, nous considérons un problème de sécurisation multicouche du réseau optique et un problème de sécurisation simultanée des deux couches du réseau. Nous présentons tout d'abord ces problèmes ainsi qu'une formulation en termes de programmes linéaires en nombres entiers pour chacun d'eux. Nous décrivons également une extension du deuxième problème. Nous présentons ensuite deux métaheuristiques de type recuit simulé pour résoudre ces problèmes de manière approchée. Nous donnons également des résultats expérimentaux obtenus à l'aide de ces algorithmes.

7.1 Sécurisation multicouche du réseau optique

7.1.1 Présentation du problème MSOND

Dans les problèmes MSIPND et MCSIPND, nous avons supposé que la couche transport était fixée et sécurisée, et nous avons cherché à sécuriser la couche cliente. Dans ce nouveau problème, on suppose que la couche cliente est fixée et l'on désire sécuriser la couche transport. Considérons, par exemple, le modèle Overlay où les plans de commande et de transfert sont séparés. Si le routage dans la couche cliente est établi suivant les demandes et une certaine fiabilité, on peut se poser le problème de la sécurisation de la couche transport suivant le routage de la couche cliente.

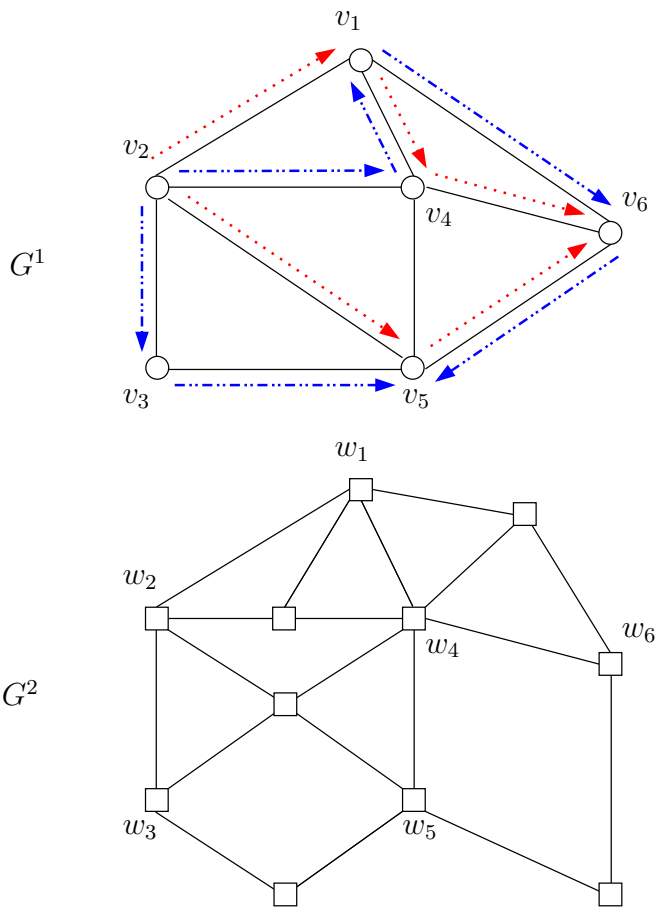
En d'autres termes, supposons qu'un ensemble de demandes entre des routeurs IP est

défini et supposons également que deux chemins LSP sommet-disjoints sont définis pour chaque demande entre deux routeurs. A chaque routeur LSR est associé un brasseur OXC de la couche optique. Chaque liaison de la couche optique porte un certain coût fixe (coût d'installation). On dit que le chemin (OCT) de la couche transport associé à un LSP de la couche cliente, respecte l'ordre des routeurs du LSP, si cet OCT emprunte les brasseurs OXC associés aux routeurs du LSP de la couche cliente dans l'ordre dans lequel ils apparaissent dans le LSP correspondant. Le chemin de la couche transport peut emprunter d'autres brasseurs OXC, qu'ils soient associés ou non à un routeur LSR. Le *problème de sécurisation de la couche optique* (MSOND Problem, *Multilayer Survivable Optical Network Design Problem*) consiste à déterminer, pour chaque LSP du réseau IP, un chemin OCT dans la couche optique respectant l'ordre des routeurs du LSP, de telle manière que deux chemins correspondant à la même demande soient sommet-disjoints et le coût total soit minimum.

Ce problème peut être modélisé de la manière suivante. On associe à la couche cliente, un graphe $G^1 = (V^1, E^1)$ où les sommets de V^1 correspondent aux routeurs et les arêtes aux liaisons entre les routeurs. Notons que pour chaque demande, on a deux chemins dans G^1 entre les sommets associés aux routeurs extrémités de la demande. Ces chemins représentent les LSP associés à la demande. On associe à la couche transport, un graphe $G^2 = (V^2, E^2)$ où les sommets de V^2 correspondent aux brasseurs OXC et les arêtes aux fibres optiques. A chaque sommet $v_i \in V^1$, $i = 1, \ldots, 6$ est associé un sommet $w_i \in V^2$. On suppose aussi que chaque arête e de G^2 est munie d'un coût fixe $c(e) > 0$. Le problème consiste donc à déterminer, pour chaque chemin dans G^1, un chemin dans G^2 entre les brasseurs associés aux routeurs extrémités, en suivant l'ordre des routeurs du chemin de G^1 associé, de telle manière que deux chemins de G^2 associés à la même demande soient sommet-disjoints et le coût total soit minimum.

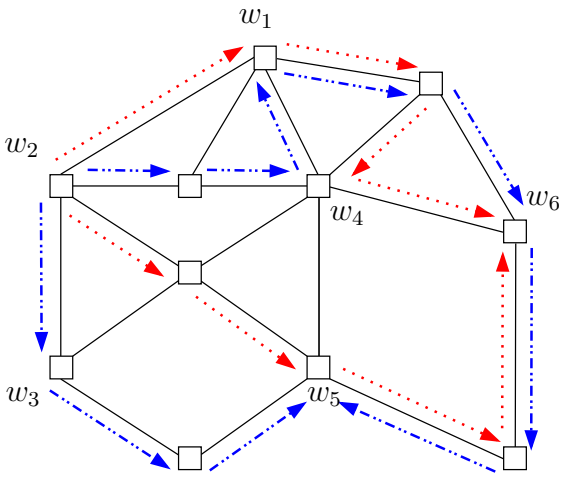
La figure 7.1 présente un exemple de problème de sécurisation de la couche optique. La couche cliente est représentée par le graphe G^1. Les sommets sont nommés $v_1, \ldots, v_6$. Nous considérons ici deux demandes, la première allant de v_2 à v_6 et la deuxième de v_2 à v_5. Les LSP de la couche cliente sont connus et représentés par les flèches, chaque type correspondant à une demande. La couche transport est représentée par le graphe G^2. Chaque sommet $v_i \in V^1$ est associé au sommet $w_i \in V^2$. Les autres sommets de V^2 ne sont pas nommés.

La figure 7.2 donne une solution réalisable du problème. Les deux chemins représentant une même demande sont bien disjoints et respectent l'ordre des routeurs dans le chemin de la couche cliente. En effet, pour la première demande entre v_2 et v_5, le premier chemin emprunte dans l'ordre w_4 puis w_1 et w_6 pour arriver à w_5 et le deuxième passe par w_3 pour rejoindre w_5. Il en est de même pour la demande entre v_2 et v_6.



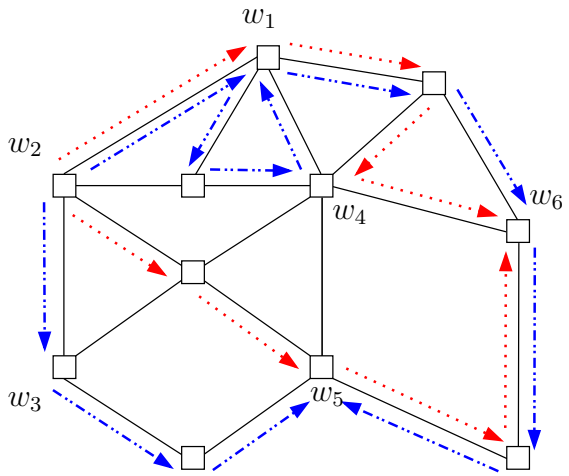
Origine	Dest.	LSP
v_2	v_5	
v_2	v_6	

FIG. 7.1 – Exemple du problème



Origine	Dest.	LSP
v_2	v_5	
v_2	v_6	

FIG. 7.2 – Solution réalisable du problème



Origine	Dest.	LSP
v_2	v_5	
v_2	v_6	

FIG. 7.3 – Autre solution réalisable du problème

La figure 7.3 donne également une solution réalisable du problème. Les deux chemins représentant une même demande sont bien disjoints. De même, ils respectent bien l'ordre des routeurs. En effet, pour la demande entre v_2 et v_6, les chemins respectent bien l'ordre des routeurs. Pour la demande entre v_2 et v_5, il y a bien un chemin passant par w_3 pour rejoindre w_5. Le deuxième chemin emprunte bien les sommets w_4, w_1 et w_6 pour atteindre w_5. On peut cependant remarquer que ce chemin passe par w_1 entre w_2 et w_4, alors qu'il n'y est pas contraint. Ce chemin passe donc deux fois par le sommet w_1.

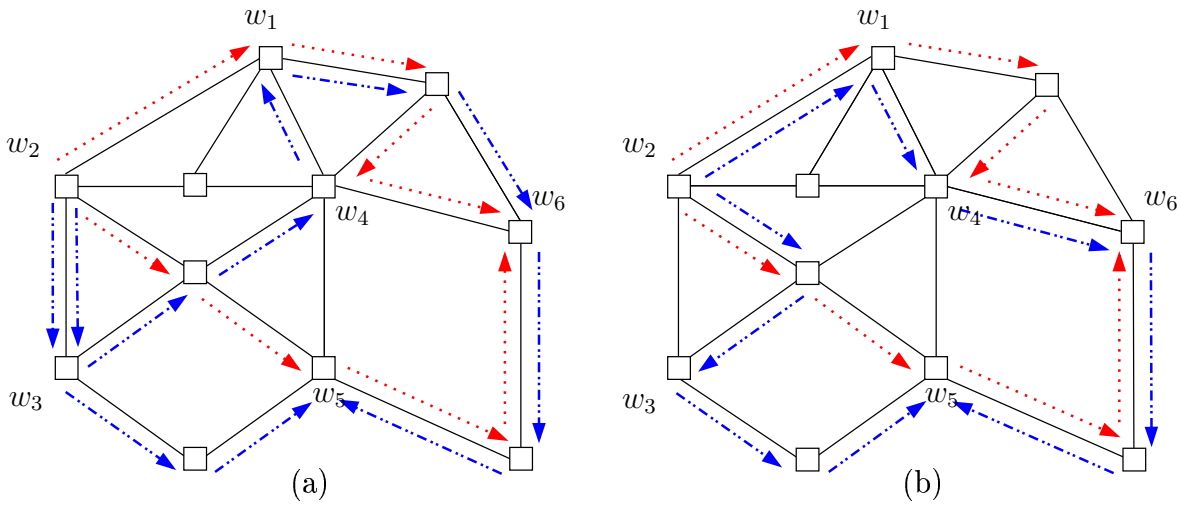


FIG. 7.4 – Solutions non réalisables du problème

Les chemins représentés sur la figure 7.4(a) ne forment pas une solution réalisable. Les deux chemins de la demande entre v_2 et v_5 passent tous les deux par w_3. Si ce

brasseur (ou le lien entre w_2 et w_3) venait à tomber en panne, la demande ne pourrait pas être acheminée suivant les chemins de la couche cliente. De même, la figure 7.4(b) ne donne pas une solution réalisable. Un des chemins représentant la demande entre v_2 et v_5 ne suit pas l'ordre des sommets imposé par le chemin de la couche cliente correspondant. En effet, le chemin ne passe jamais par w_4 entre w_2 et w_1.

Dans la section suivante, nous présentons une formulation sous la forme d'un programme linéaire en nombres entiers pour le problème MSOND.

7.1.2 Formulation du problème

Soit K l'ensemble des demandes. Pour $k \in K$, soit (O_k, D_k) la paire origine-destination de k. On note O'_k et D'_k, les brasseurs associés à O_k et D_k. Pour une demande $k \in K$, soient L^1_k et L^2_k les deux chemins sommet-disjoints dans G^1 entre O_k et D_k. Pour $e \in E^2$, soit $c(e) > 0$ le coût associé à l'arête e. Le problème MSOND consiste à déterminer pour tout $k \in K$, deux chemins sommet-disjoints dans G^2 entre O'_k et D'_k respectant l'ordre des sommets de L^1_k et L^2_k respectivement et tel que le coût soit minimum.

Soit $\hat{x} \in \mathbb{R}^{E^2}$ tel que $\hat{x}(e) = 1$ si l'arête e appartient à la solution et 0 sinon. Si $k \in K$ et $i \in \{1, 2\}$, soit $x^{i,k} \in \mathbb{R}^{E^2}$ un vecteur associé à k tel que $x^{i,k}(e) = 1$ si e appartient au $i^{\text{ème}}$ chemin de G^2 entre O'_k et D'_k correspondant à L^i_k. Pour une demande $k \in K$, on note $L^i_k = (v^{i,1}_k = O_k, v^{i,2}_k, \ldots, v^{i,t_i}_k = D_k)$, $i = 1, 2$. On note par $w^{i,j}_k$ le sommet de G^2 correspondant à $v^{i,j}_k$ pour $k \in K$, $i = 1, 2$ et $j = 1, \ldots, t_i$. Considérons le programme linéaire en nombres entiers suivant :

$$\text{Minimiser} \sum_{e \in E^2} c(e)\hat{x}(e)$$

$$(P) \quad \begin{array}{lll}
\sum_{e \in \delta_{G^2}(S)} x^{i,k}(e) \geq 1 & \forall\, k \in K,\ \forall\, S \subset V^2, & \\
& w^{i,j}_k \in S,\ w^{i,j+1}_k \in V^2 \backslash S, & (7.1) \\
& j = 1, \ldots, t_i - 1,\ i = 1, 2, & \\
\displaystyle\sum_{e \in \delta_{G^2}(u)} (x^{1,k}(e) + x^{2,k}(e)) \leq 2 & \forall\, u \in V^2 \backslash \{O'_k, D'_k\}, \quad \forall\, k \in K, & (7.2) \\
\displaystyle\sum_{k \in K} \sum_{i=1,2} x^{i,k}(e) \leq M\hat{x}(e) & \forall\, e \in E^2, & (7.3) \\
\hat{x}(e) \in \{0, 1\} & \forall\, e \in E^2, & (7.4) \\
x^{i,k}(e) \in \{0, 1\} & \forall\, e \in E^2, \quad \forall\, k \in K, \quad i = 1, 2. & (7.5)
\end{array}$$

Les contraintes (7.1) expriment le fait que les chemins entre O'_k et D'_k doivent respec-

ter l'ordre des routeurs. Les contraintes (7.2) imposent aux sommets de V^2 de n'être utilisés au maximum que par un seul des deux chemins d'une même demande (chemins sommet-disjoints). On suppose que M est choisi suffisamment grand. Les contraintes (7.3) permettent de relier les variables $\hat{x}$ et x.

Théorème 7.1 *Le problème MSOND est équivalent au problème* (P).

Preuve. Soit $(\hat{x}, x^{i,k},\ i = 1,2,\ k \in K)$ une solution du programme P ci-dessus. Soit $T_k^i = \{e \in E^2 \mid x^{i,k}(e) = 1\}$ pour $k \in K$, $i = 1,2$. T_k^i contient un chemin entre O_k et D_k respectant l'odre des routeurs du chemin L_k^i. Ainsi T_k^i contient un chemin $T_k^{i,j}$ entre chaque paire de sommets $(w_k^{i,j}, w_k^{i,j+1})$, $j = 1, \ldots, t_i - 1$. De plus, ces chemins peuvent être considérés comme étant arête-disjoints. Soit $\widetilde{T}_k^i = \cup_{j=1,\ldots,t_i-1} T_k^{i,j}$. Il est clair que $\widetilde{T}_k^i$ est un chemin entre O'_k et D'_k. Par les mêmes contraintes, $\widetilde{T}_k^i$ respecte l'ordre des sommets du chemin L_k^i, pour $k \in K$, $i = 1,2$. En conséquence, entre chaque paire de sommets (O'_k, D'_k) il existe au moins deux chemins respectant l'ordre des routeurs des chemins L_1^k et L_2^k correspondants. De plus, par les contraintes (7.2), ces chemins sont sommet-disjoints.
D'autre part, il est clair qu'à toute solution du problème MSOND, on peut associer des vecteurs $(\hat{x}, x^{i,k},\ i = 1,2,\ k \in K)$ qui vérifient les contraintes (7.1)-(7.5), ce qui établit l'équivalence du problème de sécurisation présenté et du programme en nombres entiers P. □

Dans ce problème, les chemins de la couche de transport doivent respecter l'ordre des routeurs du chemin de la couche cliente associé, mais il peut y avoir plusieurs fois le même brasseur dans un chemin. Dans de nombreuses applications pratiques, les chemins ne peuvent utiliser qu'une seule fois chaque brasseur. Pour considérer cette nouvelle condition, on note par $G_2^{k,i,j}$, $k \in K$, $i = 1,2$, $j = 1, \ldots, t_i - 1$, le graphe obtenu de G_2 en supprimant les sommets de $T_{i,j}^k = \{w_k^{i,1}, \ldots, w_k^{i,j-1}, w_k^{i,j+2}, \ldots, w_k^{i,t_i}\}$ et les arêtes qui leur sont incidentes.

Si l'on remplace la contrainte (7.1) par la contrainte (7.6) suivante, dans le modèle ci-dessus, nous obtenons une formulation de ce nouveau problème :

$$\begin{array}{ll} \sum_{e \in \delta_{G_2^{k,i,j}}(S)} x^{i,k}(e) \geq 1 & \forall\, k \in K,\ \forall\, S \subset V_2 \backslash T_{i,j}^k, \\ & w_k^{i,j} \in S,\ w_k^{i,j+1} \in V_2 \backslash (S \cup T_{i,j}^k), \\ & j = 1, \ldots, t_i - 1,\ i = 1,2, \end{array} \tag{7.6}$$

En effet, la démonstration est similaire à la précédente.

Une autre question naturelle qui mérite d'être étudiée est d'étendre l'étude du problème MSOND au modèle Peer où les couches cliente et transport forment un seul réseau et où le routage et la topologie sont étudiés simultanément sur la couche cliente et sur la couche transport. Cette question fait l'objet de la section suivante.

7.2 Sécurisation simultanée des deux couches

7.2.1 Présentation du problème PMSND

Pour ce problème, on se place dans le cas d'un modèle Peer qui demande un modèle de service unifié. Les plans de commande et de transfert sont gérés de la même façon sur les deux couches. Ce problème concerne la sécurisation simultanée des deux couches, ce qui permettrait un gain important au niveau du coût d'installation du réseau.

Supposons qu'un ensemble de demandes entre des routeurs IP est défini. A chaque routeur LSR, on associe un brasseur OXC de la couche transport. Chaque liaison, qu'elle soit dans la couche cliente, ou dans la couche transport, est munie d'un coût fixe (coût d'installation). Le *problème de sécurisation multicouche dans un modèle Peer* (PMSND Problem, *Peer Multilayer Survivable Network Design Problem*) consiste à trouver pour chaque demande, deux chemins LSP sommet-disjoints de la couche cliente, ainsi que les deux chemins OCT associés de la couche transport, également sommet-disjoints, respectant l'ordre des routeurs imposé par les LSP, de telle sorte que le coût total soit minimum.

Ce problème peut être modélisé de la manière suivante. On associe à la couche cliente, un graphe $G^1 = (V^1, E^1)$ où les sommets de V^1 correspondent aux routeurs et les arêtes aux liaisons possibles entre les routeurs. On associe de même à la couche transport, un graphe $G^2 = (V^2, E^2)$ où les sommets de V^2 correspondent aux brasseurs optiques et les arêtes aux fibres optiques. On suppose que chaque arête $e \in E^1 \cup E^2$ est munie d'un coût fixe $c(e) > 0$. On peut supposer que les graphes G^1 et G^2 sont complets. Les arêtes inutilisables auront un coût associé infini. On dispose d'un certain nombre de demandes entre des routeurs de V^1. On veut pouvoir écouler ces demandes, même en cas de panne d'une arête ou d'un sommet, de la couche cliente ou de la couche transport. Le problème PMSND consiste donc à déterminer, pour chaque demande, deux chemins entre les routeurs extrémités de la demande, empruntant uniquement des arêtes de E^1, qui soient sommet-disjoints, ainsi que les chemins leur correspondant dans la couche transport, utilisant des arêtes de E^2, respectant l'ordre des routeurs imposé par les deux chemins de la couche cliente, qui soient sommet-disjoints et tel que

le coût total soit minimum. Un chemin de la couche transport peut utiliser plusieurs fois un même brasseur.

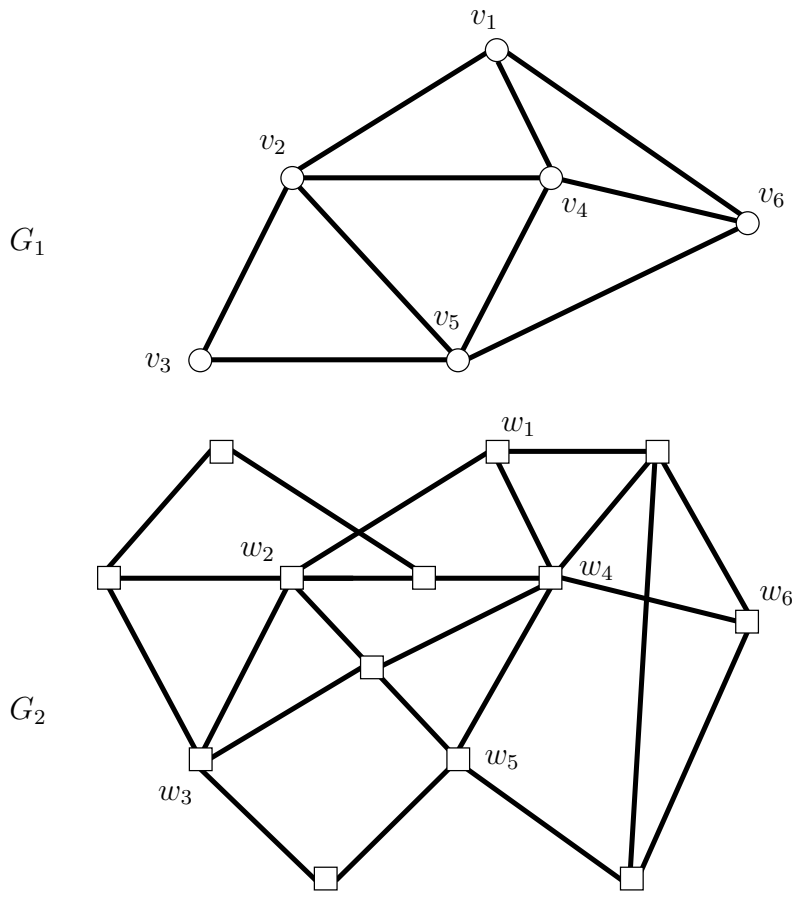


FIG. 7.5 – Exemple du problème

La figure 7.5 présente un exemple de problème de sécurisation simultanée des deux couches. Les sommets $v_i \in V^1$ représentent les routeurs de la couche cliente. De même, les sommets $w_i \in V^2$ représentent les brasseurs de la couche transport. Le brasseur $w_i \in V^2$ est associé au routeur $v_i \in V^1$. L'ensemble E^1 représente l'ensemble des liaisons possibles dans la couche cliente. De même, l'ensemble E^2 représente l'ensemble des liaisons possibles sur la couche transport. On dispose de demandes entre les routeurs de la couche cliente et l'on cherche à sécuriser simultanément les deux couches de telle sorte que la demande puisse toujours être écoulée, même en cas de panne d'un sommet ou d'une arête d'une des deux couches.

On suppose par exemple qu'il existe une demande entre les routeurs v_2 et v_5. La figure 7.6 présente une solution réalisable du problème. En effet, dans la couche cliente,

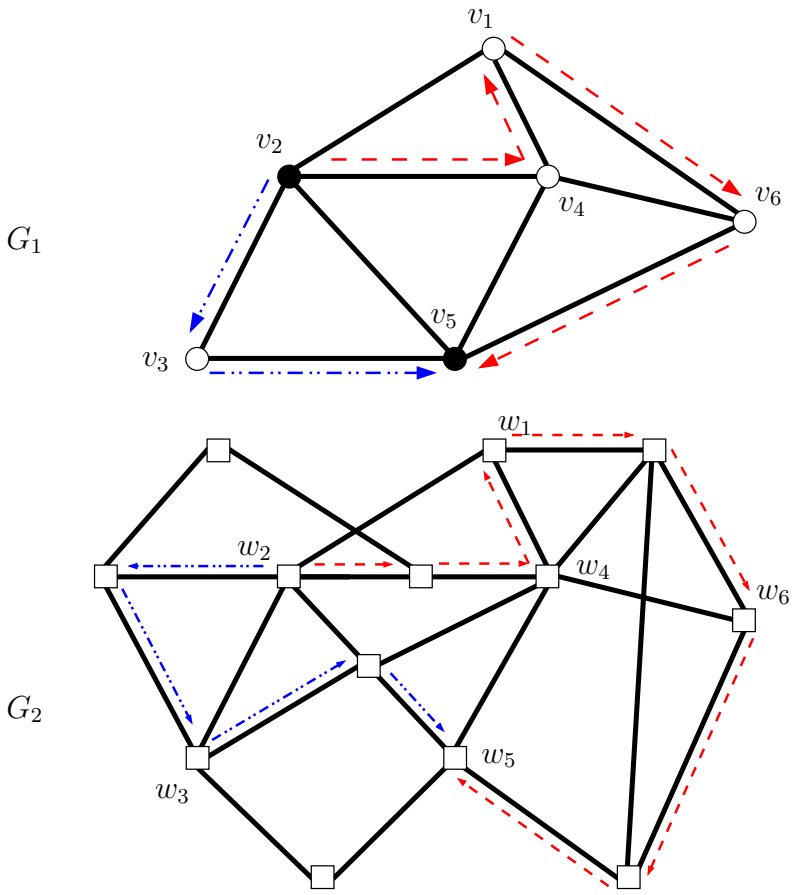


FIG. 7.6 – Solution réalisable

nous avons bien deux chemins allant de v_2 à v_5 qui sont sommet-disjoints. Le premier chemin, $(v_2, v_4, v_1, v_6, v_5)$ a pour chemin correspondant dans la couche transport, le chemin passant entre autres par w_2, w_4, w_1, w_6 et w_5. Ce chemin respecte bien l'ordre des routeurs imposé par le chemin correspondant de la couche cliente. De même, le deuxième chemin de la couche cliente (v_2, v_3, v_5) a pour chemin correspondant dans la couche transport, le chemin passant par w_2, w_3 et w_5 qui respecte également l'ordre des routeurs. Comme les chemins de la couche cliente, les deux chemins de la couche transport sont bien sommet-disjoints.

7.2.2 Formulation du problème

Soit K l'ensemble des demandes dans G^1. Pour $k \in K$, soit (O_k, D_k) la paire origine-destination de k. On note par O'_k et D'_k les brasseurs associés à O_k et D_k. On note par

w_i le sommet de G^2 correspondant au sommet v_i de G^1. Les graphes G^1 et G^2 sont supposés complets. On suppose que chaque arête e de G^1 et G^2 est munie d'un poids $c(e) > 0$. Le problème PMSND consiste à déterminer pour chaque demande $k \in K$, deux chemins sommet-disjoints L_k^1 et L_k^2 dans G^1 entre O_k et D_k et deux chemins sommet-disjoints T_k^1 et T_k^2 dans G^2 entre O'_k et D'_k tels que les sommets de T_k^i suivent le même ordre que les sommets de L_k^i pour $i = 1, 2$ et le coût total est minimum.

Soit $\hat{x} \in I\!R^{E^1}$ tel que $\hat{x}(e) = 1$ si l'arête e de $|E^1|$ appartient à la solution et 0 sinon. Si $k \in K$ et $i \in \{1, 2\}$, soit $x^{i,k} \in I\!R^{E^1}$ un vecteur associé à i et à k tel que $x^{i,k}(e) = 1$ si e appartient au $i^{\text{ème}}$ chemin dans G^1 entre O_k et D_k et 0 sinon. De même, soit $\hat{y} \in I\!R^{E^2}$ tel que $\hat{y}(f) = 1$ si l'arête f de E^2 appartient à la solution et 0 sinon. Soit $y^{i,k} \in I\!R^{E^2}$ un vecteur associé à i et à k tel que $y^{i,k}(f) = 1$ si f appartient au $i^{\text{ème}}$ chemin dans G^2 entre O'_k et D'_k et 0 sinon.

Comme il est montré ci-dessous, le problème est équivalent au programme en nombres entiers suivant :

$$\text{Minimiser} \sum_{e \in E^1} c(e)\hat{x}(e) + \sum_{f \in E^2} c(f)\hat{y}(f)$$

$$(Q) \quad \begin{aligned}
& x^{i,k}(\delta_{G^1}(W)) \geq 1 && \forall\, W \subset V^1,\ O_k \in W,\ D_k \in \overline{W}, \\
& && \forall k \in K, \quad i = 1, 2, && (7.7)\\
& y^{i,k}(\delta_{G^2}(W)) - x^{i,k}(v_p v_q) \geq 0 && \forall\, v_p v_q \in E^1, \\
& && \forall\, W \subset V^2,\ w_p \in W,\ w_q \in \overline{W}, && (7.8)\\
& && \forall\, k \in K, \quad i = 1, 2, \\
& \sum_{e \in \delta_{G^1}(u)} (x^{1,k}(e) + x^{2,k}(e)) \leq 2 && \forall u \in V^1 \backslash \{O_k, D_k\}, \quad \forall k \in K, && (7.9)\\
& \sum_{f \in \delta_{G^2}(u)} (y^{1,k}(f) + y^{2,k}(f)) \leq 2 && \forall u \in V^2 \backslash \{O'_k, D'_k\}, \quad \forall k \in K, && (7.10)\\
& \sum_{k \in K} \sum_{i=1,2} x^{i,k}(e) \leq M\hat{x}(e) && \forall\, e \in E^1, && (7.11)\\
& \sum_{k \in K} \sum_{i=1,2} y^{i,k}(f) \leq M\hat{y}(f) && \forall\, f \in E^2, && (7.12)\\
& \hat{x}(e) \in \{0, 1\} && \forall\, e \in E^1, && (7.13)\\
& \hat{y}(f) \in \{0, 1\} && \forall\, f \in E^2, && (7.14)\\
& x^{i,k}(e) \in \{0, 1\} && \forall\, e \in E^1, \quad \forall k \in K, \quad i = 1, 2, && (7.15)\\
& y^{i,k}(f) \in \{0, 1\} && \forall\, f \in E^2, \quad \forall k \in K, \quad i = 1, 2. && (7.16)
\end{aligned}$$

Les contraintes (7.7) expriment le fait que pour chaque demande $k \in K$, il existe

au moins deux chemins entre O_k et D_k dans G^1. Les contraintes (7.8) expriment le fait qu'il existe également deux chemins entre O'_k et D'_k dans G^2, associés aux chemins de G^1 qui respectent l'ordre des routeurs. Les contraintes (7.9) et (7.10) imposent à ces chemins d'être sommet-disjoints. On suppose que M est choisi suffisamment grand. Les contraintes (7.11) et (7.12) permettent de relier les variables $\hat{x}$ et x ainsi que les variables $\hat{y}$ et y. Les contraintes (7.13)-(7.16) sont les contraintes d'intégrité.

Théorème 7.2 *Le problème PMSND est équivalent au problème* (Q).

Preuve. Soient $(\hat{x}, x^{i,k}, k \in K, i = 1,2)$ et $(\hat{y}, y^{i,k}, k \in K, i = 1,2)$ une solution du programme (Q) ci-dessus. Soient $L^i_k = \{e \in E_1 \mid x^{i,k}(e) = 1\}$ et $T^i_k = \{e \in E_2 \mid y^{i,k}(e) = 1\}$. Par les contraintes (7.7), L^i_k contient un chemin $\widetilde{L}^i_k$ entre O_k et D_k. Soient $V(\widetilde{L}^i_k) = (O_k = v^{i,1}_k, v^{i,2}_k, \ldots, v^{i,t_i}_k = D_k)$ pour $i = 1,2$. Par les contraintes (7.8), il s'ensuit que T^i_k contient un chemin $T^{i,j}_k$ entre $w^{i,j}_k$ et $w^{i,j+1}_k$ pour $j = 1, \ldots, t_i - 1$. Soient $\widetilde{T}^i_k = \bigcup_{j=1,\ldots,t_i-1} T^{i,j}_k$ pour $i = 1,2$. Il est clair que $\widetilde{T}^i_k$ est un chemin entre O'_k et D'_k respectant l'ordre des sommets de $\widetilde{L}^i_k$. De plus, les contraintes (7.9)-(7.10) impliquent que les chemins $\widetilde{T}^1_k$ et $\widetilde{T}^2_k$ ($\widetilde{L}^1_k$ et $\widetilde{L}^2_k$) sont sommet-disjoints. En conséquence $\bigcup_{k\in K,\, i=1,2} L^i_k \cup T^i_k$ est une solution du problème PMSND.
D'autre part, il est clair qu'à toute solution du problème, on peut associer des vecteurs $(\hat{x}, x^{i,k}, \hat{y}, y^{i,k}, \quad i = 1,2, \quad k \in K)$ qui vérifient les contraintes (7.7)-(7.16), ce qui établit l'équivalence du problème PMSND et du programme en nombres entiers (Q). □

7.2.3 Plusieurs points d'entrée et de sortie dans la couche transport

Dans le problème précédent, nous avons considéré qu'un routeur était associé à un seul brasseur. Supposons maintenant qu'à chaque routeur peuvent être associés plusieurs brasseurs de la couche transport. Cela permet à un routeur d'avoir plusieurs entrées et plusieurs sorties possibles dans la couche transport. Chaque liaison, qu'elle soit dans la couche cliente, ou dans la couche transport, ou qu'elle représente une interface UNI entre un routeur et un brasseur, est munie d'un coût fixe (coût d'installation). Ce problème de sécurisation consiste toujours à trouver pour chaque demande deux chemins de la couche cliente sommet-disjoints, ainsi que leurs deux représentants dans la couche transport, également sommet-disjoints, respectant l'ordre des routeurs imposé par les chemins de la couche cliente, de telle manière que le coût total soit

minimum. Nous considérons ici qu'un brasseur peut apparaître plusieurs fois dans un même chemin.

Ce problème peut être modélisé de la manière suivante. Les deux couches sont représentées par un seul graphe $G = (V^1 \cup V^2, E^1 \cup E^2 \cup E^3)$ où V^1 correspond aux routeurs de la couche cliente et V^2 aux brasseurs de la couche transport. Le graphe G possède trois types d'arêtes. Les arêtes de E^1 sont entre deux sommets de V^1 et relient donc deux routeurs de la couche cliente. Les arêtes de E^2 prennent leurs extrémités dans V^2 et représentent les liaisons optiques de la couche transport. E^3 contient les arêtes représentant les interfaces UNI entre les routeurs et les brasseurs. On suppose que chaque arête $e \in E^1 \cup E^2 \cup E^3$ est munie d'un coût fixe $c(e) > 0$. On peut supposer que le graphe G est complet. Les arêtes inutilisables auront un coût associé infini. On dispose d'un certain nombre de demandes entre des routeurs de V^1. On veut pouvoir écouler ces demandes, même en cas de panne d'une arête ou d'un sommet. Le problème consiste donc à déterminer, pour chaque demande, deux chemins entre les routeurs extrémités de la demande, empruntant uniquement des arêtes de E^1, qui soient sommet-disjoints, ainsi que les chemins leur correspondant dans la couche transport, utilisant des arêtes de $E^2 \cup E^3$, passant par les sommets des chemins de la couche cliente dans l'ordre, qui soient sommet-disjoints et tel que le coût total soit minimum.

La figure 7.7 présente un exemple de ce problème de sécurisation de la couche cliente. Les sommets $v_i \in V^1$ représentent les routeurs de la couche cliente. De même, les sommets $w_i \in V^2$ représentent les brasseurs de la couche transport. L'ensemble E^1 est constitué des arêtes (v_i, v_j) où $v_i, v_j \in V^1$ et représente l'ensemble des liaisons possibles dans la couche cliente. De même, l'ensemble E^2 est constitué des arêtes (w_i, w_j) où $w_i, w_j \in V^2$ et représente l'ensemble des liaisons possibles sur la couche transport. Les arêtes (v_i, w_j), où $v_i \in V^1$ et $w_j \in V^2$, ayant une extrémité dans V^1 et l'autre dans V^2, représentent les interfaces UNI entre les routeurs et les brasseurs. On remarque que chaque routeur a au moins un brasseur associé et qu'un routeur peut avoir des interfaces avec des brasseurs différents. En effet, le routeur $v_2 \in V^1$ a une interface avec le brasseur $w_2 \in V^2$, mais aussi avec le brasseur $w_4 \in V^2$. Cela permet au routeur v_2 d'avoir deux points d'entrée différents dans la couche transport. On dispose de demandes entre les routeurs de la couche cliente et l'on cherche à sécuriser simultanément les deux couches de telle sorte que la demande puisse toujours être écoulée, même en cas de panne d'un sommet ou d'une arête d'une des deux couches.

On suppose par exemple qu'il existe une demande entre les routeurs v_2 et v_5. La figure 7.8 présente une solution réalisable du problème. En effet, dans la couche cliente, nous avons bien deux chemins allant de v_2 à v_5 qui sont sommet-disjoints. Le premier chemin, $(v_2, v_4, v_1, v_6, v_5)$ a pour chemin correspondant dans la couche transport, le

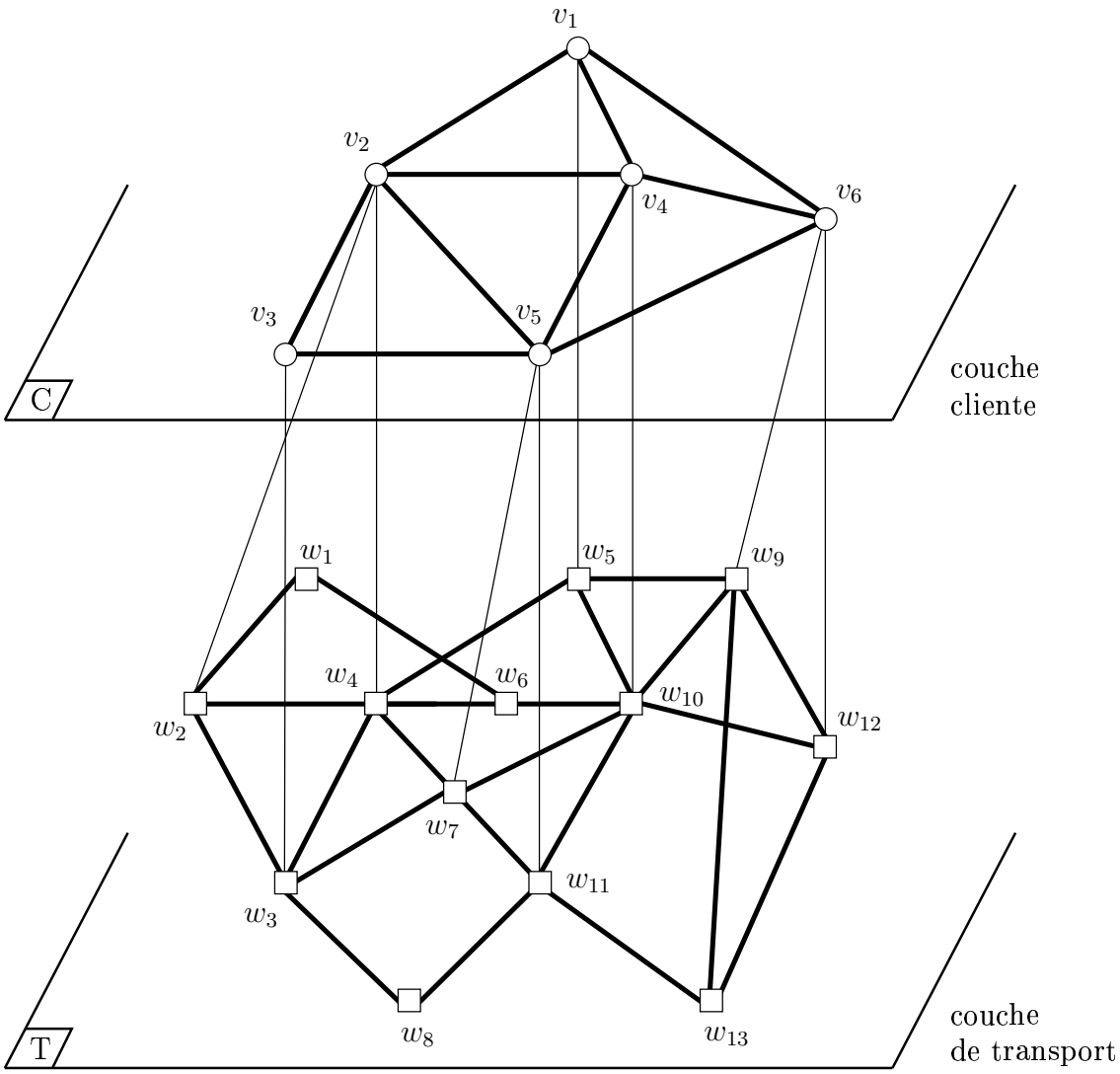


FIG. 7.7 – Exemple du problème

chemin $(v_2, w_4, w_6, w_{10}, v_4, w_{10}, w_5, v_1, w_5, w_9, v_6, w_{12}, w_{13}, w_{11}, v_5)$. Ce chemin emprunte bien, dans l'ordre, les routeurs v_2, v_4, v_1, v_6, v_5 constituant le chemin correspondant de la couche cliente. De même, le deuxième chemin de la couche cliente, (v_2, v_3, v_5) a pour chemin correspondant dans la couche transport, le chemin $(v_2, w_2, w_3, v_3, w_3, w_7, v_5)$ qui passe bien par v_2, puis v_3 pour aboutir à v_5. Comme les chemins de la couche cliente, les deux chemins de la couche transport sont bien sommet-disjoints. On remarque que les deux chemins utilisent un point d'entrée différent dans la couche transport à partir de v_2 et deux points de sortie différents pour rejoindre v_5. En effet, w_2 et w_4 sont les deux points d'entrée. De même w_7 et w_{11} sont les deux points de sortie de la couche

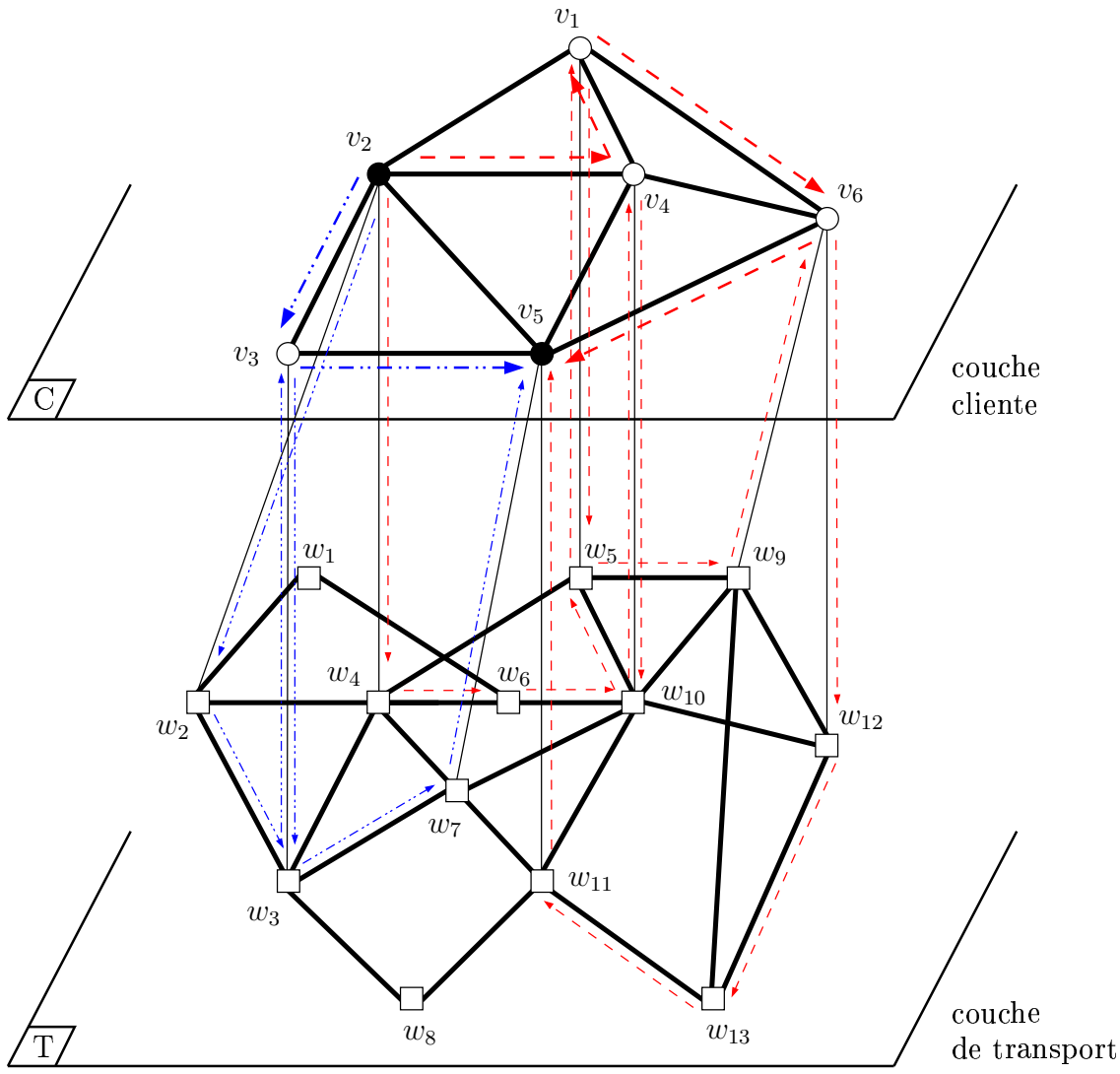


FIG. 7.8 – Solution réalisable

transport. On remarque également que l'accés au routeur v_6 se fait à l'aide de deux interfaces UNI (avec le brasseur w_9 et le brasseur w_{12}). Néanmoins, l'accès à la couche cliente au niveau des routeurs v_1, v_3 et v_4 se fait grâce à la même interface utilisée pour la sortie comme pour l'entrée. Il serait intéressant de pourvoir imposer automatiquement des interfaces d'entrée et de sortie différentes.

Soit K l'ensemble des demandes dans G^1. Pour $k \in K$, soit (O_k, D_k) la paire origine-destination de k. On suppose que chaque arête $e \in E^1 \cup E^2 \cup E^3$ est munie d'un poids $c(e) > 0$. Soit $\hat{x} \in \mathbb{R}^{E^1}$ tel que $\hat{x}(e) = 1$ si l'arête e de E^1 appartient à la solution et 0 sinon. Si $k \in K$ et $i \in \{1, 2\}$, soit $x^{i,k} \in \mathbb{R}^{E^1}$ un vecteur associé à i et à k tel que

$x^{i,k}(e) = 1$ si e appartient au $i^{\text{ème}}$ chemin utilisant des arêtes de E^1 entre O_k et D_k. De même, soit $\hat{y} \in I\!R^{E^2}$ tel que $\hat{y}(f) = 1$ si l'arête e de E^2 appartient à la solution et 0 sinon. Soit $y^{i,k} \in I\!R^{E^2}$ un vecteur associé à i et à k tel que $y^{i,k}(f) = 1$ si f appartient au $i^{\text{ème}}$ chemin utilisant des arêtes de $E^2 \cup E^3$ entre O_k et D_k. On définit également $\hat{z} \in I\!R^{E^3}$ tel que $\hat{z}(f) = 1$ si l'arête f de E^3 appartient à la solution et 0 sinon. Soit $z^{i,k} \in I\!R^{E^3}$ un vecteur associé à i et à k tel que $z^{i,k}(f) = 1$ si f appartient au $i^{\text{ème}}$ chemin utilisant des arêtes de $E^2 \cup E^3$ entre O_k et D_k.
Soit $v_p, v_q \in V^1$, on pose $I_{p,q} = \delta_{G(E^3)}(v_p) \cup \delta_{G(E^3)}(v_q)$, l'ensemble des arêtes de E^3 incidentes à v_p ou à v_q.

Comme il est montré ci-dessous, le problème est équivalent au programme en nombres entiers suivant :

$$\text{Minimiser} \sum_{e \in E^1} c(e)\hat{x}(e) + \sum_{f \in E^2} c(f)\hat{y}(f) + \sum_{f \in E^3} c(f)\hat{z}(f)$$

$$x^{i,k}(\delta_{G(E^1)}(W)) \geq 1 \qquad \forall\, W \subset V^1,\ O_k \in W,\ D_k \in \overline{W},\ \forall\, k \in K,\quad i = 1,2, \tag{7.17}$$

$$y^{i,k}(\delta_{G(E^2)}(W)) + z^{i,k}(\delta_{G(I_{p,q})}(W)) - x^{i,k}(v_p, v_q) \geq 0 \qquad \forall\, (v_p, v_q) \in E^1,\ \forall\, W \subset V^2 \cup v_p \cup v_q,\ v_p \in W,\ v_q \in \overline{W},\ \forall\, k \in K,\quad i = 1,2, \tag{7.18}$$

$$\sum_{e \in \delta_{G(E^1)}(u)} (x^{1,k}(e) + x^{2,k}(e)) \leq 2 \qquad \forall u \in V^1 \backslash \{O_k, D_k\},\quad \forall k \in K, \tag{7.19}$$

$$\sum_{f \in \delta_{G(E^2)}(u)} (y^{1,k}(f) + y^{2,k}(f)) \leq 2 \qquad \forall u \in V^2 \backslash \{O'_k, D'_k\},\quad \forall k \in K, \tag{7.20}$$

$$\sum_{k \in K} \sum_{i=1,2} x^{i,k}(e) \leq M\hat{x}(e) \qquad \forall\, e \in E^1, \tag{7.21}$$

$$\sum_{k \in K} \sum_{i=1,2} y^{i,k}(f) \leq M\hat{y}(f) \qquad \forall\, f \in E^2, \tag{7.22}$$

$$\sum_{k \in K} \sum_{i=1,2} z^{i,k}(e) \leq M\hat{z}(f) \qquad \forall\, f \in E^3, \tag{7.23}$$

$$\hat{x}(e) \in \{0,1\} \qquad \forall\, e \in E^1, \tag{7.24}$$

$$\hat{y}(f) \in \{0,1\} \qquad \forall\, f \in E^2, \tag{7.25}$$

$$\hat{z}(f) \in \{0,1\} \qquad \forall\, f \in E^3, \tag{7.26}$$

$$x^{i,k}(e) \in \{0,1\} \qquad \forall\, e \in E^1,\quad \forall k \in K,\quad i = 1,2, \tag{7.27}$$

$$y^{i,k}(f) \in \{0,1\} \qquad \forall\, f \in E^2,\quad \forall k \in K,\quad i = 1,2, \tag{7.28}$$

$$z^{i,k}(f) \in \{0,1\} \qquad \forall\, f \in E^3,\quad \forall k \in K,\quad i = 1,2. \tag{7.29}$$

Les contraintes (7.17) expriment le fait que pour chaque demande $k \in K$, il existe au moins deux chemins utilisant des arêtes de E^1 entre O_k et D_k. Les contraintes (7.18) expriment le fait qu'il existe également deux chemins utilisant des arêtes de $E^2 \cup E^3$, associés aux chemins précédents, qui passent par les routeurs des premiers chemins dans l'ordre. Un même brasseur peut néanmoins apparaitre plusieurs fois dans le même chemin. Les contraintes (7.19) et (7.20) imposent à ces chemins d'être sommet-disjoints. On suppose que M est choisi suffisamment grand. Les contraintes (7.21)-(7.23) permettent de relier les variables $\hat{x}$ et x, $\hat{y}$ et y et $\hat{z}$ et z. Les contraintes (7.24)-(7.29) sont les contraintes d'intégrité.

Soit $(\hat{x},\, x^{i,k},\, \hat{y},\, y^{i,k},\, \hat{z},\, z^{i,k}, \quad i = 1,2, \quad k \in K)$ une solution du programme ci-dessus. Soit $k \in K$ et (O_k, D_k) la paire origine-destination de k. Les contraintes (7.17) impliquent qu'il existe au moins deux chemins utilisant des arêtes de E^1 entre O_k et D_k. Soit $e = v_p v_q \in E^1$ tel que $x^{i,k}(e) = 1$. Les contraintes (7.18) impliquent que $y^{i,k}(\delta_{G(E^2)}(W)) \geq 1$ ou bien $z^{i,k}(\delta_{G(I_{p,q})}(W)) \geq 1, \quad \forall\, W \subset V^2 \cup v_p \cup v_q,\, w_p \in W,\, w_q \in \overline{W}$. Il existe donc au moins un chemin utilisant soit des arêtes de E^2, soit des arêtes de $I_{p,q}$ ($I_{p,q}$ étant l'ensemble des arêtes de E^3 incidentes à v_p ou v_q) entre w_p et w_q (associés à v_p et v_q). Donc il existe au moins deux chemins entre O'_k et D'_k utilisant des arêtes de E^2 et E^3, associés aux chemins utilisant des arêtes de E^1 et ces chemins respectent l'ordre des routeurs. Les contraintes (7.28) et (7.29) impliquent que ces chemins sont sommet-disjoints. Puisque les coûts des arêtes sont strictements positifs, les chemins ainsi trouvés sont uniques.

D'autre part, il est clair qu'à toute solution du problème, on peut associer des vecteurs $(\hat{x},\, x^{i,k},\, \hat{y},\, y^{i,k},\, \hat{z},\, z^{i,k} \quad i = 1,2, \quad k \in K)$ qui vérifient les contraintes (7.17)-(7.29), ce qui établit l'équivalence du problème de sécurisation simultanée des deux couches présenté dans ce chapitre et du programme en nombres entiers ci-dessus.

Dans la section suivante, nous présentons un algorithme de résolution approchée pour chacun des problèmes MSOND et PMSND présentés au début de ce chapitre.

7.3 Algorithmes de résolution et résultats expérimentaux pour les problèmes MSOND et PMSND

Nous avons développé une métaheuristique de type recherche locale pour ces deux problèmes dans le cas où les brasseurs ne peuvent être utilisés qu'une seule fois dans un chemin. Nous avons utilisé la méthode du recuit simulé. Dans la suite, nous présentons cette méthode, puis nous décrivons le système de voisinage que nous avons utilisé.

7.3.1 Algorithme de recuit simulé

Soit S l'ensemble des points pouvant être visités durant la recherche locale. La structure de voisinage donne les règles de déplacement dans l'espace de recherche. On note f la fonction objectif que l'on désire minimiser. La méthode du recuit simulé est une méthode de recherche locale dans laquelle on accepte, sous certaines conditions, de se déplacer d'une solution $s \in S$ vers une solution $s' \in N(s)$ où $N(s)$ représente le voisinage de s telle que $f(s') \geq f(s)$. Cela revient à accepter de dégrader la solution courante pour permettre de sortir d'un minimum local. L'algorithme 7.1 illustre les principales étapes de la méthode du recuit simulé.

Algorithme 7.1 Algorithme de recuit simulé

1 : Choisir une solution $s \in S$ ainsi qu'une température initiale T.
2 : Tant qu'aucun critère d'arrêt n'est satisfait faire
3 : Choisir aléatoirement $s' \in N(s)$,
4 : Générer un nombre réel aléatoire r dans $[0, 1]$,
5 : Si $r < p(T, s, s')$ alors poser $s := s'$,
6 : Mettre à jour T,
7 : Fin tant que.

Nous avons choisi d'utiliser la fonction de Boltzmann comme fonction p, c'est-à-dire

$$p(T, s, s') = e^{\frac{f(s)-f(s')}{T}}.$$

De cette façon, si $f(s') < f(s)$ (*i.e.* s' est meilleure que s), alors $p(T, s, s') > 1$ et donc $p(T, s, s') > r$ avec r un nombre réel choisi aléatoirement dans $[0, 1]$. La solution s' est toujours acceptée. Si maintenant $f(s') > f(s)$, si T est grand, s' est presque sûr d'être acceptée, mais si T devient petit, s' va probablement être refusée.

Nous avons choisi une température initiale suffisamment élevée pour permettre une bonne exploration de l'espace de recherche. Puis petit à petit, la température décroît jusqu'à tendre vers 0, ce qui signifie que la méthode n'acceptera plus de détériorer la solution courante. Pour choisir la température initiale, nous générons aléatoirement un ensemble de paires (s, s') où $s' \in N(s)$ et nous déterminons la température initiale de telle sorte que toutes les solutions s' aient de très grandes chances d'être acceptées. La mise à jour de l'étape 6 de l'algorithme 7.1 consiste à multiplier T par un réel α. Afin de faire décroître la température T, nous avons considéré α avec pour valeur 0.95, 0.97 ou 0.99.

Nous allons maintenant présenter notre choix de solution initiale. Le problème MSOND a été modélisé en terme de graphe en utilisant deux graphes G^1 et G^2. Le premier est

fixé et nous connaissons pour chaque demande $k \in K$ deux chemins de G^1 notés L_k^1 et L_k^2 qui sont sommet-disjoints. Le graphe G^2 peut être considéré complet et on sait que tout sommet de V^1 a un correspondant dans V^2. On note w_j le sommet de V^2 associé à v_j de V^1. Une manière de choisir les T_k^1 et T_k^2 associés à L_k^1 et L_k^2 respectivement consiste, pour chaque arête $e = v_j v_l \in L_k^i$, $i = 1, 2$, à considérer l'arête $w_j w_l$ de E^2 dans le chemin T_k^i. Les chemins T_k^1 et T_k^2 sont bien sommets-disjoints comme les chemins L_k^1 et L_k^2 le sont et respectent l'ordre des routeurs dans G_1.

Nous allons à présent décrire l'opérateur de voisinage que nous avons utilisé pour le problème MSOND. Soient une demande k, $i \in \{1, 2\}$ et une arête $v_j v_l \in L_k^i$, la solution s' qui diffère d'une solution s uniquement par la portion du chemin T_k^i entre w_j et w_l fait partie du voisinage de s. Pour définir cette portion de chemin manquante, on construit le graphe $\bar{G} = (\bar{V}, \bar{E})$ tel que tout sommet de V^2 n'appartenant ni à L_k^1 ni à L_k^2 appartient à $\bar{V}$, $w_j w_l \in \bar{V}$ et toute arête reliant deux sommets de $\bar{V}$ qui n'appartient ni à T_k^1, ni à T_k^2 appartient à $\bar{E}$. La valuation sur les arêtes de $\bar{G}$ est identique à celle sur G^2, excepté pour les arêtes déjà considérées dans la solution qui ont un coût nul. On exécute alors un algorithme de plus courts chemins sur le graphe $\bar{G}$ entre w_j et w_l qui nous permet de compléter la portion de chemin manquante.

Ce système de voisinage peut être facilement étendu au problème PMSND en considérant successivement la modification des chemins de G^1 et celle des chemins de G^2. Lorsqu'une nouvelle solution est basée sur le reroutage d'une demande dans le graphe G^1, le routage associé dans le graphe G^2 est construit de la même façon qu'une solution initiale pour le problème MSOND. Ce qui permet toujours d'avoir une solution réalisable pour le problème.

Nous avons implémentés ces deux algorithmes basés sur la méthode du recuit simulé. Dans la suite, nous présentons quelques résultats expérimentaux pour le problème MSOND ainsi que pour le problème PMSND.

7.3.2 Résultats expérimentaux

Avant de présenter les différents résultats expérimentaux que nous avons obtenus, nous donnons un bref descriptif du contexte informatique. Les algorithmes ont été développés en langage C++ et testés sur un Pentium IV cadencé à 2,4 Ghz avec 1024 Mo de mémoire vive, sous système linux. Nous avons fixé un temps maximum de 30 minutes. Les résultats présentés ont été obtenus à partir d'instances basées sur les instances de la TSP Library [76]. Le graphe G^1 est obtenu en considérant un sous-ensemble de sommets d'instances de la TSP Library et le graphe G^2 n'est autre que

l'instance complète considérée. Les graphes de la TSP Library que nous avons testés ont entre 52 et 439 sommets et des coûts basés sur une distance euclidienne. Le nombre de sommets est donné dans le nom de l'instance. Par exemple, l'instance berlin52 possède 52 sommets. Pour le problème PMSND, les coûts pour le graphe G^1 sont des coûts unitaires. Les demandes sont choisies aléatoirement et l'on considère des ensembles de 20 à 500 demandes.

Nos résultats expérimentaux sont reportés dans les tables des sections suivantes. Les différentes colonnes de ces tables représentent :

Inst	:	le nom de l'instance de la TSP Library,
$\|V^1\|$	:	le nombre de sommets de G^1,
$\|V^2\|$	:	le nombre de sommets de G^2,
$\|K\|$	:	le nombre de demandes,
SI	:	la valeur de solution initiale,
5 min	:	la valeur de la meilleure solution trouvée au bout de 5 min,
10 min	:	la valeur de la meilleure solution trouvée au bout de 10 min,
20 min	:	la valeur de la meilleure solution trouvée au bout de 20 min,
30 min	:	la valeur de la meilleure solution trouvée au bout de 30 min.

7.3.2.1 Résultats expérimentaux pour le problème MSOND

Nous avons tout d'abord esssayé de faire varier le coefficient α gérant la diminution de la température. Les figures 7.9 et 7.10 présentent l'évolution de la valeur de la meilleure solution en fonction du nombre d'itérations pour les instances berlin52 et eil101 et 20 demandes, avec α égal à 0.95, 0.97 et 0.99.

Pour l'instance berlin52, la meilleure solution a été obtenue pour $\alpha = 0.99$ et cette solution a été trouvée très rapidement, après seulement 10000 itérations. Dans le cas de l'instance eil101, c'est $\alpha = 0.97$ qui a donné la meilleure solution. On remarque que α à 0.95 a permis de trouver une solution intermédiaire pour les deux instances. Dans la suite des résultats nous avons utilisé $\alpha = 0.95$.

La table 7.1 regroupe les résultats obtenus pour le problème MSOND avec $\alpha = 0.95$.

Lorsque l'algorithme de recuit simulé n'a pas accepté de nouvelle solution depuis 40000 itérations, on considère qu'il est stable et le programme s'arrête. Pour les instances pour lesquelles l'algorithme s'est stabilisé avant 30 minutes d'exécution, nous avons utilisé des tirets montrant que le programme a bien été stoppé. C'est par exemple

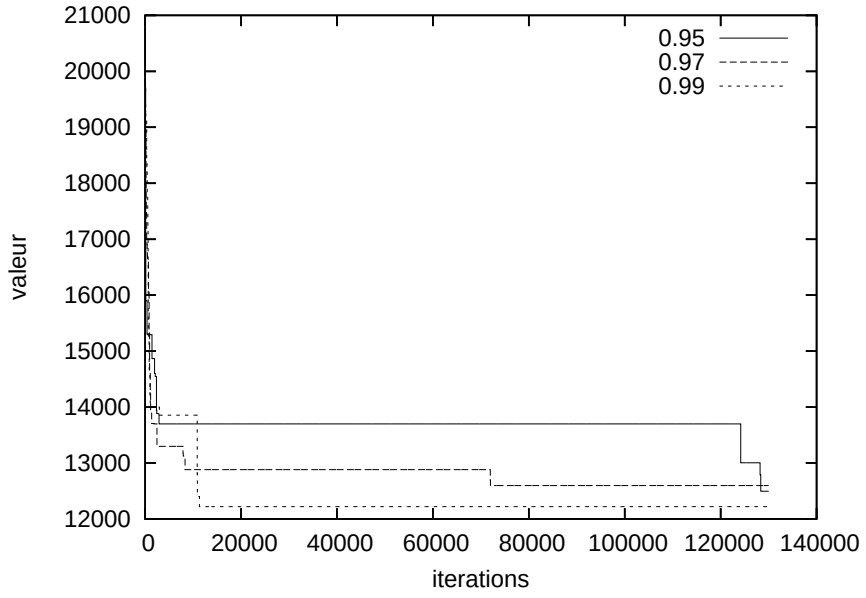


FIG. 7.9 – Instance berlin52 et 20 demandes pour α=0.95, 0.97 et 0.99

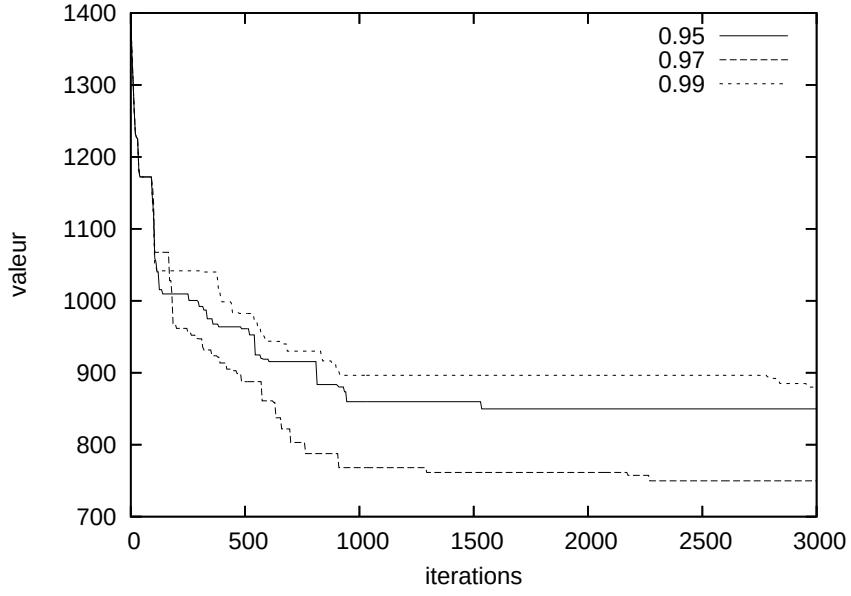


FIG. 7.10 – Instance eil101 et 20 demandes pour α=0.95, 0.97 et 0.99

le cas de la plupart des instances basées sur berlin52 et eil101 pour lesquelles l'algorithme s'est stabilisé assez rapidement, certaines fois même au bout de 5 minutes. La meilleure solution trouvée apparaît en gras dans la table.

Inst	$\|V^1\|$	$\|V^2\|$	$\|K\|$	SI	5 min	10 min	20 min	30 min
berlin52	40	52	20	24395,50	13698,50	**12494,60**	12494,60	12494,60
berlin52	40	52	40	42302,30	**24218,30**	-	-	-
berlin52	40	52	60	61732,80	**35946,40**	-	-	-
berlin52	40	52	80	83627,40	**45061,30**	-	-	-
berlin52	40	52	100	95878,70	**49935,40**	-	-	-
berlin52	40	52	200	173320,00	**92071,70**	92071,70	-	-
berlin52	40	52	500	340803,00	**164738,00**	164738,00	164738,00	-
eil101	70	101	20	1396,61	**849,67**	849,67	849,67	849,67
eil101	70	101	40	2738,84	**1617,20**	-	-	-
eil101	70	101	60	3744,97	**2040,19**	-	-	-
eil101	70	101	80	4952,48	**2788,12**	2788,12	-	-
eil101	70	101	100	5938,43	**3411,36**	-	-	-
eil101	70	101	200	11479,10	**5099,58**	5099,58	-	-
eil101	70	101	500	26186,60	**10251,30**	10251,30	-	-
ch150	120	150	20	14118,90	**8078,27**	8078,27	8078,27	8078,27
ch150	120	150	40	27326,70	**17394,50**	17394,50	17394,50	17394,50
ch150	120	150	60	41583,70	24491,00	24057,60	23678,80	**23507,10**
ch150	120	150	80	55031,20	31414,90	30538,40	29818,30	**29215,40**
ch150	120	150	100	68728,10	**42729,40**	42729,40	42729,40	42729,40
ch150	120	150	200	140838,00	75734,00	**75707,40**	75707,40	75707,40
ch150	120	150	500	325365,00	156518,00	151013,00	**150336,00**	150336,00
d198	160	198	20	1396,61	**849,67**	849,67	849,67	849,67
d198	160	198	40	88572,80	49210,60	46519,60	**46382,50**	46382,50
d198	160	198	60	132824,00	**93001,60**	93001,60	93001,60	93001,60
d198	160	198	80	174167,00	**112139,00**	112139,00	112139,00	112139,00
d198	160	198	100	212442,00	**129329,00**	129329,00	129329,00	129329,00
d198	160	198	200	403182,00	202573,00	185035,00	172056,00	**163049,00**
d198	160	198	500	989683,00	487598,00	446802,00	**431397,00**	431397,00
gil262	220	262	20	3954,01	2730,79	2730,79	**2651,41**	2651,41
gil262	220	262	40	7861,13	5017,88	**4923,96**	4923,96	4923,96
gil262	220	262	60	12177,10	7486,19	**7241,65**	7241,65	7241,65
gil262	220	262	80	16313,90	10057,50	9700,63	9273,90	**9143,87**
gil262	220	262	100	20720,00	**12805,80**	12805,80	12805,80	12805,80
gil262	220	262	200	40359,70	22557,60	**22529,00**	22529,00	22529,00
gil262	220	262	500	98340,10	53850,60	**46985,50**	46985,50	46985,50
pr439	380	439	20	166066,00	89507,60	86661,00	**86643,50**	86643,50
pr439	380	439	40	361386,00	231267,00	229517,00	228981,00	**228905,00**
pr439	380	439	60	519006,00	319711,00	**319450,00**	319450,00	319450,00
pr439	380	439	80	685435,00	428727,00	424273,00	**418955,00**	418955,00
pr439	380	439	100	833778,00	544206,00	530276,00	**528608,00**	528608,00
pr439	380	439	200	1690780,00	1149220,00	1057910,00	1037750,00	**1031720,00**
pr439	380	439	500	4337390,00	3302890,00	2812940,00	2480290,00	**2399110,00**

TAB. 7.1 – Résultats expérimentaux pour le problème MSOND

On remarque que la solution obtenue après 5 minutes d'exécution est deux fois moins grande que la solution initiale. Pour l'instance eil101 avec 500 demandes, ce rapport atteint 2.5. L'algorithme se stabilise relativement vite. En effet, pour un grand nombre d'instances parmi les plus petites, la meilleure solution a été trouvée au bout de 5 minutes et n'a plus été modifiée par la suite. Utiliser un deuxième opérateur de voisinage qui perturberait la solution permettrait peut être d'améliorer de nouveau cette solution.

Pour certaines instances plus grandes, l'algorithme permettrait encore d'améliorer la solution lorsque le temps limite de 30 minutes a été atteint. C'est par exemple le cas de l'instance ch150 avec 60 ou 80 demandes, de l'instance d198 avec 200 demandes, l'instance gil262 avec 80 demandes et l'instance pr439 avec 40, 200 ou 500 demandes. On pourrait donc espérer trouver une meilleure solution.

7.3.2.2 Résultats expérimentaux pour le problème PMSND

Nous présentons maintenant les résultats expérimentaux que nous avons obtenus pour le problème PMSND. Les figures 7.11 et 7.12 représentent l'évolution de la valeur de la meilleure solution en fonction du nombre d'itérations pour les instances berlin52 et eil101 avec 20 demandes et pour différentes valeurs de α.

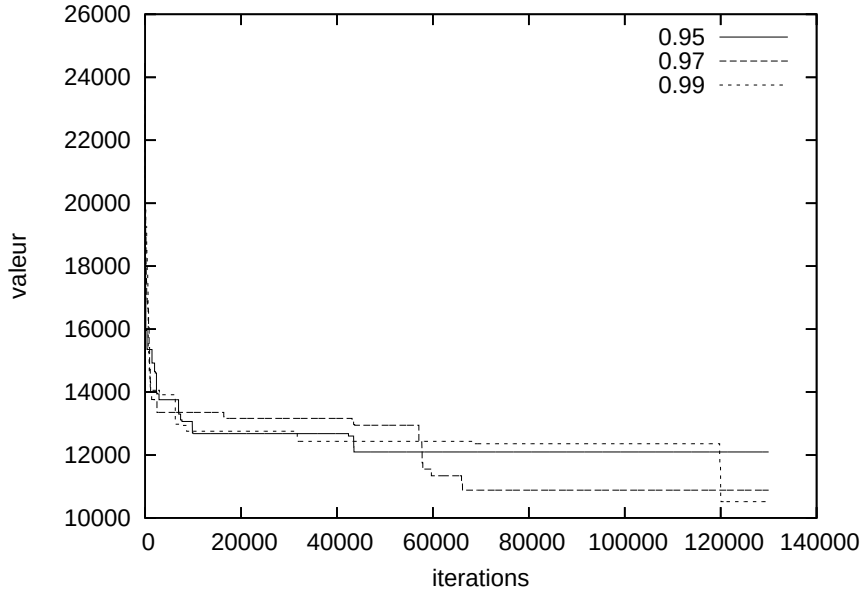


FIG. 7.11 – Instance berlin52 et 20 demandes pour α=0.95, 0.97 et 0.99

Comme pour le problème MSOND, nous n'avons pas pu déterminer une valeur pour

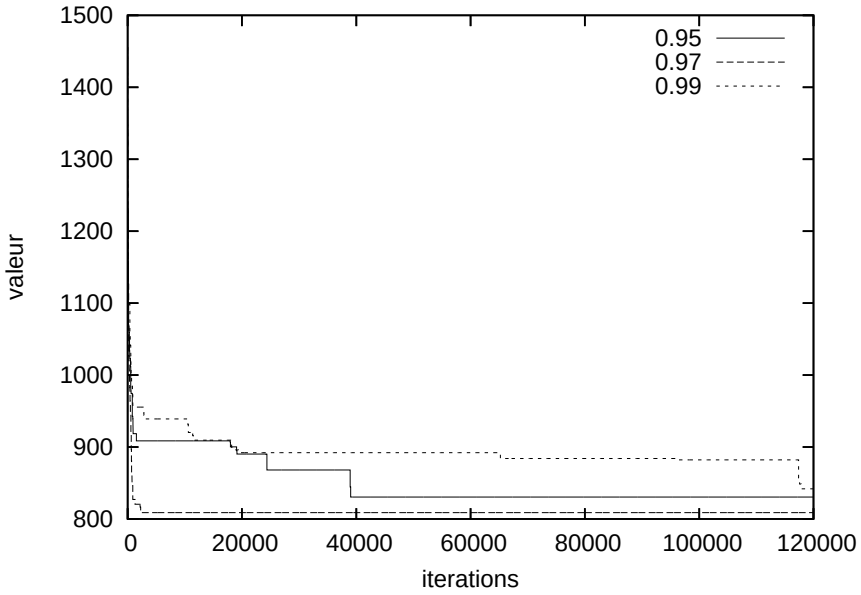


FIG. 7.12 – Instance eil101 et 20 demandes pour α=0.95, 0.97 et 0.99

α qui permette de trouver une bonne solution pour toutes les instances. Nous avons choisi de prendre $\alpha = 0.95$ comme pour le problème MSOND.

La table 7.2 regroupe les résultats obtenus pour le problème PMSND avec $\alpha = 0.95$.

Cette table est similaire à la table 7.1. La valeur de la solution trouvée représente la somme des coûts sur les arêtes du graphe G^1 et des coûts sur les arêtes G^2. On peut remarquer que puisque les coûts sur G^1 sont unitaires, ils n'influent pas beaucoup sur cette valeur. Néanmoins la relation forte entre les chemins de G^1 et les chemins de G^2 donne une place importante à la topologie de la couche cliente, comme à celle de la couche transport.

L'amélioration entre la solution initiale et la solution après 5 minutes est identique à celle de la table 7.1. En effet, en moyenne, elle est réduite de moitié. On remarque que toutes les instances ont été arrêtées au bout de 30 minutes. Mais pour les instances berlin52 et eil101 avec tous les ensembles de demandes, la meilleure solution trouvée l'avait été au bout de 5 minutes.

Nous pouvons également remarquer que 8 instances ne semble pas s'être stabilisées après 30 minutes d'exécution. Nous pouvons donc espérer améliorer la meilleure solution en accordant un temps limite plus élevé.

Inst	$\|V^1\|$	$\|V^2\|$	$\|K\|$	Init	5 min	10 min	20 min	30 min
berlin52	40	52	20	24451,50	**12097,30**	12097,30	12097,30	12097,30
berlin52	40	52	40	42407,30	**24323,30**	24323,30	24323,30	24323,30
berlin52	40	52	60	61879,80	**36093,40**	36093,40	36093,40	36093,40
berlin52	40	52	80	83814,40	**45248,30**	45248,30	45248,30	45248,30
berlin52	40	52	100	96095,70	**50152,40**	50152,40	50152,40	50152,40
berlin52	40	52	200	173690,00	**92441,70**	92441,70	92441,70	92441,70
berlin52	40	52	500	341439,00	**165374,00**	165374,00	165374,00	165374,00
eil101	70	101	20	1455,61	890,29	**830,55**	830,55	830,55
eil101	70	101	40	2856,84	**1735,20**	1735,20	1735,20	1735,20
eil101	70	101	60	3913,97	**2209,19**	2209,19	2209,19	2209,19
eil101	70	101	80	5176,48	**3012,12**	3012,12	3012,12	3012,12
eil101	70	101	100	6203,43	**3676,36**	3676,36	3676,36	3676,36
eil101	70	101	200	11969,10	**5589,58**	5589,58	5589,58	5589,58
eil101	70	101	500	27175,60	**11240,30**	11240,30	11240,30	11240,30
ch150	120	150	20	14176,90	**8136,27**	8136,27	8136,27	8136,27
ch150	120	150	40	27443,70	**17511,50**	17511,50	17511,50	17511,50
ch150	120	150	60	41760,70	24668,00	24234,60	23114,80	**22404,80**
ch150	120	150	80	55266,20	31669,00	**30994,80**	30994,80	30994,80
ch150	120	150	100	69018,10	**43019,40**	43019,40	43019,40	43019,40
ch150	120	150	200	141387,00	76283,00	**76256,40**	76256,40	76256,40
ch150	120	150	500	326610,00	157887,00	152258,00	**151581,00**	151581,00
d198	160	198	20	52937,50	27601,60	27398,50	**26786,50**	26786,50
d198	160	198	40	88837,50	53833,70	53818,60	48648,80	**48321,60**
d198	160	198	60	133148,00	**92729,00**	92729,00	92729,00	92729,00
d198	160	198	80	174550,00	**112395,00**	112395,00	112395,00	112395,00
d198	160	198	100	212881,00	**132683,00**	132683,00	132683,00	132683,00
d198	160	198	200	403783,00	208092,00	193356,00	180868,00	**172281,00**
d198	160	198	500	991888,00	490001,00	448312,00	432368,00	**432045,00**
gil262	220	262	20	4014,01	2790,79	2790,79	**2395,72**	2395,72
gil262	220	262	40	7981,13	5137,88	**5043,96**	5043,96	5043,96
gil262	220	262	60	12356,10	7665,19	**7420,65**	7420,65	7420,65
gil262	220	262	80	16552,90	10296,50	9939,63	9690,96	**9497,09**
gil262	220	262	100	21017,00	**13102,80**	13102,80	13102,80	13102,80
gil262	220	262	200	40941,70	23139,60	23139,60	**23111,00**	23111,00
gil262	220	262	500	99742,10	55735,30	49595,50	**48387,50**	48387,50
pr439	380	439	20	166126,00	89567,60	86721,00	**86703,50**	86703,50
pr439	380	439	40	361506,00	229956,00	229101,00	229025,00	**228334,00**
pr439	380	439	60	519186,00	319891,00	**319630,00**	319630,00	319630,00
pr439	380	439	80	685675,00	424514,00	**419195,00**	419195,00	419195,00
pr439	380	439	100	834078,00	546629,00	530576,00	**528908,00**	528908,00
pr439	380	439	200	1691370,00	1178340,00	1058610,00	1038340,00	**1032310,00**
pr439	380	439	500	4338850,00	3231480,00	2705930,00	2397160,00	**2335430,00**

TAB. 7.2 – Résultats expérimentaux pour le problème PMSND

7.4 Conclusion

Dans ce chapitre, nous avons présenté le problème de sécurisation multicouche du réseau optique et le problème de sécurisation multicouche dans un modèle Peer. Nous avons également décrit une extension du deuxième. Pour ces trois problèmes, nous avons donné une formulation sous la forme d'un programme linéaire en nombres entiers. Nous avons également décrit des métaheuristiques basées sur la méthode du recuit simulé pour résoudre le problème MSOND et le problème PMSND. Nous avons appliqué ces algorithmes à des instances tirées de la TSP Library [76]. Nous avons présenté quelques résultats préliminaires. Ces algorithmes restent encore à être améliorés et il serait intéressant d'exploiter nos formulations pour développer un algorithme de coupes et branchements pour les problèmes afin de trouver une solution exacte. Cela nous permettrait entre autres, de vérifier la qualité de nos solutions obtenues heuristiquement. Il serait également intéressant de développer d'autres métaheuristiques pour pouvoir les comparer à celles basées sur le recuit simulé.

Conclusion

Dans cette thèse, nous avons étudié quatre problèmes d'optimisation dans les réseaux de télécommunications multicouches.

Dans un premier temps, nous avons étudié le problème de sécurisation multicouche du réseau IP. Pour ce problème, nous avons donné une modélisation sous forme de graphes, ainsi qu'une formulation comme un programme linéaire en nombres entiers. Nous avons ensuite étudié une approche polyédrale pour le problème, ce qui nous a amenés à définir de nouvelles classes de contraintes valides. Nous avons étudié l'aspect facial de ces contraintes. En particulier, nous avons décrit des conditions pour que ces contraintes définissent des facettes. Nous avons également introduit des opérations de réduction de graphe définies par rapport à une solution de la relaxation linéaire du problème. Celles-ci permettent de ramener la recherche de contraintes violées dans des graphes réduits. Ces opérations ainsi que les contraintes identifiées ont été par la suite utilisées pour développer un algorithme de coupes et branchements pour le problème de sécurisation multicouche du réseau IP. Ce dernier est mis en œuvre pour résoudre des instances aléatoires du problème ainsi que des intances réelles fournies par France Télécom.

Nous nous sommes ensuite intéressés à une extension de ce problème qui considère également le dimensionnement du réseau IP en plus de la sécurisation. Nous avons étudié deux versions du problème de sécurisation multicouche avec capacités du réseau IP (avec ou sans liaisons multiples). Pour chacune des deux versions, nous avons donné deux formulations de ce problème comme des programmes linéaires mixtes faisant intervenir soit des variables de flot associées aux liaisons du réseau, soit des variables de flot associées aux chemins. Dans le deuxième cas, le nombre de variables étant exponentiel, nous avons utilisé une méthode de génération de colonnes pour résoudre la relaxation linéaire du programme linéaire mixte. Nous avons introduit de nouvelles classes de contraintes valides. Nous avons décrit des conditions pour que certaines de ces contraintes définissent des facettes du polytope des solutions. Ces résultats ont conduit au développement, pour chaque version du problème, d'un algorithme de coupes et branchements

et d'un algorithme de coupes, génération de colonnes et branchements. Ceux-ci ont été utilisés pour résoudre des instances aléatoires du problème, mais aussi des instances réelles ayant un grand intérêt pour les opérateurs de télécommunications.

Enfin, dans une dernière partie, nous avons considéré le problème de sécurisation multicouche du réseau optique et le problème de sécurisation multicouche dans un modèle Peer. Nous avons introduit une formulation pour ces problèmes en termes de programmes linéaires en nombres entiers. Nous avons développé une métaheuristique de type recuit simulé pour résoudre ces problèmes de manière approchée.

Ce travail ouvre des perspectives très intéressantes, aussi bien d'un point de vue polyédral qu'algorithmique. Tout d'abord, sur le plan polyédral, nous allons tenter d'identifier de nouvelles contraintes. Cette investigation polyédrale nous permettra surtout d'améliorer notre algorithme de coupes et branchements et de coupes, génération de colonnes et branchements. D'un point de vue algorithmique, il serait utile d'améliorer les algorithmes de séparation des différentes classes de contraintes. Il serait aussi intéressant de caractériser le polytope associé au problème de sécurisation multicouche du réseau IP pour certaines classes de graphes, comme par exemple les graphes série-parallèles, qui ont une topologie creuse très proche de celle utilisée dans la pratique. Une autre perspective intéressante consiste à faire une étude de nos formulations entières pour les deux derniers problèmes. Elles pourraient être utilisées pour développer des algorithmes exacts pour ces problèmes. Il serait également très intéressant d'étendre ces deux problèmes en considérant également le dimensionnement du réseau en plus de sa sécurisation.

Une autre extension possible des problèmes étudiés, qui mérite d'être aussi examinée, est de supposer des types de connexité différents sur les sommets comme par exemple la topologie dite (1,2)-fiable où les sommets ont des types de connexité 1 ou 2 selon leur importance dans le réseau. Cette topologie a un grand intérêt pour les opérateurs de télécommunications.

Glossaire

ATM Asynchronous Transfert Mode
ATM est une technologie de réseau récente permettant de transférer simultanément sur une même ligne des données et de la voix. Le réseau ATM transfère les données de façon asynchrone, ce qui signife qu'il transmet dès qu'il le peut.

BGP Border Gateway Protocol
BGP est un protocole de routage entre des systèmes autonomes. (Un système autonome est un réseau ou un groupe de réseaux ayant même administration et même routage.)

CBR Constraint-Based Routing
CBR est un protocole de routage permettant de calculer des chemins en respectant des contraintes.

DWDM Dense WDM
La technologie WDM est dense si l'espace entre deux longueurs d'onde permises est inférieure à 100 GHz.

FA Forwarding Adjacency
La FA consiste en la création de liens virtuels à travers le réseau optique, permettant la propagation de l'information sur les lightpaths entre routeurs.

FEC Forward Equivalence Class
Les FEC sont des classes d'équivalence définies en fonction de la qualité de service et du TE, permettant de classer les paquets à leur entrée dans le backbone MPLS. Chaque paquet d'une même FEC suivra le même chemin.

GMPLS Generalized MPLS

GMPLS est une généralisation du plan de commande MPLS pouvant être utilisé sur la couche de transport comme sur la couche cliente.

ICMP Internet Control Message Protocol
ICMP est contenu dans le protocole IP et gère les messages de contrôle permettant de signaler l'apparition d'erreurs.

IETF Internet Engineering Task Force

IGP Interior Gateway Protocol
L'IGP est un ensemble de protocoles gérant le routage intérieur d'un réseau.

IP Internet Protocol
Protocole permettant l'élaboration et le transport des datagrammes IP (paquets de données). Il défini leur représentation, leur routage et leur expédition.

IS-IS Intermediate System - Intermediate System
IS-IS est un protocole de routage à état de liens comme OSPF.

ISO International Standard Organisation

LDP Label Distribution Protocol
LDP est un protocole permettant de distribuer les labels à tous les LSR.

LMP Link Management Protocol
LMP fait partie de GMPLS et gère les canaux de contrôle.

LSP Label Switched Path
Un LSP est l'ensemble des LSR utilisés par une même FEC.

LSR Label Switching Router
Les LSR sont les routeurs, dans un environnement MPLS, permutant les labels. Les edge LSR sont situés à la périphérie du backbone MPLS (Ingress LSR à l'entrée et Egress LSR à la sortie).

MPLS Multi Protocol Label Switching
MPLS est un système basé sur la commutation de label, permettant de gérer la signalisation et le routage dans la couche cliente.

NNI Network to Network Interface
NNI est une interface logique entre deux sous-réseaux optiques.

OCT Optical Channel Trail
Les OCT sont les circuits établis sur un OTN.

OSI Open System Interconnection
L'OSI est le modèle de base pour les réseaux, défini par l'ISO. Il comporte 7 niveaux : la couche physique, la couche liaison de données, la couche réseau, la couche de transport, la couche session, la couche présentation et la couche application.

OSPF Open Shortest Path First
OSPF est un protocole de routage à état de liens. Il envoie aux routeurs adjacents l'état de la liaison qui les sépare. Chaque routeur est donc capable de dresser une carte de l'état du réseau et peut par conséquent choisir à tout moment la route la plus appropriée pour un message donné.

OTN Optical Transport Networks
Ce sont des réseaux de transmission haute capacité capables de gérer efficacement les énormes capacités de la transmission sur fibre optique.

OXC Optical Cross-Connect
Les OXC sont des brasseurs optiques (centres d'aiguillage où les différentes longueurs d'onde sont orientées vers leur fibre de destination). Il existe des brasseurs opaques et des brasseurs transparents.

PDH Plesiochronous Digital Hierarchy
PDH est une technique de multiplexage temporel. Le principal défaut est qu'il faut démultiplexer l'ensemble des voix pour accéder à une seule.

RFC Request For Comments
Les RFC sont un ensemble de documents contenant les spécifications techniques sur divers points de la suite de protocoles TCP/IP.

RIP Routing Information Protocol

RIP est un protocole de routage où chaque routeur communique aux autres routeurs le nombre de sauts qui les sépare. RIP utilise un algorithme de Bellman-Ford.

SDH Synchronous Digital Hierarchy
SDH est une technique de multiplexage temporel permettant l'insertion et l'extraction d'une voix sans avoir à démultiplexer toutes les voix.

TCP Transport Control Protocol
Le protocole TCP permet d'avoir un service de transfert de haute fiabilité entre deux ordinateurs en mode connecté.

TE Traffic Engineering
Le TE permet un meilleur emploi des liens de secours du réseau.

UDP User Datagram Protocol
Le protocole UDP s'appuie sur IP et permet à une application d'envoyer un message court à une autre application. Il ne garantit pas la délivrance du paquet.

UNI User to Network Interface
UNI est une interface logique entre un sous-réseau optique et un sous-réseau IP.

VPN Virtual Private Network
Un VPN est un ensemble de sites placés sous une même autorité administrative, ou groupés suivant un intérêt particulier.

WDM Wavelenght Division Multiplexing
WDM est une technique de multiplexage fréquentiel. Elle permet d'injecter simultanément dans la même fibre plusieurs trains de signaux numériques avec des longueurs d'onde distinctes.

Bibliographie

[1] http ://www.informatik.uni-koeln.de/abacus/.

[2] http ://www.ilog.com/products/cplex/.

[3] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows : Theory, Algorithms, and Applications.* Prentice Hall, New Jersey, 1993.

[4] A. Arvidsson and A. Krzesinski. The design of optimal multi-service mpls networks. *Telektronikk 97 (2/3)*, pages 116–129, 2001.

[5] A. Atamtürk and D. Rajan. On splittable and unsplittable flow capacitated network design arc-set polyhedra. *Mathematical Programming 92*, pages 315–333, 2002.

[6] D. Avis. On the extreme rays of the metric cone. *Canadian Journal of Mathematics 32*, pages 126–144, 1980.

[7] M. Baïou, F. Barahona, and A. R. Mahjoub. Separation of partition inequalities. *Mathematics of Operation Research 25*, pages 243–254, 2000.

[8] F. Barahona. Separating from the dominant of the spanning tree polytope. *Operations Research Letters 12*, pages 201–203, 1992.

[9] F. Barahona and A. R. Mahjoub. On two-connected subgraph polytopes. *Discrete Mathematics 147*, pages 19–34, 1995.

[10] C. Barnhart, C. A. Hane, E. L. Johnson, and G. Sigismondi. A column generation and partitioning approach for multi-commodity flow problems. *Telecommunication Systems 3*, pages 239–258, 1995.

[11] C. Barnhart, C. A. Hane, and P. H. Vance. Using Branch-and-Price-and-Cut to solve origin-destination integer multicommodity Flow problem. *Operations Research 48*, pages 318–326, 2000.

[12] C. Barnhart, E. L. Johnson, G. L. Nemhauser, G. L. Savelsberg, and P. H. Vance. Branch-and-price : Column generation for solving huge integer programs. *Operations Research 46*, pages 316–329, 1998.

[13] D. Bienstock and G. Muratore. Stong inequalities for capacitated survivable network design problems. *Mathematical Programming 89*, pages 127–147, 2000.

[14] S. Borne, E. Gourdin, B. Liau, and A. R. Mahjoub. Design of survivable IP-over-optical networks. *Annals of Operations Research 146*, pages 41–73, 2006.

[15] G. Calvignac, B. Fondeviole, J-L. Le Roux, D. Sol, M. Bouillon, L. Le Beller, J. Legras, M. Le Foll, and K. Mehadji. Synthèse des travaux IETF, OIF et ITU sur l'interaction entre les plans de commande IP et optique. Technical report, France Télécom R & D, Février 2002.

[16] S. Chopra. The k-edge connected spanning subgraph polyhedron. *SIAM Journal on Discrete Mathematics 7*, pages 245–259, 1994.

[17] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings 3rd Annual ACM Symposium on Theory of Computing*, pages 151–158, New York, 1971. Association for Computing Machinery.

[18] G. Cornuéjols, J. Fonlupt, and D. Naddef. The traveling salesman problem on a graph and some related integer polyhedra. *Mathematical Programming 33*, pages 1–27, 1985.

[19] B. Craignou. Synthèse des projets EURESCOM FASHION (P1012) et SCORPION (P1116) concernant l'évolution des réseaux de transport optiques. Technical report, France Télécom R & D, Avril 2002.

[20] W. H. Cunningham. Optimal attack and reinforcement of a network. *Journal of Association for Computing Machinery 32*, pages 549–561, 1985.

[21] G. Dahl and M. Stoer. A Cutting Plane Algorithm for Multicommodity Survivable Network Design Problems. *INFORMS Journal on Computing 10*, pages 1–11, 1998.

[22] G. Dahl and M. Stoer. A Cutting Plane Algorithm for Multicommodity Survivable Network Design Problems. *INFORMS Journal on Computing 10*, pages 1–11, 1998.

[23] G. Dantzig and P. Wolfe. Decomposition principle for linear programming. *Operations Research 8*, pages 101–111, 1960.

[24] M. Didi Biha. *Graphes k-arête connexes et polyèdres.* PhD thesis, Université de Bretagne Occidentale, 1998.

[25] M. Didi Biha and A. R. Mahjoub. k-edge connected polyhedra on series-parallel graphs. *Operations Research Letters 19*, pages 71–78, 1996.

[26] M. Didi Biha and A. R. Mahjoub. The k-edge connected subgraph problem I : Polytopes and critical extreme points. *Linear Algebra and its Applications 381*, pages 117–139, 2004.

[27] E. W. Dijkstra. A note on two problems in connection with graphs. *Numerische Mathematik 1*, pages 269–271, 1959.

[28] J. Edmonds. Covers and packings in a family of sets. *Bull. American Mathematical Society 68*, pages 494–499, 1962.

[29] J. Edmonds. Maximum matching and a polyhedron with 0,1-vertices. *Journal of Research of the National Bureau of Standards (B) 69*, pages 9–14, 1965.

[30] M. Elf, C. Gutwenger, M. Jünger, and G. Rinaldi. Branch-and-Cut Algorithms for Combinatorial Optimization and Their Implementation in ABACUS. In M. Jünger and D. Naddef, editors, *Computational Combinatorial Optimization*, LNCS 2241, pages 157–222. 2001.

[31] R. E. Erikson, C. L. Monma, and Jr. A. F. Veinott. Send-and-plit method for minimum-concave-cost network flows. *Mathematics of Operations Research 12*, pages 634–664, 1987.

[32] B. Feng, A-G. Karasen, P. T. Huth, and B. Slagsvold. State-of-the-art of IP routing. *Telektronikk 97 (2/3)*, pages 130–144, 2001.

[33] L. R. Ford and D. R. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics 8*, pages 399–404, 1956.

[34] B. Fortz, T. McCormick, A. R. Mahjoub, and P. Pesneau. Two-edge connected subgraphs with bounded rings : Polyhedral results and Branch-and-Cut. *Mathematical Programming 105*, pages 85–111, 2005.

[35] M. R. Garey and D. S. Johnson. *Computers and Intractability : A Guide to the Theory of NP-completeness*. Freeman, San Francisco, 1979.

[36] P. Gilmore and R. E. Gomory. A linear programming approach to the cutting stock problem. *Operations Research 9*, pages 849–859, 1961.

[37] P. Gilmore and R. E. Gomory. A linear programming approach to the cutting stock problem - part II. *Operations Research 11*, pages 863–888, 1963.

[38] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum-flow problem. *Journal of the Association for Computing Machinery 35*, pages 921–940, 1988.

[39] R. E. Gomory and T. C. Hu. Multi-terminal network flows. *Journal of the Society for Industrial and Applied Mathematics 9*, pages 551–570, 1961.

[40] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimisation. *Combinatorica 1*, pages 70–89, 1981.

[41] M. Grötschel and C. L. Monma. Integer polyhedra arising from certain networks design problems with connectivity constraints. *SIAM Journal of Discrete Mathematics 3*, pages 502–523, 1990.

[42] M. Grötschel, C. L. Monma, and M. Stoer. Polyhedral approaches to network survivability. In F. Hwang F. Roberts and C. Monma, editors, *Reliability of Computer and Communication Networks 5*, Series Discrete Mathematics and Computer Science, pages 121–141. AMS/ACM, 1991.

[43] M. Grötschel, C. L. Monma, and M. Stoer. Computational results with a cutting plane algorithm for designing communication networks with low-connectivity constraints. *Operations Research 40*, pages 309–330, 1992.

[44] M. Grötschel, C. L. Monma, and M. Stoer. Facets for polyhedra arising in the design of communication with low-connectivity constraints. *SIAM Journal on Optimisation 2*, pages 474–504, August 1992.

[45] M. Grötschel, C. L. Monma, and M. Stoer. Design of survivable networks. *Handbooks in Operations Research and Management Science 7,* M.O. Ball et al. (Eds), pages 617–671, 1995.

[46] D. Gusfield. Very simple algorithms and programs for all pairs network flow analysis. Technical report, Computer Science Division, University of California, Davis, 1987.

[47] D. Gusfield. Very simple methods for all pairs network flow analysis. *SIAM Journal of Computing 19*, pages 143–155, 1990.

[48] D. Huygens, M. Labbé, A. R. Mahjoub, and P. Pesneau. Two edge-disjoint hop-constrained paths : Valid inequalities and branch-and-cut. *Networks, A paraitre*, 2006.

[49] D. Huygens, A. Ridha Mahjoub, and P. Pesneau. Two edge hop-constrained paths and polyhedra. *SIAM Journal on Discrete Mathematics 18*, pages 287–312, 2004.

[50] M. Iri. On an extension of the maximum-flow minimum-cut theorem to multicommodity flows. *Journal of Operations Research Society of Japan 13*, pages 129–135, 1971.

[51] T. Jensen. Internet protocol and transport protocols. *Telektronikk 97 (2/3)*, pages 20–38, 2001.

[52] K. L. Jones, I. J. Lustig, J. M. Farvolden, and W. B. Powell. Multicommodity networks flows : The impact of formulation on decomposition. *Mathematical Programming 62*, pages 95–117, 1993.

[53] H. Kerivin. *Réseaux fiables et polyèdres.* PhD thesis, Université Blaise Pascal, Clermont-Ferrand II, 2000.

[54] H. Kerivin and A. R. Mahjoub. Design of Survivable Networks : A Survey. *Networks 46*, pages 1–21, 2005.

[55] H. Kerivin, A. R. Mahjoub, and C. Nocq. $(1,2)$-Survivable Networks : Facets and Branch-and-Cut. In M. Grötschel, editor, *The Sharpest Cut : The Impact of Manfred Padberg and his Work*, pages 121–150. MPS-SIAM Series on Optimization, 2004.

[56] J. B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society 7*, pages 48–50, 1956.

[57] I. Loiseau, A. Ceselli, N. Maculan, and Matteo Salani. Génération de colonnes en programmation linéaire en nombres entiers. In V. Th. Paschos, editor, *Optimisation combinatoire : Concepts fondamentaux.* chapitre 8. Hermès, Paris, 2005.

[58] M. V. Lomonosov. Combinatorial approaches to multiflow problems. *Discrete Applied Mathematics 11*, pages 1–93, 1985.

[59] M. E. Lübbecke and J. Desrosiers. Selected Topics in Column Generation. *Operations Research 53*, pages 1007–1023, 2005.

[60] T. L. Magnanti, P. Mirchandani, and R. Vachani. The convex hull of two core capacitated network design problems. *Mathematical Programming 60*, pages 233–250, 1993.

[61] T. L. Magnanti, P. Mirchandani, and R. Vachani. Modeling and solving the two-facility capacitated network loading problem. *Operations Research 43*, pages 142–157, 1995.

[62] T. L. Magnanti and S. Raghavan. Strong formulations for network design problems with connectivity requirements. *Networks 45*, pages 61–79, 2005.

[63] A. R. Mahjoub. Two-edge connected spanning subgraphs and polyhedra. *Mathematical Programming 64*, pages 199–208, 1994.

[64] A. R. Mahjoub. On perfectly two-edge connected graphs. *Discrete Mathematics 170*, pages 153–172, 1995.

[65] A. R. Mahjoub. Approches Polyédrales. In V. Th. Paschos, editor, *Optimisation combinatoire : Concepts fondamentaux.* chapitre 9. Hermès, Paris, 2005.

[66] A. R. Mahjoub and C. Nocq. On the linear relaxation of the 2-node connected subgraph polytope. *Discrete Applied Mathematics*, pages 389–416, 1999.

[67] K. Menger. Zur allgemeinen kurventheorie. *Fundamanta Mathematicae 10*, pages 96–115, 1927.

[68] T. L. Mirchandani. *Polyhedral structure of the capacitated network design problem with an application to the telecommunication industry.* PhD thesis, MIT, Cambridge, 1989.

[69] M. W. Padberg and G. Rinaldi. A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Review 33*, pages 60–100, 1991.

[70] P. Pesneau. *Conception de réseaux 2-connexe avec contraintes de bornes.* PhD thesis, Université Blaise Pascal, Clermont-Ferrand II, 2003.

[71] IETF. 1981. J. Postel. *Internet Protocol. DARPA Internet Program Protocol Specification*, (RFC791).

[72] IETF. 1981. J. Postel. *Transmission Control Protocol. DARPA Internet Program Protocol Specification*, (RFC793).

[73] R. C. Prim. Shortest connection networks and some generalization. *Bell System Technical Journal 36*, pages 1389–1401, 1957.

[74] G. Pujolle. *Les réseaux.* Eyrolles, 2005.

[75] B. Rajagopalan, D. Pendarakis, D. Saha, R. S. Ramamoorthy, and K. Bala. IP over optical networks : Architectural aspects. *IEEE Communications Magazine*, pages 94–102, Sept. 2000.

[76] G. Reinelt. TSPLIB - a traveling salesman problem library. *ORSA Journal on Computing 3*, pages 376–384, 1991.

[77] J. L. Le Roux and G. Calvignac. GMPLS for IP/OTN : Concepts, architectures & implementations status. Séminaire, France Telecom R&D/DAC, Décembre 2002.

[78] A. Schrijver. *Combinatorial Optimization : Polyhedra and Efficiency.* Algorithms and Combinatorics 24. Springer-Verlag, 2003.

[79] M. Stoer. *Design of survivable networks.* Lectures Notes in Mathematics 1531. Springer-Verlag, 1992.

[80] M. Stoer and G. Dahl. A polyhedral approach to multicommodity survivable network design. *Numerische Mathematik 68*, pages 149–167, 1994.

[81] S. Thienel. *ABACUS - A Branch-And-CUt System.* PhD thesis, Universität zu Köln, 1995.

[82] F. Vanderbeck. *Decomposition and Column Generation for Integer Programming.* PhD thesis, Université catholique de Louvain, Louvain, Belgique, 1994.

[83] P. Winter. Generalized Steiner problem in series-parallel networks. *Journal of Algorithms 7*, pages 549–566, 1986.