Algorithmique et télécommunications : Coloration et multiflot approchés et applications aux réseaux d'infrastructure

Hervé Rivano

▶ **To cite this version:**

Hervé Rivano. Algorithmique et télécommunications : Coloration et multiflot approchés et applications aux réseaux d'infrastructure. Réseaux et télécommunications [cs.NI]. Université Nice Sophia Antipolis, 2003. Français. NNT : . tel-00169842

HAL Id: tel-00169842
https://theses.hal.science/tel-00169842

Submitted on 5 Sep 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITÉ de NICE-SOPHIA ANTIPOLIS – UFR SCIENCES

École Doctorale STIC

THÈSE

pour obtenir le titre de

Docteur en SCIENCES

de l'Université de Nice Sophia Antipolis

Spécialité : **INFORMATIQUE**

présentée et soutenue par

Hervé RIVANO

Algorithmique et télécommunications : Coloration et multiflot approchés et applications aux réseaux d'infrastructure

Thèse dirigée par **Afonso FERREIRA** et **Jérôme GALTIER**

Soutenue publiquement le **28 novembre 2003** devant le jury composé de :

Examinateurs	M.	**M.**	**COSNARD**	Professeur
	M.	**M.**	**HABIB**	Professeur
Directeurs	M.	**A.**	**FERREIRA**	Directeur de Recherche
	M.	**J.**	**GALTIER**	Chercheur France Telecom
Rapporteurs	M.	**P.**	**FRAIGNIAUD**	Directeur de Recherche
	M.	**J.**	**ROLIM**	Professeur
	Mme	**C.**	**KENYON**	Professeur

À Luc et Claudie ...

Remerciements

Je tiens à remercier tout particulièrement

- Afonso Ferreira, bien entendu, qui est, plus qu'un directeur de thèse, un ami, merci pour ces longues et précieuses discussions, pour son soutien et ses conseils toujours précieux ;
- Jérôme Galtier, qui a dirigé cette thèse du coté de France Telecom, pour sa gentillesse, sa compréhension et son soutien ;
- Pierre Fraigniaud, Claire Kenyon et José Rolim, pour m'avoir fait l'honneur de relire ma thèse et pour leurs conseils avisés qui m'ont permis de l'améliorer ;
- Michel Cosnard, Pierre Fraigniaud, Michel Habib et José Rolim, pour m'avoir fait l'honneur de constituer le jury de ma thèse malgré leurs emplois du temps plus que saturés ;
- Michel Habib pour avoir accepté de présider ce jury ;
- Jean Claude Bermond sans qui MASCOTTE ne serait pas aussi accueillante, chaleureuse et responsabilisante à la fois ;
- Éphie Dériche et Patricia Lachaume, pour être toujours là quand les méandres de l'administration dépassent mes capacités, souvent donc, pour leur gentillesse surtout. Toutes ces années auraient été bien moins faciles sans elles ;
- Christian Raffaële pour son soutien et son aide si précieux tout au long de la procédure de soutenance et après ;
- Les membres de MASCOTTE, notamment Jean-Claude, Afonso, Éphie, Michel, Aubin, JF, Philippe, Olivier, Séverine, et Fréd, pour leur compagnie fort agréable, l'ambiance excellente qui règne dans l'équipe et le plaisir que l'on trouve à venir travailler chaque matin ;
- Les membres de l'équipe DMI/ISE de France Telecom R&D, notamment Patrick, Jérôme, Alexandre et Sébastien, pour leur accueil et leur gentillesse malgré une situation inhabituelle et mes habitudes de travail pas toujours très malléables ;
- David Coudert pour m'avoir supporté trois longues années comme colocataire et pour continuer à être un indispensable ami ;
- Ma famille, bien sur, je n'ai pas besoin de dire pourquoi, ils sont là et c'est bien ;

– Fabrice et Virgine pour leur amitié exceptionnelle et pour avoir toujours été là quand il le fallait sans que la réciproque ne soit assez vraie ;
– Hubert et Bertrand, pour le souvenir de nos aventures québecquoises et de nos délires au foyer, pour notre amitié si précieuse, et pour cette certitude que l'avenir nous réserve bien d'autres moment sympathiques, autour d'un café ou d'un pot de nutella ;
– François, Vincent, Pierre, Jym et PF (les deux autres tiers de MIM) et tous les vieux adeptes du foyer que la distance n'a pas éloigné ;
– Mes amis trop lointains, notamment Axel, Sandrine, Valérie, H&K, ... Tous les citer est impossible, mais chacun m'est précieux ;
– Ceux qui sont plus près mais tout aussi indispensables, Béné, Blaise et Isa, Jérôme et Val, Sandrine, Céline, Cathy, Assia, ... Sans eux Nice ne serait pas aussi belle ;
– Les camarades du MJS et de NG, mais plus particulièrement Anne-Ju, Séb, Ben, FX, Yann, Nadia, Audrey, Jean-François et Nathy, Fred, Yo, Bassem, ... Avec eux, l'espoir est possible en cette terre de mission que sont les Alpes Maritimes ;

Durant deux ans de ma thèse j'ai été employé par France Télécom Recherche et Développement sur contrat CIFRE. L'entreprise conserve donc la propriété intellectuelle des travaux menés du 1er octobre 2001 au 1er octobre 2003.

Table des matières

Table des figures

Introduction

La maîtrise du secteur des réseaux d'infrastructure de télécommunication est un enjeu stratégique pour tous les opérateurs prétendant à une envergure internationale. Cependant, déployer de tels réseaux nécessite de pouvoir en garantir l'efficacité. L'une des garanties primordiales qu'exigent les opérateurs, le dimensionnement des réseaux, consiste à garantir qu'un réseau supporte un trafic donné sans gaspiller de ressources coûteuses. Les problématiques d'optimisation qui permettent d'atteindre cet objectif sont généralement des plus complexes à résoudre et nécessitent l'utilisation de techniques algorithmiques avancées.

Les travaux présentés dans cette thèse s'intéressent aux techniques algorithmiques aléatoires et d'approximation pour des problématiques fondamentales d'optimisation combinatoire, qui se dégagent de la modélisation – structurelle et algorithmique – du dimensionnement des réseaux d'infrastructure de télécommunication. Nous résolvons notamment la coloration fractionnaire de chemins dans les arbres de degré borné, et l'utilisons pour obtenir une approximation aléatoire du problème classique de coloration de chemins dans un graphe, dont le premier est une relaxation linéaire. Nous développons aussi des algorithmes d'approximation pour le multiflot, fondés à la fois sur un arrondi aléatoire et sur une approche combinatoire de sa relaxation linéaire, le multiflot fractionnaire.

Ces résultats trouvent des applications dans l'affectation de longueurs d'onde dans les réseaux optiques monofibres déployant la technologie de multiplexage en longueurs d'onde (WDM), et dans le routage optique dans les réseaux optiques WDM multifibres avec ou sans conversion en longueurs d'onde.

Les réseaux d'infrastructure des télécommunications, enjeu stratégique

Le développement des télécommunications transforme progressivement le monde en un *village global*, selon l'expression consacrée, et un grand nombre de technologies participent à sa mise en œuvre : les constellations de satellites, malgré un échec commercial et un coût toujours prohibitif, permettent d'être joignable sur la quasi totalité du globe ; les communications téléphoniques mobiles, banalisées dans les pays développés avec les réseaux GSM et considérées dans les pays en voie de développement et les zones rurales comme la solution idéale de télécommunication grâce au faible coût relatif de l'infrastructure à déployer, convergent vers des réseaux multimédia avec les technologies UMTS ou WIFI ; les fibres optiques supportent les réseaux métropolitains à haut débit des zones à forte densité de population et d'activité économique, ainsi que les réseaux d'infrastructure trans et inter-continentaux qui assurent la plupart des connexions internationales.

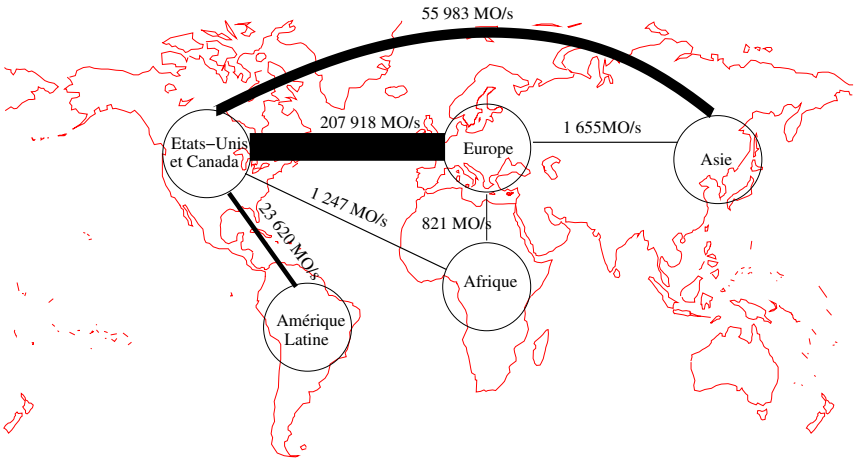


FIG. 1: *Bande passante intercontinentale à la mi-2002 (Mo/s).*

L'importance économique des télécommunications n'est plus à démontrer. La mutation des pays développés vers une économie principalement de service, conjuguée à la mondialisation financière et commerciale, a donné un élan considérable et un caractère stratégique au déploiement des réseaux intercontinentaux à très haut débit : la capacité des liaisons intercontinentales, dont la figure 1 donne l'état à la mi-2002, est très fortement reliée au volume d'échanges commerciaux et financiers (Duval 2003). De plus, c'est un marché en forte croissance, comme l'illustre, pour l'Union Européenne, la figure 2 (Union Européenne 2002).

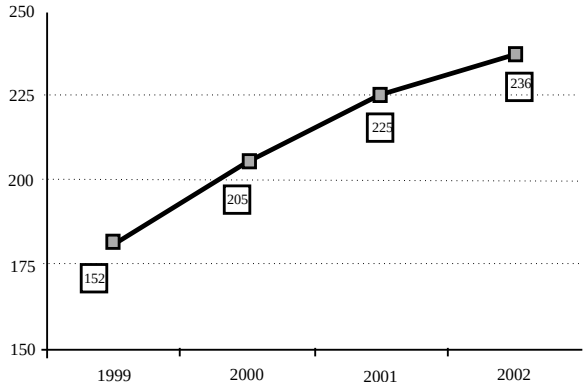


FIG. 2: *Marché des télécommunications de l'Union Européenne (milliards d'euro).*

Dans un tel contexte, il est capital pour tous les opérateurs d'envergure internationale de garder une maîtrise du secteur des réseaux d'infrastructures et de participer à la croissance globale des capacités de communication mondiales. La figure 3 illustre par exemple le lien étroit entre la croissance de la capacité des réseaux longue distance de France Télécom et celle, globale, du lien transatlantique (Mémento FT R&D 19 2002). C'est donc à ce type de réseaux que s'intéressent principalement nos recherches qui visent à développer des algorithmes efficaces pour les problématiques d'optimisation inhérentes aux réseaux de télécommunication.

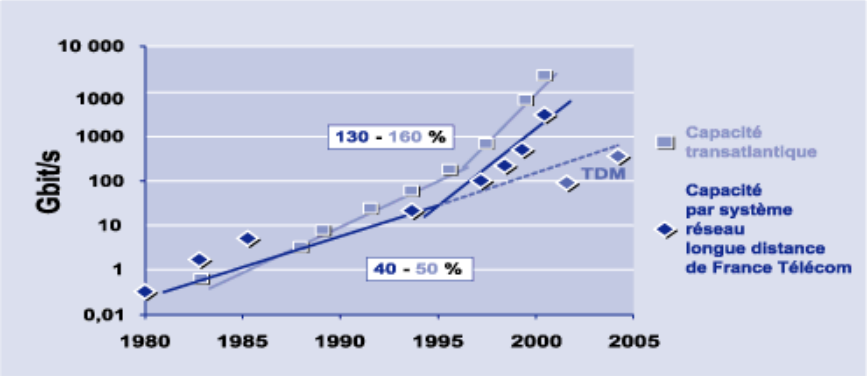


FIG. 3: *Évolution de la capacité par fibre des systèmes optiques longue distance France Télécom et de la capacité disponible sur l'Atlantique Nord.*

Des algorithmes efficaces, indispensables au dimensionnement des réseaux

Les réseaux optiques utilisant les technologies de multiplexage en longueurs d'onde (WDM pour *Wavelength Division Multiplexing*) semblent être la solution la plus appropriée aux réseaux d'infrastructures, inter-continentaux, trans-continentaux, voire métropolitains, de par leur bande passante sans équivalent. Pourtant, du point de vue d'un opérateur les investissements nécessaires sont tels que le déploiement de ce type de réseaux ne se fera pas sans garantie d'efficacité suffisante. Une des garanties primordiales est qu'un réseau qui supporte un certain trafic ne gaspille pas de ressources, autrement dit qu'il ait été correctement dimensionné.

Ce type de problématique d'optimisation implique que nos algorithmes ont vocation à guider l'installation ou la configuration d'équipements physiques sur le réseau. Le temps de calcul n'est donc pas la ressource la plus critique comme ce pourrait être le cas d'algorithmes d'optimisation à la volée pour des situations dynamiques. Malgré tout, les systèmes de communication sur le terrain sont souvent de très grande taille et ont tendance à croître rapidement comme l'indiquent les figures 1 et 3. De plus, la complexité des problèmes que nous traitons est telle qu'il est impossible d'optimiser autre chose que des *exemples jouets* de très petite taille. Pour être en mesure de traiter des instances réelles, nous cherchons à produire des solutions non optimales tout en maîtrisant la complexité en temps, mais aussi en mémoire, de nos algorithmes. Néanmoins, conformément à l'objectif initial, la principale préoccupation de nos travaux reste de pouvoir garantir que la qualité de nos solutions est la plus proche possible de l'optimum.

Pour autant, les problématiques algorithmiques d'optimisation qui ont été abordées au cours de cette thèse n'ont pas comme seul domaine d'application les réseaux d'infrastructure. En effet, une modélisation structurelle et algorithmique pertinente de ces réseaux par des notions d'optimisation combinatoire nous permet de faire émerger des problématiques fondamentales partagées par la plupart des réseaux de télécommunication. En particulier, les problématiques de routage que nous présentons dans les chapitres 3 et 5 sont très générales[1]. Un autre exemple est notre modélisation des réseaux multi-

[1]Nous avons, par exemple, étudié les problématiques de routage et re-routage dans les constellations de satellites (Ferreira, Galtier, Petit, et Rivano 2001).

fibres donnée au chapitre 3 qui utilise des structures combinatoires que l'on retrouve dans le cadre des réseaux radio *ad hoc*[2]. Enfin, la coloration de chemins, étudiée au chapitre 4, s'applique aussi bien à l'optimisation de *l'ordonnancement d'appels*[3] pour les réseaux ATM ou SDH.

Méthodologie générale

Notre recherche s'articule autour de deux axes complémentaires. Nous cherchons tout d'abord des descriptions combinatoires des problèmes plus sophistiquées que leurs descriptions naturelles. Sophistiqué signifie ici que des transformations algorithmiques préalables et *a posteriori* sont nécessaires pour construire des solutions alors que le cœur de l'optimisation manipule des notions combinatoires complexes. Il est ainsi possible d'agréger l'information d'une description complète dans des variables de plus haut niveau, mais moins nombreuses. Les optimisations résultantes sont plus compactes et permettent une plus grande efficacité de résolution exacte ou d'approximation, selon les cas. Les manipulations d'agrégation préalables peuvent s'apparenter parfois à une certaine perte d'information, mais suffisamment contrôlée pour que le travail algorithmique *a posteriori* permette de recréer l'information perdue.

Dans un deuxième temps, nous mettons en œuvre des techniques algorithmiques avancées exploitant des interprétations combinatoires des *relaxations linéaires* des problèmes et des processus aléatoires afin d'obtenir des approximations rapides et performantes.

Nous appliquons cette méthodologie à la modélisation structurelle et algorithmique des réseaux optiques multifibres. Dans ces réseaux, les nœuds sont physiquement connectés par des câbles de plusieurs fibres optiques et les connexions sont réalisées par des faisceaux lasers passant d'un nœud du réseau à un autre à travers ces fibres. L'agrégation de l'information consiste alors à ne pas considérer le trajet d'un faisceau, appelé *chemin optique*, comme la suite des fibres qu'il emprunte, mais plutôt comme la suite des câbles qui abritent ces fibres. Cela transforme la nature même des contraintes *d'affectation de*

[2]Nous cherchons à écrire un protocole d'accès au média pour les réseaux radio *ad hoc* prenant en compte la capacité de brouillage de chaque connexion (Chélius, Coudert, Ferreira, Guérin-Lassous, et Rivano 2003).

[3]La similarité de ces problèmes s'illustre, par exemple, dans (Erlebach et Jansen 1997).

longueurs d'onde inhérentes à la technologie WDM, passant des conflits pairs à pairs, que l'on manipule avec les modélisations usuelles, à des conflits de groupe locaux à chaque lien. La reconstruction du routage complet se fait ensuite en distribuant sur chaque lien les chemins de même longueur d'onde sur des fibres différentes. L'optimisation de l'affectation de longueurs d'onde à des chemins dans le réseau se modélise alors par un problème de coloration d'hypergraphe, qui s'apparente aussi à des problématiques issues de l'optimisation de *l'accès au média* dans les réseaux radio *ad hoc*.

Le problème de coloration de chemins dans un graphe modélise l'affectation de longueurs d'onde dans les réseaux optiques monofibres. Il s'agit d'affecter une couleur à chaque chemin de sorte que deux chemins adjacents aient des couleurs différentes. Nous étudions la relaxation linéaire de ce problème, la *coloration fractionnaire de chemins*, et développons un algorithme performant grâce à une double agrégation d'information. La première consiste à ne plus manipuler des couleurs sur les chemins mais des *stables*, les ensembles de chemins deux à deux disjoints. De la sorte, on perd l'information de la couleur effectivement utilisée sur un chemin pour ne garder que la connaissance des chemins de même couleur. Dans ce cas, ce sont des arguments structurels qui permettent de contrôler la perte d'information puisqu'un étiquetage *a posteriori* de chaque stable par une couleur est trivial et suffisant. La deuxième agrégation consiste à éviter d'utiliser une variable par stable, en nombre exponentiel, en considérant des *variables de traces* qui sont des sommes partielles des précédentes. Cette fois, ce sont des manipulations algorithmiques plus complexes qui permettent de reconstruire l'information manquante puisque le cœur de l'optimisation consiste à calculer *la trace* d'une coloration fractionnaire optimale, tandis que la coloration elle même est construite à partir de cette trace par un algorithme glouton.

Nous étudions enfin le problème du *routage optique* et l'approximation du calcul de *multiflot*. Le problème du routage optique consiste à affecter à chaque *requête de communication* un chemin optique. Les requêtes de communication sont des demandes de connexion point à point, d'un nœud source à un nœud destination. Là encore nous opérons une agrégation d'information en regroupant toutes les requêtes issues d'une même

source en un *multi-unicast*, qui peut alors se modéliser par un flot dans un graphe auxiliaire. D'un chemin par requête, on passe à un flot par source où les chemins sont mélangés indistinctement. Des *marches aléatoires* permettent ensuite de recréer les chemins de la solution entière à partir du trajet du flot fractionnaire dans un processus d'*arrondi aléatoire*.

Organisation détaillée de la thèse

Les techniques algorithmiques que nous développons pour les problématiques issues de la modélisation du dimensionnement des réseaux d'infrastructure se placent dans le domaine de l'algorithmique d'approximation pour l'optimisation combinatoire. Nous commençons donc cette thèse par décrire l'essentiel des notions qui y sont reliées. Le **chapitre 1** donne une description intuitive des notions de *programmation linéaire* et de *programmation linéaire en nombres entiers* les plus utilisées dans nos travaux. Les problématiques d'optimisation des réseaux de télécommunication sont très souvent $\mathcal{NP}$*-difficiles*, ce qui motive la recherche de méthodes de construction de solutions approchées. La nécessité de garanties sur la qualité de ces solutions nous amène à chercher des *algorithmes d'approximation*, dont nous décrivons le contexte d'utilisation. Nous présentons enfin les problèmes classiques d'optimisation combinatoire que sont la coloration de graphes et de chemins et le multiflot, que nous étudierons ensuite. Nous montrons que les problèmes de coloration de graphes et de chemins sont équivalents et présentons une écriture du premier comme un problème de *couverture* des sommets par des *stables*. Les écritures classiques du multiflot en formulation *sommet-arc* et *arc-chemin* sont présentés, ainsi qu'un tour d'horizon des techniques d'approximation les plus performantes.

Le **chapitre 2** présente l'état de l'art des modèles et algorithmes pour l'optimisation des réseaux optiques. Nous décrivons les différentes technologies utilisées dans ces réseaux (multiplexage en longueurs d'onde, routeurs tout optiques, câbles multifibres, conversion en longueurs d'onde, ...) et leur influence dans les modèles combinatoires et les optimisations. Nous concluons ce chapitre sur le besoin d'une modélisation structurelle et algorithmique plus pertinente des réseaux multifibres qui permettra une véritable application de techniques algorithmiques à ces solutions technologiques.

Le **chapitre 3** est consacré à la proposition et la validation d'un modèle pour les réseaux optiques multifibre fondé sur une nouvelle lecture des contraintes d'optimisation dues au multiplexage en longueurs d'onde. Nous mettons en avant la notion de *conflits de groupe* locaux à chaque lien du réseau. Nous la modélisons, d'une part par la structure d'*hypergraphe des conflits*, pour le problème d'affectation de longueurs d'onde, et d'autre part par une capacité dans un réseau de flot, pour le *routage optique* avec ou sans conversion en longueurs d'onde. Ces travaux ont fait l'objet des publications (Rivano 2001), (Coudert et Rivano 2002), (Ferreira, Pérennes, Richa, Rivano, et Stier 2003a), et (Ferreira, Pérennes, Richa, Rivano, et Stier 2003b).

Dans le **chapitre 4**, nous nous intéressons au problème de coloration de chemins par le biais de la relaxation linéaire de son écriture en problème de couverture dérivée de celle de la coloration de graphes : la *coloration fractionnaire de chemins*. Nous donnons un algorithme efficace de calcul d'une telle coloration fractionnaire dans les arbres de degré borné où le problème est polynomial. Nous étendons ce résultat aux graphes de largeur arborescente bornée puis nous montrons comment construire, dans les arbres binaires, une coloration fractionnaire dite *équilibrée* de coût inférieur à $\frac{7}{5}$ fois la charge de l'ensemble de chemins. Nous montrons enfin comment utiliser une coloration fractionnaire optimale dans un processus d'arrondi aléatoire pour obtenir le meilleur algorithme d'approximation sur les arbres de degré borné. Ces travaux ont fait l'objet de la publication (Caragiannis, Ferreira, Kaklamanis, Pérennes, et Rivano 2001) et des techniques similaires ont été employées pour la *coloration contrainte* des arcs d'un graphe biparti (Caragiannis, Ferreira, Kaklamanis, Pérennes, Persiano, et Rivano 2001).

Les problèmes de multiflot servent à la modélisation d'un grand nombre de situations différentes, de la conception de circuits VLSI aux problèmes de transport routier en passant par le routage optique dans les réseaux multifibres. Nous présentons dans le **chapitre 5** une amélioration d'un des algorithmes d'approximation aléatoire les plus efficaces (Raghavan 1994). Notre algorithme est un arrondi aléatoire incrémental où sont introduites des dépendances entre les choix aléatoires par le biais de recalculs de la relaxation linéaire du problème : le *multiflot fractionnaire*. Le temps de calcul de cette relaxation est donc un facteur primordial dans la complexité de notre algorithme. Pour limiter celle-ci, nous

présentons une amélioration d'un des meilleurs algorithmes d'approximation du multiflot fractionnaire en introduisant des techniques de calcul de plus courts chemins dynamiques. Nous donnons aussi des spécialisations de ces algorithmes aux réseaux de flot issus du routage optique. Ces travaux ont fait l'objet des publications (Coudert et Rivano 2002) et (Bouklit, Coudert, Lalande, Paul, et Rivano 2003).

Cette thèse a été menée conjointement au sein du projet commun CNRS/INRIA/UNSA MASCOTTE[4] et de France Télécom R&D[5]. Une partie des travaux présentés a été effectué en collaboration avec des collègues étrangers : Vincenzo Auletta[6], Ioannis Caragiannis[7], Christos Kaklamanis[7], Pino Persiano[6], Andréa Richa[8] et Nicolas Stier Moses[9].

[4] `http://www-sop.inria.fr/mascotte`

[5] `http://www.rd.francetelecom.com`

[6] Dipartimento di Informatica ed Applicazioni, Univertia di Salerno, Baronissi, Italie.

[7] Computer Technology Institute, Patras, Grèce.

[8] Department of Computer Science and Engineering, Arizona State University, Tempe, USA.

[9] MIT Operations Research Center, Cambridge, USA.

Chapitre 1

Quelques notions utiles d'algorithmique

Les méthodes d'optimisation combinatoire forment un vaste pan de l'algorithmique : celui des techniques de résolution (exacte ou approchée) d'un cas, à la fois particulier et très général, de l'optimisation mathématique. Plus exactement, si un problème consiste en la maximisation ou la minimisation sur un ensemble fini d'une fonction à valeurs dans $\mathbb{R}$, alors il s'agit d'un problème d'optimisation combinatoire, la fonction est appelée *fonction objective* et le domaine d'optimisation, *ensemble des solutions réalisables*.

À l'évidence, la minimisation du coût d'un réseau de télécommunication, une des questions que se posent les opérateurs, peut être modélisée par des problèmes d'optimisation combinatoire : l'ensemble des réseaux possibles interconnectant les sites de l'opérateur est fini, et la fonction objective est la somme des coûts de chaque élément du réseau. Cette thèse s'intéresse tout particulièrement à certains aspects algorithmiques d'optimisation combinatoire pour les télécommunications. Il s'agit plus particulièrement d'algorithmique d'approximation, utilisant souvent la programmation linéaire et la programmation linéaire en nombres entiers. Nous en exposons dans ce chapitre les notions les plus utilisées dans nos travaux. Sont supposées connues les notions les plus classiques d'algorithmique et de théorie des graphes, telles que l'algorithmique séquentielle élémentaire, la hiérarchie des classes de complexité, les graphes usuels (généraux, complets, arbres, cycles, etc.), les chemins, etc. Pour un exposé de ces domaines, voir les ouvrages "*Introduction to algorithms*" (Cormen et coll. 1990), "*Graphes*" (Berge 1983), et "*Concrete mathematics*" (Graham et coll. 1994).

Le but de ce chapitre n'est pas de faire un cours complet et rigoureux sur les notions de programmation linéaire et d'optimisation combinatoire, mais d'en présenter certaines du point de vue intuitif de l'algorithmicien, qui les utilise comme des outils de modélisation et d'approximation. Pour une formation complète sur ces domaines, on pourra se reporter aux ouvrages suivants, dont sont tirés la plupart des connaissances exposées dans ce chapitre :

- "*Linear Programing*" (Chvátal 1983) pour l'apprentissage de la programmation linéaire classique ;
- "*A Course in Combinatorial Optimization*" (Schrijver 2000) pour un cours sur les notions les plus importantes en optimisation combinatoire, principalement sous l'angle de l'optimisation convexe, dont la programmation linéaire fait partie ;
- "*Introduction to Linear Optimization*" (Bertimas et Tsitsiklis 1997) pour un état de l'art très complet des techniques de programmation linéaire ;
- "*Complexity and Approximation*" (Ausiello et coll. 1999) pour tout ce qui concerne l'approximation des problèmes d'optimisation combinatoire ;
- "*Randomized Algorithms*" (Motwani et Raghavan 1995) pour un excellent tour d'horizon des techniques aléatoires en algorithmique d'approximation.

Ce chapitre est organisé de la manière suivante. Les sections 1.1 et 1.2 exposent quelques définitions et propriétés de la *programmation linéaire* et de la *programmation linéaire en nombres entiers*, sélectionnées et présentées selon leur utilité dans nos travaux, plus que selon leur intérêt mathématique. Nous verrons, entre autres, que ce qui nous intéresse finalement est la programmation linéaire en nombres entiers, qui est malheureusement $\mathcal{NP}$-difficile, ce qui motive le développement d'heuristiques ou d'algorithmes d'approximation. La section 1.3 justifie le choix de nous orienter vers l'algorithmique d'approximation, et en présente le cadre formel. Enfin, les sections 1.4 et 1.5 présentent, sous l'angle de la programmation linéaire en nombres entiers, les deux problèmes fondamentaux d'optimisation combinatoires que sont la coloration de sommets ou de chemins d'un graphe et le calcul de multiflot, que nous retrouverons dans la suite de cette thèse.

1.1 Programmation linéaire

Le terme *programmation linéaire* (noté LP) rassemble les problèmes d'optimisation d'une fonction objective linéaire c sur un ensemble de solutions réalisables polyédral $\mathcal{C} \subseteq \mathbb{R}^n$. Un polyèdre est l'intersection d'un nombre fini de demi-espaces affines bordés par des hyperplans, qui définissent des contraintes linéaires sur les valeurs que peuvent prendre les variables du problème. Cet outil mathématique, cas particulier de *programmation convexe*, permet de modéliser mathématiquement un grand nombre de problèmes, tout particulièrement quand on ne s'intéresse qu'aux solutions entières, ce qui est l'objet de la *programmation linéaire en nombres entiers* (ou ILP) décrite dans la section suivante.

Tout d'abord, notons que l'on peut se restreindre sans perte de généralité aux problèmes de maximisation : si l'optimisation recherchée est une minimisation, il suffit de considérer l'opposé de la fonction objective. Ainsi, $LP \equiv \max\{c^T x \mid x \in \mathcal{C}\},\ c \in \mathbb{R}^n,\ \mathcal{C} \subseteq \mathbb{R}^n$. De même, il est aussi équivalent, d'un point de vue mathématique de ne chercher qu'à décider si un convexe est vide ou pas. En effet, $\max\{c^T x \mid x \in \mathcal{C}\} = \min\{u | \{x \in \mathcal{C}, c^T x \leq u\}$ non vide $\}$. D'un point de vue algorithmique par contre, ces deux problèmes ne sont équivalents, au sens de la réduction polynomiale, que si une dichotomie sur les valeurs de u est réalisable en temps polynomial. Des exemples simples de ce genre de problèmes dans $\mathbb{R}^2$ sont les exercices étudiés au lycée où il faut faire glisser une droite pour trouver le maximum de la fonction linéaire correspondante sur une zone délimitée par des droites (*cf* figure 1.1).

Maximiser $3x + 4y$

sous contrainte que :

$$\begin{aligned} x &\geq 1 \\ 3x + 4y &\geq 12 \\ 2x - 7y &\geq 0 \\ 6x + 7y &\leq 42 \end{aligned}$$

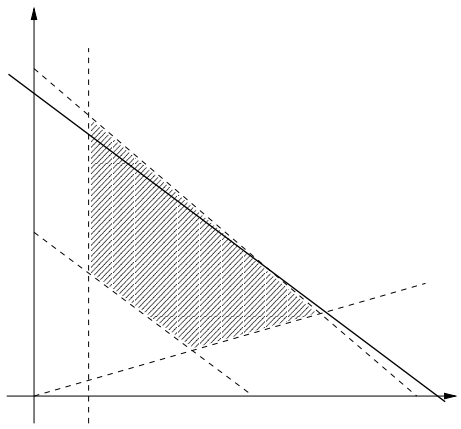


FIG. 1.1: *Exemple de programme linéaire dans* $\mathbb{R}^2$.

Dans cet exemple, le domaine convexe d'optimisation est un polygone. Étant donné que les frontières de ce domaine sont des droites, il aurait aussi pu bien s'agir d'une bande infinie ou d'un cône, et cela se généralise en dimension quelconque. Toutefois, le plus souvent les polyèdres considérés sont bornés, ne serait-ce que parce que les variables que nous manipulons dans un algorithme le sont en pratique. Dans ce cas, des arguments de compacité permettent de prouver que le polyèdre est l'enveloppe convexe d'un nombre fini de points, dits *points extrémaux* ou *sommets*. Ces sommets sont les points du polyèdre qui ne sont pas combinaison linéaire d'autres points, et un résultat fondamental de programmation linéaire est que l'optimum de ce programme est atteint en au moins un de ces points. Un tel polyèdre s'appelle un *polytope*.

1.1.1 Formulation matricielle

Un très grand nombre de problèmes se modélisent par la satisfaction d'un nombre fini de contraintes linéaires. Un programme linéaire à n variables et m contraintes peut s'écrire sous forme matricielle : $LP \equiv \max\{c^T x |\ Ax \leq b,\ x \in \mathbb{R}^n\}, A \in \mathbb{R}^{m \times n},\ b \in \mathbb{R}^n$, où x est le vecteur des n variables du programme, chaque ligne de la matrice A contient les n coefficients de l'équation directrice d'un des m hyperplans définissant les contraintes du programme, et b est le vecteur des m constantes affines de ces hyperplans.

Sous ce formalisme, l'exemple de la figure 1.1 s'écrit

$$\max \left\{ \begin{pmatrix} 3 \\ 4 \end{pmatrix}^T \begin{pmatrix} x \\ y \end{pmatrix} \;\middle|\; \begin{pmatrix} -1 & 0 \\ -3 & -4 \\ -2 & 7 \\ 6 & 7 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} -1 \\ -12 \\ 0 \\ 42 \end{pmatrix} \right\}.$$

Bien que cette écriture matricielle soit très agréable mathématiquement lors de raisonnements généraux, et ouvre la porte à l'utilisation de toute la puissance de l'algèbre linéaire, nous préférerons souvent l'écriture sous forme d'inégalités et égalités sur des variables, bien plus lisible lorsque variables et contraintes ont un sens combinatoire.

1.1.2 Séparation

Un problème fondamental relié aux ensembles convexes est celui de leur *description*. En effet, les polytopes que l'on manipule sont donnés par l'ensemble des hyperplans les délimitant, mais cet ensemble peut être arbitrairement grand, notamment exponentiel en la taille du problème que ce programme est censé résoudre. Un bon exemple d'une telle situation est la formulation *arc-chemin* du problème de multiflot donnée plus loin en section 1.5.2. Le polytope pourrait aussi bien être donné implicitement par un *oracle* qui décide si un point est dans le polytope ou non.

Étant donné un convexe fermé (et en particulier un polytope) et un point en dehors de cet ensemble, le *problème de la séparation* consiste à trouver un hyperplan qui sépare le point et l'ensemble, c'est-à-dire tel que le point soit dans un des 2 demi-espaces affines délimités par l'hyperplan tandis que le convexe est dans l'autre. Intuitivement, le point hors de l'ensemble ne satisfait pas à toutes les contraintes, et l'hyperplan séparateur est une contrainte violée par cette solution. Le problème de la séparation est donc un oracle de description implicite du convexe. Une propriété classique des ensembles convexes fermés est qu'un tel hyperplan séparateur existe toujours.

Théorème 1 *Soit $\mathcal{C}$ un ensemble convexe fermé et $z \notin \mathcal{C}$. Il existe un hyperplan séparant z et $\mathcal{C}$.*

La preuve de ce théorème se base sur la compacité de $\mathcal{C}$ (Schrijver 2000). L'aspect fondamental de ce résultat, pour nous qui utilisons la programmation linéaire comme un outil de modélisation et d'optimisation, s'illustre particulièrement dans le théorème **Sep=Opt** (Grötschel et coll. 1981) rappelé ci-dessous. Ce théorème fait le lien entre les complexités des problèmes de séparation et d'optimisation. La preuve utilise l'algorithme de l'ellipsoïde présenté plus loin, pour montrer principalement que l'oracle de séparation permet de décider en un nombre polynomial d'étapes si le convexe est vide ou non.

Théorème 2 (Sep=Opt) *Soit $\mathcal{K}$ une classe d'ensembles convexes. Il existe un algorithme polynomial pour résoudre le problème de séparation sur $\mathcal{K}$ si et seulement s'il existe un algorithme polynomial pour résoudre le problème d'optimisation sur $\mathcal{K}$.*

De manière plus générale, on pourrait s'intéresser aux convexes *bien définis*, c'est à dire ceux pour lesquels il existe un oracle de séparation polynomial, pour lesquels le théorème 2 s'applique directement. Cette classe contient évidemment les programmes linéaires avec un nombre polynomial de contraintes, puisqu'il suffit de les vérifier toutes pour séparer. Un autre exemple de convexe bien défini est une sphère. Une sphère est l'intersection d'un nombre infini de demi-espaces, mais un oracle de séparation consiste à comparer la distance d'un point au centre de la sphère avec son rayon, l'hyperplan séparateur étant, le cas échéant, l'hyperplan tangent à la sphère perpendiculaire à l'axe entre le point et le centre. Le théorème précédent implique donc que l'optimisation sur une sphère peut se faire en temps polynomial.

Dans les cas qui nous préoccupent, à savoir que nous avons toujours un nombre fini de contraintes, la difficulté vient de leur nombre. Supposons par exemple qu'un problème d'origine combinatoire se modélise par l'optimisation d'un programme linéaire nécessitant un nombre de contraintes exponentiel en la taille du problème, les exemples sont pléthore. Le temps de résolution d'un tel programme est, *a priori*, exponentiel lui aussi. Cependant, le théorème 2 certifie que si, et seulement si, le polyèdre est bien défini, on est capable de trouver en temps polynomial une contrainte violée par un point infaisable, alors il est possible d'optimiser le programme linéaire en temps polynomial. Ce résultat est particulièrement intéressant dans les cas où le problème de séparation a une interprétation exploitable en termes combinatoires. Cela permet d'obtenir aisément des preuves de polynomialité.

1.1.3 Dualité

La dualité est une notion fondamentale de la programmation linéaire.

Si l'on se donne un programme linéaire $\max\{c^T x |\ Ax \leq b,\ x \in \mathbb{R}^n\}$, $A \in \mathbb{R}^{m \times n}$, il est facile de trouver des bornes inférieures à la valeur optimale de la fonction de coût : il suffit de trouver une solution valide. Trouver des bornes supérieures demande plus de travail. Pour cela on construit le programme linéaire *dual*, $\min\{b^T y |\ A^T y \geq c,\ y \in \mathbb{R}^{+m}\}$. Le programme initial est alors appelé le *primal* et il est clair, vu l'écriture matricielle, que le dual du dual est le primal.

Sans rentrer dans des détails sans objet pour cette thèse, signalons que le dual est obtenu en cherchant des combinaisons linéaires des contraintes du primal, dont les coefficients sont les y_i, qui permettent de construire des bornes supérieures aux valeurs que peut prendre la fonction objective du primal. En effet, si l'on se donne une solution valide x du primal et une solution valide y du dual, on a l'inégalité suivante.

$$b^T y = y^T b \geq y^T (Ax) = \left(y^T A\right) x = \left(A^T y\right)^T x \geq c^T x.$$

Le théorème suivant énonce que cette inégalité entre les valeurs objectives des solutions valides est une égalité pour les optima des deux programmes linéaires.

Théorème 3 (Théorème de dualité) *Si le programme linéaire primal a une solution optimale x^*, alors le dual a une solution optimale y^* et les fonctions de coût coïncident : $c^T x^* = b^T y^*$.*

On trouve différentes versions de la preuve de ce théorème. Notamment Chvátal (1983) en propose une fondée sur le fonctionnement de *l'algorithme du simplexe* décrit plus loin. Schrijver (2000) en donne une autre qui utilise le lemme de Farkas (1896), qui caractérise l'existence de solutions positives ou nulles à un système d'équations linéaires $Ax = b$.

L'intérêt principal du théorème 3 est double. D'un point de vue algorithmique, il permet, dans les stratégies dites *primal/dual*, d'utiliser les variables du dual pour guider la construction d'une solution au primal et, éventuellement, d'en garantir la distance à l'optimum en garantissant l'écart entre la solution construite et une solution du dual. C'est le cas des algorithmes d'approximation du multiflot décrits en section 1.5.4 et au chapitre 5. Dans ce genre d'utilisations, une interprétation combinatoire du problème défini par le dual est essentielle pour développer un algorithme. Par contre, les programmes de résolution utilisent souvent le dual pour évaluer leur proximité à l'optimum.

D'un point de vue complexité, ce théorème montre l'équivalence entre l'optimisation du primal et celle du dual. Cela permet d'étudier un problème sous deux angles différents, voire de combiner ces deux approches. De plus, si le primal a n variables et m contraintes, le dual a lui m variables et n contraintes, or il est souvent plus avantageux

de manipuler un nombre restreint de variables tandis que le nombre de contraintes est moins problématique. Par ailleurs, ce résultat couplé au théorème 2 permet de montrer des résultats de complexité qui ne sont pas directs, par exemple lorsque le problème de séparation du dual a un sens combinatoire plus clair que celui du primal comme c'est le cas au chapitres 4 et 5.

1.1.4 Algorithmes de résolution

La formulation matricielle des programmes linéaires a beaucoup de similarités avec les systèmes d'équations linéaires usuels. Il n'est donc pas étonnant que des techniques de résolution de programmes linéaires soient très proches de l'algorithme du pivot de Gauss. Fourier avait d'ailleurs proposé une méthode d'élimination analogue à celle-ci, adaptée aux systèmes d'inégalités. Toutefois, cette méthode est pratiquement inexploitable car elle génère des équations auxiliaires en trop grand nombre, et la taille du système subit une véritable explosion. Les méthodes utilisées en pratique sont bien plus récentes.

L'algorithme du simplexe

L'algorithme du *simplexe*, proposé en 1947 bien que publié plus tardivement (Dantzig 1963), exploite le fait que les solutions optimales sont des points extrémaux du polytope, c'est-à-dire qu'ils sont localisés sur la surface du convexe des solutions réalisables et ne sont pas combinaison linéaire d'autres points du polytope. Fort de cette constatation, le simplexe suit des *règles de pivot* pour parcourir les points extrémaux, en ne détériorant jamais la fonction objective. La principale différence avec la méthode de Fourrier est que le simplexe utilise une caractérisation algébrique des points extrémaux : ce sont les points faisables ayant n ou plus contraintes serrées, où n est le nombre de variables du problème. Différentes règles de pivot existent donnant lieu à diverses variantes du simplexe.

Ces algorithmes ont la particularité d'avoir de très bonnes performances en pratique, bien qu'il existe des situations, dites dégénérées, où leur comportement devient catastrophique. En effet, études empiriques (Chvátal 1983) et analyses du temps moyen de calcul (Bertimas et Tsitsiklis 1997) montrent que le simplexe trouve une solution optimale en un nombre linéaire d'itérations, chaque itération prenant $O(mn)$ opérations sur un pro-

gramme à n variables et m contraintes. Cependant, Klee et Minty (1972) ont montré qu'il existe des familles de programmes linéaires pour lesquels le simplexe nécessite un nombre exponentiel d'itérations avec les règles classiques de pivot. Ce résultat est intuitivement dû au fait qu'un polytope de $\mathbb{R}^n$ délimité par m contraintes peut avoir de l'ordre de $\binom{n+m}{n}$ points extrémaux. L'algorithme de simplexe ne tire en fait pas avantage de l'extrême compacité de la représentation implicite de ces sommets, et il arrive que les règles de pivot forcent à suivre un chemin exponentiellement long avant de toucher un point extrémal optimal. Toutefois, il est conjecturé que la longueur du chemin le plus court entre deux points extrémaux est linéaire :

Conjecture 4 (Hirsch) *Le diamètre d'un polytope à n variables et m contraintes est au plus $m - n + 1$.*

L'algorithme de l'ellipsoïde

D'autres méthodes de résolution des programmes linéaires ont été développées dans le but d'obtenir des algorithmes polynomiaux.

La méthode dite de *l'ellipsoïde* a été mise au point dans le courant des années 70 par Shor (1970), puis Yudin et Nemirovskii (1977). Cette méthode s'applique à tous les convexes bien définis par le biais de l'oracle de séparation, et consiste en une sorte de généralisation à la dimension n de la recherche dichotomique, avec laquelle elle coïncide en dimension 1. Ici, l'intervalle coupé en deux est généralisé par un ellipsoïde englobant initialement tout le convexe puis réduit à chaque itération. Khachian (1979) a établi que cet algorithme décide si un programme linéaire est vide ou pas en temps $O(n^6 \log n)$. Grötschel, Lovász, et Schrijver (1993) ont ensuite montré le même résultat en prenant en compte les aspects de calculs en précision finie imposés par l'implémentation de l'algorithme. Ces performances sont meilleures que celles du simplexe d'un point de vue théorique puisqu'elles prouvent qu'il existe un algorithme polynomial de résolution de programme linéaire. D'un point de vue pratique par contre, la méthode de l'ellipsoïde est très mauvaise et le simplexe, bien qu'exponentiel dans le pire des cas, est très souvent plus rapide.

Les algorithmes de point intérieur

Les méthodes de *point intérieur* prennent un parti opposé à celui du simplexe. Là où le simplexe parcourt l'espace des solutions de point extrémal en point extrémal, les méthodes de point intérieur génèrent une suite de solutions valides en restant à l'intérieur du polytope. Une première méthode a été introduite par Karmarkar (1984). Plus tard, Gonzaga (1989) en a donné une amélioration permettant de trouver une solution optimale en $O(n^3L)$ itérations où L est la *complexité binaire* du programme linéaire, c'est à dire le nombre de bits nécessaires à l'écriture de la matrice des contraintes. Dans ces deux algorithmes, la stratégie consiste à minimiser une fonction de potentiel non linéaire dont une partie tend vers $-\infty$ à mesure que l'on se rapproche d'une solution optimale, tandis qu'une autre partie tend vers $+\infty$ à mesure que l'on se rapproche du bord du polytope, la somme de ces deux partie valant 0 sur un point optimal. La minimisation de cette fonction amène à ne s'approcher du bord du polytope que si l'on se rapproche suffisamment d'une solution optimale, évitant ainsi les errances du simplexe. D'autres améliorations ont été faites, utilisant différentes méthodes et fonctions de potentiel. En règle générale, les algorithmes de point intérieur pour la programmation linéaire sont plus efficaces que le simplexe sur de très gros programmes.

Il est intéressant de noter que les méthodes de point intérieur s'appliquent aussi pour des optimisations non linéaires et font toujours l'objet de nombreuses études de recherche opérationnelle comme (Armand et coll. 2001).

1.2 Programmation linéaire en nombres entiers

Comme il est rappelé dans l'introduction de ce chapitre, l'optimisation combinatoire concerne l'optimisation sur des structures finies. À la limite, il pourrait s'agir de structures infinies mais toujours discrètes. La programmation linéaire n'en fait donc pas partie à proprement parler puisque le domaine de recherche est continu, donc en dehors du champ d'application direct de l'informatique si l'on refuse des solutions approchées. Il existe toutefois des algorithmes de résolution de programme linéaire exacts qui restent dans des espaces discrets en se restreignant aux nombres rationnels ($\mathbb{Q}$). Ces algorithmes sont très intéressants car s'ils n'offrent généralement que de piètres performances opérationnelles,

ils permettent de travailler en précision infinie : en effet on peut montrer que l'optimum d'un programme linéaire à coefficients dans $\mathbb{Q}$ reste dans $\mathbb{Q}$ et que le simplexe ne manipule que des rationnels que l'on peut décrire avec un nombre polynomial de bits. De plus, $\mathbb{Q}$ étant dense dans $\mathbb{R}$, il n'y a pas une grande perte d'expressivité par rapport à la programmation linéaire "classique".

Cependant, dans la majeure partie des cas, la modélisation d'un problème en termes d'optimisation combinatoire réclame une solution qui soit à coordonnées entières. Bien sûr, notamment lors de modélisations de réseaux de télécommunication, certains problèmes peuvent permettre une interprétation directe de solutions à coordonnées réelles ou rationnelles. Par exemple, imaginons un réseau où des entités feraient des demandes de bande passante. Si le protocole sous-jacent permet de ne satisfaire qu'une fraction de chaque requête, utiliser des variables à valeurs dans $\mathbb{Q}^+$ est légitime, au moins en première approximation. Ce n'est pas le cas dans des situations moins flexibles où les variables du programme modélisent des quantités purement combinatoires.

Pour ce genre de cas, la programmation linéaire permet toujours une bonne description du problème, à ceci près qu'il faut contraindre les variables à être entières. Du fait de ces contraintes d'intégrité, les théorèmes donnés précédemment ne s'appliquent plus pour la plupart puisqu'ils sont souvent fondés sur des considérations de compacité de l'espace des solutions, ce qui nécessite la continuité des variables et la résolution des *programmes linéaires en nombres entiers* en devient extrêmement difficile.

Une question naturelle se pose ici : «*Pourquoi ne pas résoudre le programme linéaire sans les contraintes d'intégrité et arrondir la solution obtenue au point à coordonnées entières le plus proche ?*» La réponse est aussi simple que la question : ce point là n'est pas obligatoirement une solution valide comme le montre la figure 1.2.

Le programme linéaire où l'on omet les contraintes d'intégrité s'appelle la *relaxation linéaire* du problème initial. Il est malheureusement assez facile de se convaincre de la difficulté de chercher le point valide à coordonnées entières optimal. En effet, la programmation linéaire en nombres entiers est $\mathcal{NP}$-difficile en général et non-approximable puisque c'est le cas de nombreux problèmes d'optimisation combinatoire qui peuvent

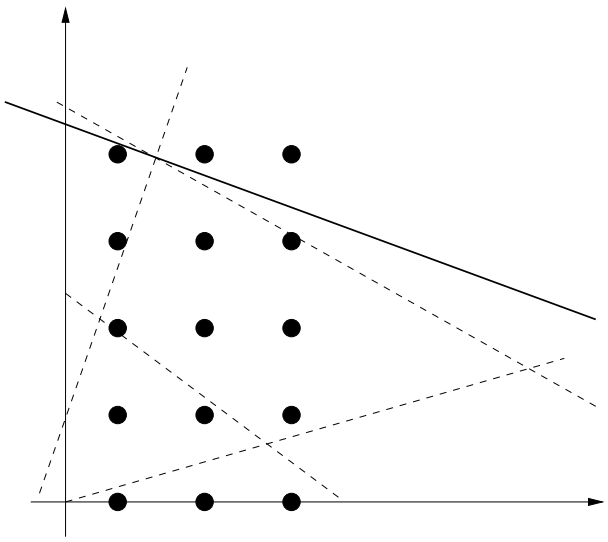


FIG. 1.2: *L'optimum de la relaxation linéaire est loin de l'optimum entier.*

s'écrire comme des programmes linéaires en nombres entiers avec un nombre de variables et de contraintes polynomiales en la taille du problème. La coloration de graphes est un exemple que nous verrons par la suite.

Les écritures en programmes linéaires en nombres entiers sont toutefois intéressantes dans certains cas. Il arrive que la relaxation linéaire donne des *solutions fractionnaires* qui permettent de construire de bonnes *approximations* des solutions entières. Dans tous les cas, l'optimal de la relaxation linéaire d'un problème de maximisation donne une borne supérieure à l'optimal entier puisque qu'une solution optimale entière est, en particulier, une solution valide de la relaxation linéaire. De même, l'optimal fractionnaire d'un problème de minimisation est une borne inférieure à l'optimal entier. Une conséquence de cette remarque et de la non-approximabilité de la programmation linéaire en nombres entiers est que l'écart entre l'optimal fractionnaire et l'optimal entier n'est pas borné en général. Nous en verrons une illustration avec une formulation élémentaire du problème de coloration de sommets d'un graphe dans la section 1.4

1.3 Algorithmique et approximation

Une bonne définition de l'*algorithmique d'approximation* serait la réponse à la question :

« `Que faire face à des problèmes` $\mathcal{NP}$`-complets ou` $\mathcal{NP}$`-difficiles ?` »

Cette question n'est pas anodine puisqu'une telle classification signifie que l'on ne connaît pas d'algorithme résolvant ces problèmes en temps polynomial. Souvent même, il n'existe pas d'autre solution que la méthode "*force brutale*" : tester un nombre exponentiel de candidats pour trouver l'optimal ou même une solution valide. Cela signifie que, dès que les entrées du problème grossissent, même très faiblement en proportion, il devient très vite impossible, à l'échelle humaine, de trouver une solution. Il n'est plus question alors d'échelle opérationnelle. Il faut donc faire son deuil de l'obtention de solutions exactes et tenter d'obtenir de "*bonnes*" solutions, les meilleures possibles, mais dans un temps raisonnable.

Une manière courante d'aborder cet objectif est le développement d'heuristiques. Le terme "heuristique" désigne, par abus de langage, les techniques et algorithmes construisant des solutions à un problème sans en garantir ni l'optimalité ni une mesure de la distance à l'optimal. La plupart des avancées récentes dans ce domaine concernent l'utilisation de paradigmes généraux principalement basés sur le concept de recherche locale. Ces approches sont souvent appelées "*métaheuristiques*" ; les plus connues d'entre elles sont le recuit simulé, la recherche tabou et les algorithmes génétiques (Reeves 2003). Les performances des métaheuristiques sont malheureusement souvent dépendantes d'un paramétrage très sensible à chaque instance. Malgré cela et l'absence totale de garantie de performance, ces techniques sont largement utilisées, particulièrement lorsqu'il s'agit de traiter des problèmes de très grandes taille et difficulté (Floriani 2001).

Les algorithmes d'approximation sont ceux dont la distance à l'optimal et le temps de calcul sont bornés.

1.3.1 Questions de "temps raisonnable"

La notion de "*bonnes* solutions" étant mieux définie, à savoir que leur distance à l'optimal est bornée, il reste à préciser le terme de "temps raisonnable". Plusieurs écoles

de pensée s'affrontent d'ailleurs à ce sujet.

"Raisonnable" pourrait vouloir dire, par exemple, "en moins d'une minute". Cette approche se justifie si l'on se place dans un contexte très figé tel qu'un contexte de production où les capacités de calcul et la taille des problèmes à traiter sont fixées. Le cas des systèmes embarqués en est un bon exemple pour lesquels les aspects "temps réels" ont en plus une grande importance. Cette définition perd par contre tout son sens si elle est confrontée à l'évolution quasiment exponentielle des capacités de calcul des ordinateurs et à une forte croissance de la taille des problèmes.

Un contexte prospectif où l'on est confronté à une évolution rapide de la taille des problèmes demande une vision à plus long terme. Dans ce cas, "raisonnable" peut signifier "qui se résout en temps polynomial". Cette notion, bien moins restrictive que la précédente, est elle aussi contestable. Une très belle illustration est donnée par l'actualité scientifique récente.

Agrawal, Kayal, et Saxena (2002) ont démontré que le test de primalité d'un entier est polynomial et ont donné un algorithme qui résout le problème en $O(n^{12})$ étapes où n est le nombre de bits nécessaires au codage de l'entier à tester . Le meilleur algorithme connu précédemment résolvait le problème en $O(n^{\log \log n})$, qui est bien super polynomial. Cela dit, les plus grands nombres premiers que l'on considère actuellement s'écrivent avec quelques dizaines de chiffres, ce qui veut dire un $(\log \log n)$ bien inférieur à 12 ($4,3$ pour des nombres à 20 chiffres). Un algorithme polynomial d'une telle complexité n'apporte donc aucune amélioration pratique immédiate (alors que d'un point de vue théorique, ce résultat est fondamental et peut être le premier pas vers un algorithme de bien plus faible complexité). La comparaison entre l'algorithme du simplexe et celui de l'ellipsoïde pour la résolution de programmes linéaires, exposé au paragraphe 1.1.4, est du même ordre. En général, l'algorithme de l'ellipsoïde n'est pas un choix pratique bien qu'il arrive à l'algorithme du simplexe de prendre un temps exponentiel.

Dans le cadre de nos recherches, le problème est double. D'un coté, il n'existe généralement pas d'algorithme de complexité "acceptable", ni dans l'absolu comme pour le test de primalité, ni en moyenne ou "dans la plupart des cas" comme pour le simplexe. En conséquence, un algorithme polynomial qui serait plus lent qu'un algorithme exponentiel

à un certain moment ne le restera pas longtemps à mesure que les tailles de problèmes deviennent plus importantes. Ce phénomène s'illustre parfaitement dans certaines expérimentations menées dans les chapitres suivants où l'on voit que sur de très petits cas une solution exacte se calcule plus vite qu'une solution approchée. D'un autre coté, la taille des problèmes envisagés est importante et croît rapidement. La complexité des algorithmes approchés est donc une question primordiale pour qu'ils soient capables de traiter des instances réelles et nous nous efforcerons tant que possible de limiter celle-ci.

Notre approche algorithmique consiste donc, en général, à améliorer le compromis entre temps de calcul et qualité d'approximation pour les problèmes d'optimisation combinatoire que nous rencontrerons.

1.3.2 Formalisation

Il existe plusieurs types d'algorithmes d'approximation. Étant donné l'objet de nos travaux, nous nous limiterons au cas des algorithmes qui approximent un problème d'optimisation en produisant une solution valide dont la valeur objective ne s'éloigne pas trop de l'optimum.

Définition 1 *Étant donné un problème d'optimisation P et une fonction $\rho : \mathbb{N} \to \mathbb{R}$, une* ρ-approximation *de P est un algorithme polynomial A tel que :*

$$\forall\ I\ \in\ P,\ \frac{1}{\rho(|I|)}\ \leq\ \frac{A(I)}{OPT(I)}\ \leq\ \rho(|I|)$$

où $OPT(I)$ est le coût optimal de l'instance I du problème P.

ρ est appelé le facteur d'approximation *de A.*

Il est évident que, pour un problème donné, une seule des deux inégalités de la formule précédente n'a de sens, suivant qu'il s'agisse d'un problème de maximisation ou de minimisation. Le fait de noter de la même manière un algorithme qui ne donne jamais moins de la moitié de l'optimal et un autre qui ne donne jamais plus de deux fois l'optimal peut paraître surprenant, mais a un sens : à tout problème de minimisation correspond un problème de maximisation associé et réciproquement.

Suivant cette définition, le cas évidemment le plus favorable est un facteur d'approximation du genre $1 + o(1)$, c'est à dire qui tend vers 1 à mesure que la taille de l'instance

du problème croît. Il ne sera malheureusement pas possible de se trouver toujours dans ce type de situations. Pour un état de l'art complet sur ces notions, on pourra consulter le livre de Ausiello, Crescenzi, Gambosi, Kann, Marchetti-Spaccamela, et Protasi (1999) et le dernier chapitre de celui de Hochbaum (1997).

1.4 Coloration de graphes et de chemins

Cette section est consacrée au rappel des définitions classiques, mais centrales dans nos travaux, des problèmes de coloration de graphes et de chemins dans les graphes.

Les graphes sont des structures combinatoires qui ont permis de modéliser la plupart des réseaux usuels, et de donner une formalisation algorithmique à des problématiques fondamentales d'optimisation. Les problèmes équivalents de coloration de sommets ou de chemins d'un graphe font partie de ces questions algorithmiques fondamentales que l'on retrouve dans l'optimisation de nombreux réseaux.

1.4.1 Coloration de graphe

Le problème de décision de la coloration des sommets d'un graphe peut s'écrire

« `Étant donné un graphe` $\mathcal{G}$ `et un entier` w`, est-il possible de colorier les sommets de` $\mathcal{G}$ `avec moins de` w `couleurs de sorte que deux sommets adjacents aient des couleurs différentes ?` »

Autrement dit, la coloration consiste à trouver une affectation d'éléments d'un ensemble donné (les couleurs) aux sommets telle que deux sommets adjacents ne se voient pas affecter du même élément. Le nombre minimum de couleurs nécessaires à une coloration d'un graphe G est appelé le *nombre chromatique* de G et noté $\chi(G)$. Le problème de coloration de graphes est connu pour être $\mathcal{NP}$-difficile et parmi les plus difficiles à approximer : en effet, pour tout $\epsilon > 0$, il est $\mathcal{NP}$-difficile de décider si le nombre chromatique d'un graphe à n sommets est inférieur à une constante α quelconque ou s'il est supérieur à $\alpha n^{1-\epsilon}$ (Hochbaum 1997; Bellare et coll. 1998).

Le problème d'optimisation consiste à minimiser la taille de l'ensemble des couleurs. Vu sous cet angle, ce problème peut s'écrire comme le programme linéaire en nombres

entiers 1.

Programme linéaire 1 (coloration de graphe)

$$
\begin{array}{lrcll}
\textit{Minimiser} & \sum_{i=1}^{w} y_i & & \textit{le nombre de couleurs utilisées} \\
\textit{t.q.} & \sum_{i=1}^{w} x_{i,v} & = & 1 & \forall\ v\ \textit{un sommet} \\
& x_{i,u} + x_{i,v} & \leq & 1 & \forall\ i \in [1 \ldots w],\ \forall\ (u,v)\ \textit{un arc} \\
& x_{i,v} & \leq & y_i & \forall\ i \in [1 \ldots w],\ \forall\ v\ \textit{un sommet} \\
& x_{i,v} & \in & \{0,1\} & \forall\ i \in [1 \ldots w],\ \forall\ v\ \textit{un sommet}
\end{array}
$$

Une telle écriture du problème n'est malheureusement pas efficace pour l'obtention de bonnes solutions, principalement parce que les variables sont de très bas niveau et ne prennent pas en compte la structure de l'affectation. En effet, la nature même du problème, le fait que deux sommets adjacents aient des couleurs différentes, est contenu dans l'alliance des contraintes "$x_{i,u} + x_{i,v} \leq 1$" et des valeurs binaires des variables. En perdant la binarité des variables, la relaxation linéaire du programme 1 n'appréhende plus le problème et l'affectation "$\forall\ u,\ x_{0,u} = x_{1,u} = \frac{1}{2},\ \forall\ i \geq 2\ x_{i,u} = 0$" sera toujours une solution triviale dont la valeur objective est $\sum_i y_i\ =\ 1$.

Autrement dit, cette formulation n'apporte pas grand chose au problème de coloration de graphe. Ce n'est pas le cas pour tous les problèmes car, même si la relaxation linéaire de l'écriture est très mauvaise, ce genre de formulation a au moins l'avantage de fixer les concepts manipulés et de poser le problème sans ambiguïté. En ce qui concerne la coloration de graphe, il est possible de faire mieux. La section suivante présente une écriture plus sophistiquée au sens où les variables prennent en compte une grande partie de la structure du problème, amenant à une relaxation linéaire beaucoup plus utile à la résolution du problème entier.

1.4.2 La coloration de sommets vue comme une couverture

Une meilleure approche de la coloration de graphes consiste à la considérer comme un problème de couverture. Rappelons ici que la couverture des sommets par une collection

d'ensembles couvrants est une partie de la collection telle que chaque sommet appartienne à au moins un des ensembles choisis. Dans la suite nous expliquons que le problème de coloration revient à trouver une couverture de coût minimum des sommets par les stables du graphe.

Un ensemble de sommets d'un graphe qui pourraient être coloriés de la même couleur est, par définition, un ensemble qui ne contient aucune paire de sommets adjacents. C'est exactement la définition d'un stable du graphe. Si l'on se donne une coloration valide des sommets d'un graphe, on appelle *une couleur* un ensemble de sommets coloriés identiquement. Chaque couleur est donc un stable et chaque sommet est dans une et une seule de ces couleurs. Ainsi, une coloration valide des sommets est un ensemble de stables tel que tout sommet appartienne à un et un seul de ces stables : ces derniers forment une partition des sommets. Autrement dit, trouver une coloration optimale des sommets d'un graphe est équivalent à trouver une partition de coût minimal des sommets par des stables de coût unitaire.

Une partition reste encore trop contraignante à calculer, mais on peut aller plus loin en utilisant une propriété de stabilité par suppression des stables, dont ils tirent leur nom : si l'on retire un sommet d'un stable, on obtient encore un stable. Ainsi, une couverture des sommets par des stables qui n'est pas une partition (c'est à dire qu'au moins un sommet appartient à plus d'un ensemble choisi) induit une partition de coût inférieur ou égal à celui de la couverture. Il suffit en effet de retirer un sommet qui serait sur-couvert de tous les ensembles auquel il appartient sauf un, choisi arbitrairement. En conséquence, le problème de coloration des sommets d'un graphe est équivalent à trouver une couverture de coût minimal des sommets par des stables de coût unitaire.

Étant donné un graphe $\mathcal{G} = (V, E)$ et $\mathcal{I}$ l'ensemble des stables de $\mathcal{G}$, cette couverture s'écrit comme le programme linéaire en nombre entier 2.

Programme linéaire 2 (coloration en couverture)

$$\begin{array}{lll} \textit{Minimiser} & \sum_{I \in \mathcal{I}} x(I) & \textit{le nombre de couleurs utilisées} \\ t.q. & \sum_{I \in \mathcal{I}: v \in I} x(I) \geq 1 & \forall\ v\ \textit{un sommet} \\ & x(I) \in \{0,1\} & \forall\ I \in \mathcal{I} \end{array}$$

La principale difficulté pour la résolution de ce programme linéaire est due au fait que le nombre de variables est généralement exponentiel puisqu'il y en a une par stable (et s'il existe un stable I de taille s, alors il y a au moins 2^s éléments dans $\mathcal{I}$, un par sous-ensemble de I). Cela dit, nous verrons dans la suite que la relaxation linéaire du programme linéaire 2 permet parfois de construire de bonnes solutions entières. Le problème combinatoire correspondant, défini pour la première fois par Grötschel, Lovász, et Schrijver (1981), est appelé *la coloration fractionnaire* des sommets du graphe et s'écrit comme le programme linéaire 3. Le coût d'une solution optimale s'appelle le *nombre chromatique fractionnaire*, noté $\chi_f(\mathcal{G})$ dans la suite.

Programme linéaire 3 (coloration fractionnaire)

$$\begin{array}{lll} \textit{Minimiser} & \sum_{I \in \mathcal{I}} \bar{x}(I) & \\ t.q. & \sum_{I \in \mathcal{I}: v \in I} \bar{x}(I) \geq 1 & \forall\ v\ \textit{un sommet} \\ & 0 \leq \bar{x}(I) \leq 1 & \forall\ I \in \mathcal{I} \end{array}$$

Il s'agit donc de calculer une pondération fractionnaire de coût minimal des stables obéissant aux contraintes de couverture. Si $\bar{x}$ est une pondération valide des stables de $\mathcal{G}$, nous l'appelons une *coloration fractionnaire* de $\mathcal{G}$ et notons $\bar{x}(\mathcal{G})$ le coût d'une telle solution.

En général le nombre chromatique fractionnaire est aussi difficile à approximer que le nombre chromatique entier : le théorème de Lovász (1975) sur les couvertures fractionnaires montre que toute ρ-approximation du nombre chromatique fractionnaire permet

d'obtenir une $\rho.\log(n)$-approximation du nombre chromatique entier. Or celui-ci n'est pas approximable à moins de $n^{1-\epsilon}$ dans le cas général comme nous l'avons dit plus haut.

Toutefois des arguments de dualité permettent d'obtenir des résultats exploitables tels que le lemme 5 qui nous permettra de trouver des classes de graphes sur lesquels la coloration fractionnaire peut se résoudre en temps polynomial.

Lemme 5 *Le calcul du nombre chromatique fractionnaire est équivalent au sens de la réduction polynomiale au calcul du* stable de poids maximum.

Preuve : Le dual d'un programme linéaire de couverture est le programme linéaire de *packing* où il y a une variable par contrainte du primal et une contrainte par variable du primal. Dans le cas qui nous intéresse, le dual de la coloration fractionnaire s'écrit comme le programme linéaire 4. Il s'agit dans ce problème de trouver une pondération la plus lourde possible des sommets, sous contrainte que le poids d'un stable, défini comme la somme des poids des sommets qu'il contient, soit inférieur à 1.

Programme linéaire 4 (dual de la coloration fractionnaire)

$$\begin{array}{lrcll} \textit{Maximiser} & \displaystyle\sum_{v\in V} y(v) & & & \\ t.q. & \displaystyle\sum_{v\in I} y(v) & \leq & 1 & \forall\ I\ \in\ \mathcal{I} \\ & y(v) & \geq & 0 & \forall\ v\ \textit{un sommet} \end{array}$$

Dans ce problème dual, une contrainte non triviale est violée si et seulement si il existe un stable I dont le poids est plus grand que 1. S'il en existe un, alors c'est le cas du stable de poids maximum, noté SPM, puisque son poids est au moins aussi grand que celui de I. De même, si le poids du SPM est inférieur à 1, alors le poids de tous les autres stables l'est aussi et aucune contrainte n'est violée. Le calcul du stable de poids maximum est donc un oracle de séparation pour le programme linéaire 4. Par conséquent, en appliquant le théorème 2 de séparation il vient que le programme linéaire 4 est équivalent au sens de la réduction polynomiale au calcul du stable de poids maximum. De plus, par le théorème 3 de dualité, le calcul du nombre chromatique fractionnaire et le calcul de l'optimum du

programme linéaire 4 sont aussi équivalents. Par transitivité de l'équivalence au sens de la réduction polynomiale, on obtient le lemme. □

L'inconvénient majeur de ce genre d'arguments de dualité et de séparation, c'est qu'ils ne donnent pas en pratique d'algorithme de calcul de la coloration fractionnaire, mais plutôt un résultat existentiel et éventuellement une méthode théorique de calcul du nombre chromatique fractionnaire dans de nombreux cas particuliers.

Il faut toutefois faire une remarque positive en observant le programme linéaire 4 : un grand nombre de contraintes sont trivialement inutiles et pourraient être directement éliminées. En effet, seules les contraintes relatives aux stables maximaux pour l'inclusion sont utiles : puisque le poids d'un stable est la somme des poids, positifs ou nuls, de ses sommets, le poids des stables est croissant par inclusion. Les sous-ensembles d'un stable maximal I induisent donc des hyperplans de coupe du polytope du programme linéaire qui sont parallèles à celui induit par I et moins restrictifs. Cela signifie que si un tel ensemble maximal est de taille s, $2^s - 1$ contraintes peuvent être éliminées. L'intuition qui vient alors est que le nombre de contraintes du programme linéaire pourrait ne pas poser de problème pour certaines classes de graphes. En revenant au primal, cela montre que l'on peut calculer une coloration fractionnaire en ne manipulant que des stables maximaux pour l'inclusion sans s'empêcher d'atteindre l'optimum. Cette remarque guide certains choix pour l'algorithme de coloration fractionnaire de chemins que nous présentons au chapitre 4. Cet algorithme est notamment fondé sur l'équivalence entre les problèmes de coloration de graphes et de chemins que nous présentons dans ce qui suit.

1.4.3 Coloration de chemins

Étant donné un graphe, le problème fondamental de coloration des sommets est d'affecter des couleurs aux sommets de sorte que deux sommets adjacents n'aient pas la même couleur et en utilisant un nombre minimum de couleurs différentes. Dans le cas général et dans un grand nombre de cas particuliers, ce problème est $\mathcal{NP}$-difficile et même non-approximable (Hochbaum 1997).

Le problème de coloration de chemins, lui, consiste en la coloration d'un ensemble $\mathcal{P}$ de chemins dans un graphe $\mathcal{G}$, de sorte que deux chemins partageant un arc de $\mathcal{G}$ n'aient

pas la même couleur et en minimisant le nombre de couleurs utilisées.

Ces deux problèmes ont été prouvés équivalents (Chlamtac et coll. 1992) en utilisant le *graphe des conflits* de $\mathcal{P}$ illustré en figure 1.3.

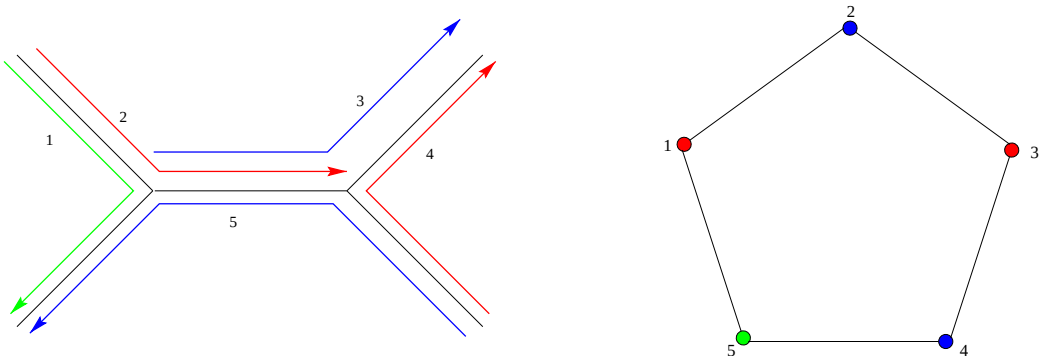


FIG. 1.3: *Le graphe des conflits d'un ensemble de chemins.*

Définition 2 *Le graphe des conflits d'un ensemble $\mathcal{P}$ de chemins dans un graphe $\mathcal{G}$ est le graphe qui possède un sommet par chemin dans $\mathcal{P}$ et un arc entre deux sommets si et seulement si les deux chemins correspondant sont* en conflit*, i.e. s'ils partagent un arc de $\mathcal{G}$.*

Si l'on a une coloration valide des sommets de graphe des conflits, on a une coloration des chemins. Par définition, si deux chemins partagent un arc du graphe $\mathcal{G}$, les sommets correspondants sont adjacents dans le graphe des conflits, donc ils ont des couleurs différentes. La coloration des chemins est donc valide et minimiser le nombre de couleurs de la coloration des sommets du graphe des conflits revient à minimiser celles de la coloration des chemins. Cela nous donne donc un sens de l'équivalence entre les deux problèmes. L'autre sens, prouvé par Chlamtac, Ganz, et Karmi (1992), est due au fait que tout graphe à n sommets est le graphe des conflits d'un ensemble de n chemins dans la grille $n \times n$.

Cette équivalence est extrêmement importante car tous les résultats de complexité sur la coloration de graphes s'importent directement pour la coloration de chemins, pour peu que l'on soit capable de caractériser précisément la relation entre une classe de graphes et une classe d'ensemble de chemins. Par exemple, la coloration de chemins a été prouvée

NP-difficile même dans le cas où le support des chemins est un anneau[1] (Tucker 1975), un arbre (Tarjan 1985), ou même une étoile non orientée (Golumbic et Jamison 1985).

Le chapitre 4 est consacré à l'étude de la transposition de la coloration fractionnaire présentée plus haut à la coloration de chemin. Les résultats sur les graphes se transposent aux chemins grâce au lemme 5 et nous donnons un algorithme efficace de calcul de la coloration fractionnaire sur des classes où elle peut se résoudre en temps polynomial.

1.5 Flot et multiflot

Le problème du multiflot est utile dans un grand nombre d'applications, notamment lorsqu'il est question de calculer des routes pour des entités qui sont en concurrence pour certaines ressources, ressources qui sont modélisées par des capacités sur les arcs du graphe support des routages possibles.

De manière formelle, un *réseau de flot* est un graphe $G = (V, E)$ avec des capacités $c(e)$ sur ses arcs, les arcs seront les supports du flot, les sommets seront des points intermédiaires ou bien des sources ou destinations du flot.

Un *flot* f de s à t est une pondération des arcs de G respectant des contraintes de *conservation de flot*, c'est à dire que la somme du flot entrant en un sommet est égale à celle sortant (Ford et Fulkerson 1962). Par analogie avec les théories physiques d'électrocinétique et de mécanique des fluides, ces contraintes s'appellent aussi *lois de Kirchoff*. L'*intensité* du flot est la quantité de flot injectée à la source s, qui, du fait de la conservation du flot, est égale à celle extraite à la destination t.

Un multiflot consiste à faire cohabiter plusieurs flots sur le réseau de sorte que la somme des flots passant sur un arc soit inférieure à sa capacité.

1.5.1 Programme linéaire *sommet-arc* : taille polynomiale

On se donne un multi-ensemble de *commodités* $C = \{(s_i, t_i),\ s_i, t_i \in V,\ i = 1, \ldots, k\}$. Un *multiflot* de C est une réunion de flots pour chaque commodité respectant les capacités de G. Il s'agit donc d'un ensemble de pondérations de E, respectant chacune les contraintes de conservation de flot et de sorte que $\forall e \in G,\ \sum_i f_i(e) \leq c(e)$.

[1] On parle alors de coloration des *graphes arc circulaires.*

Dans ce contexte, le problème du *multiflot maximum* cherche à maximiser la somme des intensités de chaque commodité qu'il est possible de router dans G en respectant les capacités. Ce problème s'écrit comme le programme linéaire 5.

Programme linéaire 5 (multiflot maximum)

$$
\begin{aligned}
\textit{Maximiser} \quad & \sum_i d_i & & & \\
\textit{t.q.} \quad & \sum_i f_i(e) & \leq & c(e) & \forall\ e\ \textit{un arc} \\
\sum_{e\in\Gamma^+(u)} f_i(e) - & \sum_{e\in\Gamma^-(u)} f_i(e) & = & 0 & \forall\ i,\ \forall\ u \neq s_i, t_i \\
\sum_{e\in\Gamma^+(s_i)} f_i(e) - & \sum_{e\in\Gamma^-(s_i)} f_i(e) & = & d_i & \forall\ i \\
\sum_{e\in\Gamma^+(t_i)} f_i(e) - & \sum_{e\in\Gamma^-(t_i)} f_i(e) & = & -d_i & \forall i \\
& f_i(e) & \geq & 0 & \forall\ i,\ \forall\ e
\end{aligned}
$$

La formulation en programme linéaire du multiflot ci-dessus est dite *sommet-arc*, en rapport aux variables et contraintes du programme. Cette écriture du problème utilise un nombre polynomial de variables et de contraintes, ce qui permet de prouver que le problème en nombres réels est polynomial et de le traiter par les algorithmes classiques de programmation linéaire. Cette formulation utilise notamment une variable par arc et par commodité qui indique la quantité de flot de la commodité qui passe sur l'arc.

Plusieurs variantes du problème de multiflot existent. Si l'on associe à chaque commodité i une *demande* d_i, le problème du *multiflot concurrent maximum* cherche à trouver le z maximum tel qu'il soit possible de faire passer un flot d'intensité $z.d_i$ pour chaque commodité tout en respectant les capacités. Il est aussi intéressant de chercher à minimiser la capacité maximale d'un arc ou la somme des capacités de sorte qu'il soit possible de router exactement d_i unités de flot pour chaque commodité i.

1.5.2 Programme linéaire *arc-chemin* : dual combinatoire

Les contraintes de conservation de flot de la formulation *sommet-arc* assurent le fait qu'une unité de flot suit un trajet qui peut être décrit par un chemin. En fait, on peut

considérer le flot d'une commodité comme un ensemble de chemins transportant chacun une certaine quantité de flot. Cette idée permet d'écrire le flot et le multiflot comme des problèmes de *packing* de chemins contraints par les capacités des arcs du réseau de flot.

Soit $G = (V, E), c$ un réseau de flot, $\{(s_i, t_i)\}$ l'ensemble des commodités, $\mathcal{P}_i$ l'ensemble des chemins de s_i à t_i dans G, et $\mathcal{P} = \cup_i \mathcal{P}_i$. Un flot de s_i à t_i s'exprime comme une pondération f des chemins de $\mathcal{P}_i$ et le flot de la commodité i est $\sum_{p \in \mathcal{P}_i} f(p)$.

Le fait que le flot soit porté par des chemins contient intrinsèquement la notion de conservation. Il reste à contraindre le flot au respect des capacités et, pour cela, il suffit de remarquer que la quantité de flot passant sur l'arc e est la somme des quantités de flot passant sur tous les chemins contenant e : $\sum_{p \in \mathcal{P}, e \in p} f(p) \ \leq \ c(e)$.

Le programme linéaire du multiflot maximum devient le suivant, logiquement qualifié d'*arc-chemin*.

Programme linéaire 6 (multiflot *arc chemin*)

$$
\begin{array}{lll}
\textit{Maximiser} & \displaystyle\sum_{p \in \mathcal{P}} f(p) & \\
t.q. & \displaystyle\sum_{p \in \mathcal{P}, e \in p} f(p) \ \leq \ c(e) & \forall \ e \ \textit{un arc} \\
& f(p) \ \geq \ 0 & \forall \ p \ \in \ \mathcal{P}
\end{array}
$$

Le problème principal de cette formulation est que le nombre de variables, c'est-à-dire le nombre de chemins, est généralement exponentiel en la taille du graphe. Le dual de ce programme linéaire a toutefois un grand intérêt. Si la variable duale associée à la contrainte de capacité de l'arc e est notée $l(e)$, le dual du programme linéaire 6 s'écrit :

Programme linéaire 7 (dual du multiflot *arc chemin*)

$$
\begin{array}{lll}
\textit{Minimiser} & \displaystyle\sum_{e} l(e)c(e) & \\
t.q. & \displaystyle\sum_{e \in p} l(e) \ \geq \ 1 & \forall \ p \ \in \ \mathcal{P} \\
& l(e) \ \geq \ 0 & \forall \ e
\end{array}
$$

Si l'on considère l comme une métrique sur les arcs, le produit $l(e)c(e)$ est une sorte de volume de l'arc e et le programme linéaire 7 peut s'exprimer combinatoirement comme

« `Minimiser du volume total du réseau de flot`
`sous contrainte que tout chemin soit de longueur supérieure à` 1. »

Le primal ayant un nombre exponentiel de contraintes, le dual a bien sûr un nombre exponentiel de variables, cependant son problème de séparation est trivialement polynomial.

Proposition 6 *Le problème de séparation du programme linéaire 7 est le calcul d'un plus court chemin par commodité.*

Preuve : L'ensemble $\mathcal{P}$ est la réunion des ensembles $\mathcal{P}_i$ des chemins de s_i à t_i, il suffit donc de vérifier que, pour tout i, l'ensemble des chemins de $\mathcal{P}_i$ est de longueur supérieure à 1. De plus, si le plus court chemin de s_i à t_i est de longueur supérieure à 1, alors tous le sont. Dans le cas contraire, la séparation est faite.

Il suffit donc de calculer autant de plus courts chemins que de commodités, ce qui est assurément polynomial. □

Outre la preuve, via le théorème 2 de séparation que la formulation *arc-chemin* du multiflot peut se traiter en temps polynomial, cette proposition montre que le multiflot est fortement relié aux calculs de plus courts chemins. Cette interprétation combinatoire et la théorie de la dualité sont à l'origine d'algorithmes combinatoires d'approximation du multiflot fractionnaire que nous décrivons en section 1.5.4.

Dans tous les cas, si les variables de flot sont réelles, les problèmes de multiflot décrits ci-dessus se résolvent en temps polynomial par la programmation linéaire. Si ces variables sont contraintes à être entières, les problèmes deviennent $\mathcal{NP}$-difficile et la recherche d'heuristiques et d'algorithmes d'approximation efficaces mobilise une grande partie de la communauté scientifique. Un type de techniques très répandu consiste en des heuristiques ou des métaheuristiques manipulant un nombre restreint de chemins dans la formulation *arc-chemin* pour être plus efficaces (Hyytiä et Virtamo 1999; Krishnaswamy et Sivarajan 2001; Kuri et coll. 2002).

Nous présentons dans la suite une technique d'approximation devenue un standard pour la construction de bonnes solutions aux problèmes $\mathcal{NP}$-difficiles, l'algorithme d'arrondi aléatoire de Raghavan (1994) dont nous proposons une amélioration au chapitre 5.

1.5.3 Approximation du multiflot entier par arrondi aléatoire

Les problèmes de multiflots s'appliquent bien sûr à l'optimisation des réseaux de télécommunication mais aussi à bien d'autres situations telles que l'optimisation des réseaux de transport ou encore la conception de circuits VLSI. Ces nombreuses applications ont motivé et motivent encore un grand nombre d'études de recherche opérationnelle ou algorithmiques. En effet, ces problèmes font partie des problèmes $\mathcal{NP}$-durs les plus difficiles à résoudre.

Raghavan (1994) a donné un algorithme de construction de multiflot fondé sur un arrondi aléatoire *par chemin* de la relaxation linéaire du problème. On peut considérer dans un premier temps que cet algorithme utilise une solution du multiflot fractionnaire dans sa formulation *arc-chemin* même s'il utilise en fait les variables *sommet-arc* pour ne pas en manipuler en nombre potentiellement exponentiel. La technique utilisée pour éviter d'utiliser les variables *arc-chemin* est présentée en section 5.1.1.

L'arrondi aléatoire par chemin choisit, aléatoirement et indépendamment, un chemin par unité de flot. Chaque chemin est choisi avec probabilité la proportion qu'il porte du flot de la commodité correspondante. L'intérêt d'arrondir le flot par chemins est que les contraintes de conservation de flot sont forcément respectées, or ce sont celles qui définissent structurellement un flot. L'approximation est donc reportée sur les contraintes de capacités, mais la solution arrondie est un multiflot.

Le fait que les arrondis aléatoires soient indépendants permet une analyse des performances de l'algorithme relativement simple et que nous détaillons en section 5.1.1. C'est aussi sa principale faiblesse, et nous proposons dans cette même section une amélioration de cet algorithme qui introduit des dépendances entre chaque choix.

1.5.4 Trouver plus vite un multiflot fractionnaire

L'autre problème posé par ces techniques d'arrondi aléatoire est qu'elles nécessitent la résolution de la relaxation linéaire du problème. Bien que celle-ci se calcule en temps polynomial, comme montré plus haut, les temps de calcul deviennent prohibitifs à mesure que la taille des problèmes augmente. Par exemple, la complexité moyenne d'un simplexe sur le programme linéaire *sommet-arc* d'un multiflot de k commodités sur un réseau à n sommets et m arcs est de l'ordre de $O(k^3m^2n + k^2m^3)$ (Bertimas et Tsitsiklis 1997). La résolution rapide des problèmes de multiflot fractionnaire est donc de grande importance.

Une technique très répandue, notamment en recherche opérationnelle, consiste à utiliser la formulation *arc-chemin* avec des techniques visant à ne considérer qu'un nombre restreint de chemins. Les techniques de *génération de colonnes* appliquées au multiflot rentrent dans cette catégorie. L'idée principale est de commencer par résoudre un premier programme linéaire avec peu de chemins puis de chercher grâce aux variables du dual un chemin à ajouter à l'ensemble des chemins pris en compte pouvant améliorer la solution précédemment obtenue et de recommencer jusqu'à ce qu'on ne trouve plus de chemin améliorant. Cette technique accélère sensiblement l'obtention d'une solution en pratique si la procédure de génération de colonne convient bien à la structure du problème, mais il n'est pas possible de garantir que la convergence de ce processus ne sera pas exponentielle (Wilhelm 2001).

Une technique heuristique consiste aussi à borner la longueur des chemins pour en limiter le nombre (Baroni et coll. 1999) ou bien à employer des métaheuristiques génériques telles que les algorithmes génétiques, le recuit simulé ou la recherche tabou sur l'ensemble des chemins. Toutes ces techniques se comportent généralement bien en pratique si elles sont convenablement paramétrées mais n'offrent aucune garantie de performance, ni du point de vue de l'approximation opérée le cas échéant, ni du point de vue du temps de calcul. De plus, le problème qui consiste à trouver un nombre minimal de chemins supportant un multiflot est $\mathcal{NP}$-difficile (Duhamel et coll. 2003).

Une autre technique enfin consiste à ne plus considérer les capacités comme des contraintes mais à pénaliser fortement les dépassements dans la fonction objective. Cette relaxation lagrangienne s'associe généralement à une construction itérative d'une solution

minimisant ces pénalités. Un historique de l'utilisation de ces techniques pour les problèmes de couverture et de *packing*, dont le multiflot fractionnaire, se trouve dans (Young 2001).

Nous détaillons dans la suite une autre approche qui vise à obtenir des multiflots fractionnaires de qualité garantie sans passer par la résolution d'un programme linéaire.

1.5.5 Approximations combinatoires du multiflot fractionnaire

Le dual de la formulation *arc-chemin* du multiflot montre qu'il y a un lien étroit entre le problème de multiflot et le calcul de plus courts chemins dans des graphes sur lesquels une métrique est posée en relation avec le flot qui passe par chaque arête. Deux approches sensiblement différentes se distinguent alors tout en partageant une idée centrale introduite par Shahrokhi et Matula (1990) : la manipulation d'une métrique sur les arcs qui mesure et est exponentielle en la quantité de flot qui passe sur chaque arc.

La première approche, historiquement, est introduite par Leighton, Makedon, Plotkin, Stein, Tardos, et Tragoudas (1995) et généralisée aux problèmes de *packing* et de couverture fractionnaire par Plotkin, Shmoys, et Tardos (1995). Les techniques principales consistent en la résolution du problème sans les contraintes de capacité, ce qui s'apparente à une relaxation lagrangienne, associée au reroutage itératif du flot jusqu'à ce que les capacités soient respectées à un facteur $(1 + \epsilon)$ près. Dans ce cadre, la métrique sur les arcs permet principalement de mesurer à quel point une capacité est violée. Cette technique a récemment été adaptée aux problèmes de *packing* généralisés par Jansen et Zhang (2002).

La seconde approche, introduite par Young (1995) dans un cadre plus général, considère le problème dans l'autre sens en se basant sur une approche primal/dual. Il s'agit de construire une solution du dual, la métrique précédente dans le cas du multiflot, qui guide et évolue avec des modifications de faibles amplitudes des variables du primal. L'application de cette approche au multiflot a été faite par Garg et Konemann (1998). Dans ce cas, l'interprétation combinatoire du dual du multiflot dans son écriture *arc-chemin* amène un algorithme principalement fondé sur l'idée que le chemin violant le plus les contraintes du dual est celui passant par les arcs les moins chargés. La stratégie consiste

donc, à chaque itération, à déterminer ce chemin par un calcul de plus court chemin pour chaque commodité, puis à augmenter le flot sur celui-ci, et à maintenir la métrique de ses arcs. Plus tard, Fleischer (2000) a amélioré la complexité de cet algorithme en la rendant indépendante du nombre de commodité du multiflot. Ces deux algorithmes dont nous proposons une amélioration sont détaillés en section 5.2.

Il est intéressant de remarquer aussi que si ces deux approches semblent à première vue différentes, elles peuvent toutefois être considérées comme similaires. En effet, si l'on imagine que pour chaque commodité (s_i, t_i) il y a un arc supplémentaire de capacité et de longueur nulles sur lesquels tous les flots passent initialement, la seconde approche opère un reroutage de ces flots de même nature que ce que propose la première. La plus grande différence est alors que les arcs dont les contraintes sont les plus violées sont systématiquement les arcs factices. On peut arguer que cette façon d'opérer force à utiliser une situation de départ trop strictement définie et qui pourrait imposer des contraintes parasites fortes aux solutions ou bien que la seconde approche consiste à éviter de faire des choix coûteux pour être plus efficace. Toujours est-il que ces deux approches permettent d'obtenir des $(1+\epsilon)$-approximations et qu'elles ont de grandes similitudes.

Chapitre 2

Modèles et algorithmes pour les réseaux optiques d'infrastructure

La communauté scientifique a abordé les questions relatives au dimensionnement des réseaux optiques à multiplexage en longueurs d'onde (WDM pour *wavelength division multiplexing*) de différentes manières. Certains ont d'emblée considéré le problème dans toute sa complexité. Cette complexité, calculatoire mais aussi conceptuelle, est largement due au fait qu'il n'y a aucune raison pour que la multitude de facteurs en présence cadre avec un modèle informatique et s'adapte à des méthodes de calcul connues et efficaces. Le but était donc d'obtenir des solutions aux problèmes de dimensionnement, par quelque méthode que ce soit, et de tenter d'améliorer les méthodes les plus effectives.

Une autre stratégie consiste à ne pas forcer les outils disponibles à s'adapter au problème mais plutôt à tenter d'adapter le modèle, quitte à le simplifier quelque peu. L'intérêt de cette stratégie est qu'elle construit un pont entre le monde réel, où le nombre et l'interdépendance des paramètres brouillent toutes les pistes, et la sphère de la théorie. L'identification des problématiques fondamentales que l'on rencontre dans l'optimisation des réseaux optiques et leur étude dans un modèle épuré et proche des notions théoriques classiques permettent d'en avoir une compréhension plus profonde. Il devient ensuite possible d'enrichir le modèle et de le rendre de plus en plus réaliste en s'appuyant sur des bases théoriques solides.

2.1 Technologie des réseaux WDM

La modélisation des réseaux en vue du développement d'algorithmes de dimensionnement exige que l'on ait une bonne connaissance des solutions technologiques mises en œuvre. Dans cette section nous nous concentrons sur les spécificités des réseaux que nous considérons par la suite, les réseaux tout optiques d'infrastructure à multiplexage en longueurs d'onde. Dans un premier temps nous décrivons les types classiques de multiplexage utilisés dans la plupart des réseaux de télécommunications afin de justifier l'utilisation du multiplexage en longueurs d'onde. Après avoir schématisé le fonctionnement des routeurs WDM tout optiques, nous montrons l'origine technologique des principales problématiques d'optimisation qui seront détaillées dans la suite.

2.1.1 Les différents types de multiplexage

Le multiplexage est la technique qui consiste à partager la bande passante d'un média entre différents flux. Les réseaux optiques d'infrastructure actuels sont basés sur le multiplexage en longueurs d'onde bien qu'il soit, par certains aspects, bien moins flexible que d'autres techniques, comme le multiplexage en temps utilisé sur les réseaux SDH par exemple. Afin de justifier ce choix technologique, nous présentons les grandes familles de multiplexage et montrons que si le multiplexage en longueurs d'onde est bien la caractéristique principale des réseaux que nous étudions, il y a en fait un empilement de plusieurs multiplexages.

Le multiplexage en temps est incontestablement le plus connu et le plus naturel. Chaque flux est découpé en quantités unitaires comme les paquets des réseaux IP, les cellules des réseaux ATM, SDH ou GSM ou encore les trames ethernet. Le multiplexage en temps consiste alors à laisser chaque flux utiliser pleinement le média à son tour pour émettre une de ses unités de trafic. Le partage peut être fait selon deux paradigmes différents.

déterministe : le temps est divisé en trains périodiques de créneaux de durée fixée. Chaque créneau est affecté à un unique flux. C'est le cas des réseaux SDH, GSM, ...;

statistique : le temps n'est pas réparti entre les flux *a priori*. Des mécanismes de contrôle d'accès au média autorisent les différents flux à envoyer un paquet tour à tour. Ces mécanismes répartissent ainsi la bande passante entre les flux en s'adaptant à leurs comportements. Ce type de multiplexage apparaît dans les réseaux ethernet, IP et dans certains réseaux radio *ad-hoc* tels que ceux utilisant la norme 802.11, aussi appelée *WiFi*.

Le multiplexage en temps suppose que chaque flux est en capacité d'utiliser toute la bande passante offerte par le média au moment où il émet. Dans le cas contraire une partie de cette bande passante est gaspillée.

Le multiplexage en code est principalement utilisé dans certains réseaux radio. Là encore, les émetteurs doivent pouvoir utiliser pleinement le média. Le partage de la bande passante s'opère par le fait que chaque émetteur multiplie son signal par un code, orthogonal à ceux des autres émetteurs en concurrence avec lui. Ainsi le récepteur reçoit un signal complexe qu'il peut projeter sur le code correspondant au flux qu'il souhaite décoder. L'orthogonalité des codes lui assure de retrouver le signal émis. Le nombre de codes nécessaires joue directement sur la longueur qu'ils doivent avoir pour être orthogonaux et plus un code est long plus il faut de temps pour transmettre une quantité donnée de trafic multipliée par ce code, ce qui divise d'autant la bande passante effective disponible pour chaque flux.

Le multiplexage en fréquences divise un média en canaux de bande passante inférieure portés par des fréquences différentes. Chaque flux est émis sur un canal différent et le récepteur n'a qu'à écouter la fréquence adéquate pour récupérer le signal. Dans ce cas le média peut offrir une bande passante très largement supérieure à la capacité d'émission de chaque flux.

Le multiplexage en longueurs d'onde est un autre nom pour le multiplexage en fréquence quand le média est optique. Les fibres optiques des réseaux optiques WDM sont capables de porter simultanément plusieurs flux de données sous forme de faisceaux laser de longueurs d'onde différentes comme décrit en figure 2.1. En pratique les flux de

données portés par les connexions optiques sont souvent des agrégations par multiplexage en temps de flux générés par des émetteurs électroniques et donc de plus faible débit.

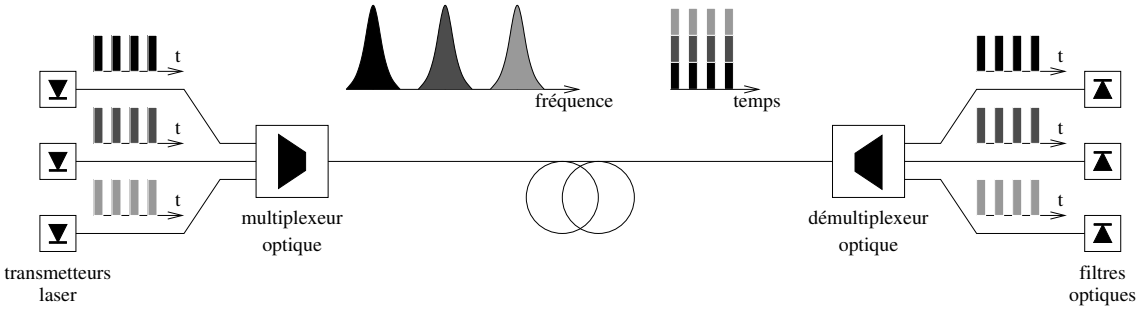
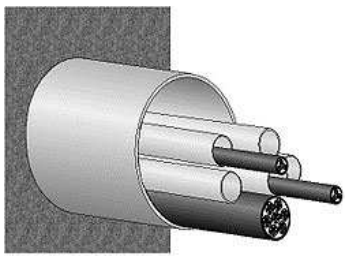


FIG. 2.1: *Multiplexage en longueurs d'onde (*WDM*).*

Le multiplexage en longueurs d'onde est nécessaire aux réseaux optiques d'infrastructure : le média optique offre une bande passante supérieure de plusieurs ordres de grandeur aux capacités d'émission des équipements électroniques actuels. Comme le montre la figure 2.2, les émetteurs à 160 Git/s sont encore à l'état de recherche alors que les systèmes optiques WDM peuvent théoriquement offrir jusqu'à 50 Tbit/s et des équipements offrant une bande passante de l'ordre du Tbit/s sont disponibles commercialement : Mitsubishi Electric propose une solution[1] pour exploiter 50 longueurs d'onde à 20Gbit/s et a, en laboratoire, un système[2] de 85 longueurs d'ondes à 20Gbit/s, les deux sur des distances transocéaniques.

Le multiplexage spatial consiste, lui, à augmenter la bande passante disponible en rassemblant plusieurs média identiques et isolés les uns des autres. C'est le cas par exemple des réseaux radio utilisant des systèmes multi-antennes (Foschini 1996) ou des faisceaux de fibres optiques, illustrés ci-contre, qui sont installés par les opérateurs pour les réseaux d'infrastructure.



[1]Source : `http://global.mitsubishielectric.com/news/sc_2001/ts_a_4_b.html`
[2]Source : `http://global.mitsubishielectric.com/bu/communication/keyt/keyt.html`

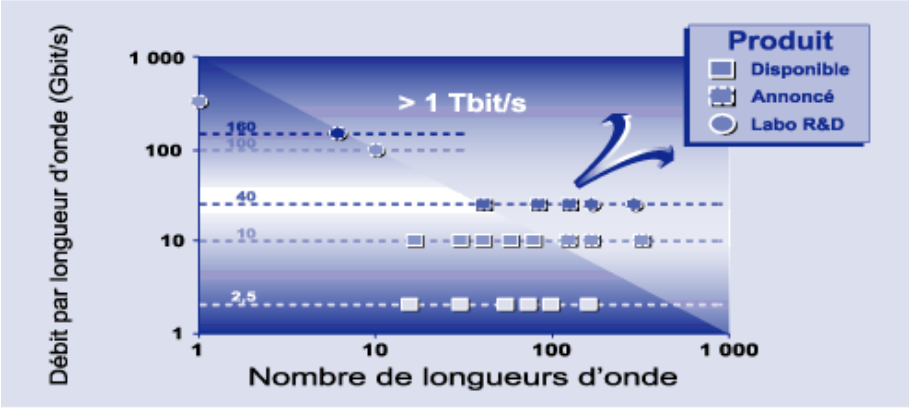


FIG. 2.2: *Disponibilité des équipements* WDM

2.1.2 Les routeurs WDM tout optiques

Afin d'organiser des communications dans un réseau, il faut permettre à un flux de données de passer de son origine à sa destination quand bien même il n'y aurait pas de câble entre ces deux nœuds, on parle alors de *communication à plusieurs sauts*. C'est le rôle d'un routeur.

Les routeurs WDM doivent donc traiter les longueurs d'onde entrant sur un port et les diriger vers leur port de sortie. Pour cela, il faut que le routeur puisse *ouvrir* les câbles pour accéder aux fibres, puis ouvrir les fibres pour accéder aux longueurs d'onde. Il faut donc un équipement par câble, puis un par fibre, puis enfin un par longueur d'onde, c'est-à-dire un étagement relativement complexe. Afin de limiter les coûts des routeurs, il peut être intéressant d'opérer des traitements de groupe : par exemple si toutes les longueurs d'onde d'une même fibre ont la même fibre de destination, il est inutile d'ouvrir la fibre pour distinguer chaque longueur d'onde. Cela permet d'éviter la présence d'un équipement de routage par fibre. Selon le même raisonnement, un autre niveau de groupage existe au sein des routeurs, la *bande* de longueurs d'onde. Si une plage complète de longueurs d'onde suit le même trajet, on les route toutes d'un seul mouvement. Le fonctionnement d'un routeur WDM est décrit par la figure 2.3 sur laquelle on distingue bien les différents niveaux de groupage : fibre, bande et longueur d'onde.

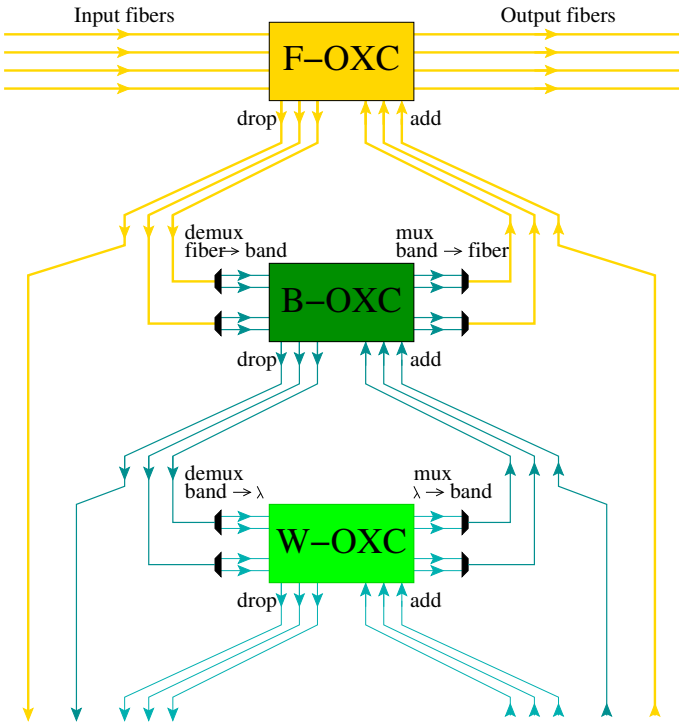


FIG. 2.3: *Organisation d'un routeur* WDM

Les routeurs tout optiques sont un type particulier de routeurs permettant de manipuler des flux à très haut débit. En effet, le débit de chaque longueur d'onde est limité par les capacités des systèmes électroniques. Si les équipements de routage (les OXC pour *optical cross-connect* sur la figure 2.3) sont des systèmes électroniques, les limitations de débit n'en deviennent que plus strictes. Les routeurs *tout optiques* évitent donc de passer par une étape électronique. Les signaux entrant sont démultiplexés par un système optique que l'on peut (abusivement) comparer à un prisme, les rayons laser arrivant des fibres en entrée sont déviés vers les fibres de sortie correspondantes à l'image de ce que feraient des lentilles et des miroirs correctement placés, les signaux sortant sont alors multiplexés sur les fibres de sortie.

Les routeurs tout optiques ne sont toujours pas réellement déployés en pratique mais devrait l'être dans les années qui viennent et sont d'ores et déjà disponibles commercialement (Ritzenthaler 2003).

La conversion en longueurs d'onde est un procédé technologique qui permet à un routeur de modifier la longueur d'onde d'un faisceau laser avant de le diriger sur la fibre appropriée. Cela permet d'autoriser deux faisceaux lasers qui arriveraient de deux fibres différentes avec la même longueur d'onde à repartir sur la même fibre avec deux longueurs d'onde différentes. Les premiers équipements de conversion en longueurs d'onde étaient opto-électroniques, une façon simple de faire de la conversion en longueurs d'onde consistant à scinder le faisceau en deux : le routeur réceptionne le faisceau, le convertit en signal électronique puis le ré-émet sur la nouvelle longueur d'onde. Les convertisseurs opto-électroniques introduisent par conséquent des temps de traitements électroniques indésirables. Des convertisseurs tout optiques ont donc été développés pour être intégrés aux routeurs optiques (Yu et coll. 2000).

Il existe plusieurs types de fonction de conversion dont les principales sont les suivantes.

- totale : tous les faisceaux entrants peuvent subir une conversion. Il n'y a de contrainte ni en nombre, ni en qualité sur les conversions. Ce type de fonction de conversion est mise en place quasiment systématiquement sur les routeurs faisant l'interface entre des réseaux gérés par deux opérateurs différents pour les "isoler" : d'une part, aucun des deux opérateurs n'a envie de donner à l'autre la moindre information sur sa façon d'affecter les longueurs d'onde aux connexions et, d'autre part, rien ne force les opérateurs à avoir les mêmes équipementiers et les longueurs d'onde utilisées sur un des réseaux à être disponibles sur l'autre (Subramaniam et coll. 1999).
- partielle : tous les faisceaux entrants sont éligibles à la conversion, mais seul un nombre restreint peut l'être. Les conversions ne sont pas limitées en qualité (Sharma et Varvarigos 2000; Auletta et coll. 2001).
- limitée : tous les faisceaux entrants peuvent subir une conversion, mais l'ensemble des longueurs d'onde accessibles à un faisceau dépend de sa longueur d'onde originale (Wang et coll. 2002).

2.1.3 Multiplexage en longueurs d'onde et mode connecté

La gestion des flux de données dans les réseaux s'est résolue historiquement par deux approches différentes. Le *routage par paquets*, très fortement relié au multiplexage temporel statistique, consiste à découper le flux en paquets et à router ces paquets de proche en proche. Ainsi la route que suit un paquet se décide à mesure que celui-ci la parcourt, c'est l'approche suivie dans les réseaux IP. La deuxième approche, appelée *commutation de circuits* consiste à établir une route au préalable puis à envoyer les données le long de ce chemin. Cette approche est celle des réseaux ATM et SDH fonctionnant sur du multiplexage temporel déterministe.

Dans le cadre des réseaux optiques, le routage par paquets pose de réels problèmes technologiques. En effet, cette technique nécessite que l'on puisse stocker un paquet dans un routeur pendant un temps non déterministe avant de l'envoyer vers la destination. C'est le modèle *store-and-forward*. Le problème est que les mémoires optiques qui permettraient de rester dans le monde des réseaux tout optiques n'existent pas : les seules mémoires optiques existant actuellement sont des enroulements de fibres qui ne peuvent retarder un flux lumineux que d'un temps déterministe, ce qui amènerait à des synchronisations *de facto* des routeurs qui seraient difficiles à gérer. De plus ces enroulements sont peu contraignants à manipuler et deviennent très volumineux à mesure que le débit des longueurs d'onde augmente. L'idée de router des paquets IP directement sur un réseau WDM tout optique semble donc difficile. D'ailleurs, les études qui sont faites sur les routeurs de paquets optiques (*Optical Packet Switch*) supposent que le multiplexage en temps est déterministe, c'est-à-dire qu'elles considèrent plutôt des réseaux de type SDH sur WDM (Eramo et Listanti 2003).

Pour autant, allier sans couche intermédiaire la souplesse des réseaux IP aux performances des réseaux WDM tout optiques est séduisante et un important effort de recherche est consacré à l'émergence des réseaux IP/WDM. Une des solutions retenue par la communauté cherche à éviter à la fois le temps d'établissement d'une route préalable à l'envoi et le stockage intermédiaire dans les routeurs. Le *routage optique de salves* (OBS pour *Optical Burst Switching*) est une demi-mesure entre la commutation de circuit et le routage par paquets qui consiste à retarder l'envoi des paquets jusqu'à ce qu'une fenêtre d'émission

soit libre puis envoyer une salve de paquets qui puisse traverser le réseau sans avoir jamais besoin d'être stockée en cours de route (Qiao et Yoo 1999; Xu et coll. 2001). Dans la suite nous ne considérerons pas ce type de routage pour lequel s'associent des problématiques protocolaires et d'établissement de route.

Les réseaux que nous étudions sont donc des réseaux WDM tout optiques fonctionnant en *mode connecté*. C'est-à-dire qu'à chaque demande de connexion il faut fournir un chemin dans le réseau. Ce fonctionnement correspond à un certain nombre de situations concrètes telles que la construction de topologies virtuelles dans les réseaux d'infrastructure (Choplin 2002). Dans les réseaux WDM tout optiques, une connexion n'est pas servie par un simple chemin du réseau, mais par un *chemin optique*, c'est-à-dire un chemin porté par des longueurs d'onde.

Le routage des requêtes est donc la tâche à accomplir. Il s'agit de trouver pour chaque requête un chemin dans le réseau reliant la source à la destination. Un des paramètres d'optimisation les plus considérés est *la charge* du routage. La charge d'une fibre du réseau est le nombre de chemins utilisant ce lien et la charge du routage est celle de la fibre la plus utilisée. Dans un contexte où la charge des liens est bornée, ou bien si l'on cherche à la minimiser, des problématiques de *multiflot* entrent en jeu.

L'affectation de longueurs d'onde consiste à choisir une longueur d'onde pour chacun des chemins optiques. Le multiplexage WDM impose que deux chemins passant par la même fibre aient des longueurs d'onde différentes. De plus, l'emploi des technologies tout optiques fait qu'en dehors de la présence de convertisseurs le faisceau laser portant une connexion aura la même longueur d'onde de bout en bout. Dans ces conditions, la contrainte de multiplexage en longueurs d'onde fait émerger des problématiques de coloration de chemins que nous développons en section 1.4.3. L'utilisation conjointe du multiplexage spatial et du multiplexage en longueurs d'onde modifie considérablement la nature des contraintes en jeu comme nous le verrons en section 2.3. L'optimisation principale de l'affectation en longueurs d'onde consiste à minimiser le nombre de longueurs d'onde nécessaire pour respecter les contraintes imposées par le multiplexage WDM. En particulier la charge du routage donne une borne inférieure à cette minimisation puisque

tous les chemins passant par la fibre de plus grande charge doivent avoir des longueurs d'onde différentes. Pour autant, les problématiques de coloration de chemins faisant plusieurs sauts peuvent amener à un écart incompressible entre la charge et le nombre optimal de longueurs d'onde d'un routage. On nomme *efficacité de l'affectation en longueurs d'onde* le ratio entre ces deux valeurs.

Le routage optique est le problème qui consiste à opérer simultanément le routage des requêtes et l'affectation de longueurs d'onde aux chemins, ce qui amène à de bien meilleures solutions que l'optimisation successive des deux sous-problèmes. En effet, il a été prouvé par Beauquier (2000) qu'il existe des familles de réseaux et de requêtes pour lesquelles l'écart entre le nombre optimal de longueurs d'onde pour un routage optique et le nombre de longueurs d'onde optimal pour le routage optimal au sens de la charge est non borné. Le routage optique met en jeu des problématiques de multiflot dans des graphes auxiliaires qui font l'objet du chapitre 5.

L'empilement de multiplexages est la réalité des réseaux actuels. La figure 2.4 décrit la situation classique d'un réseau transportant des paquets IP qui sont rassemblés sur une boucle locale SDH, transportée par un réseau WDM multifibre.

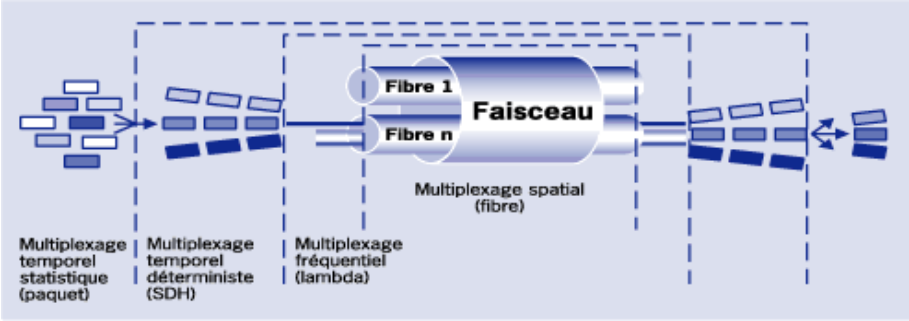


FIG. 2.4: *Empilement de multiplexages*

La coexistence de ces différents multiplexages sont à l'origine de problématiques de groupage de flux. D'une part on cherche à grouper les flux optiques qui suivent des portions communes de trajet afin de minimiser le coût des équipements de routage dans les routeurs montrés en figure 2.3. D'autre part, on cherche à grouper les flux électroniques à l'entrée du réseau qui partagent une même longueur d'onde. En effet, chaque fois qu'un

flux électronique doit être inséré ou extrait par un routeur, il faut qu'un équipement soit présent dans le routeur pour manipuler la longueur d'onde portant ce flux. Ces équipements, les ADM pour *Add-Drop Multiplexer*, sont coûteux et le groupage vise à minimiser leur nombre dans le réseau.

La conversion en longueurs d'onde permet de relâcher les contraintes de l'affectation de longueurs d'onde et, par conséquent, d'obtenir des affectations de bien meilleure efficacité (Ramamurthy et Mukherjee 1998; Subramaniam et coll. 1999; Auletta et coll. 2001; Sharma et Varvarigos 2000). En effet, du point de vue de l'affectation de longueurs d'onde, une conversion agit comme si le chemin converti était coupé en deux sous-chemins indépendants, découplant ainsi les contraintes de coloration sous-jacentes. Les optimisations concernant les équipements de conversion cherchent toutes à en minimiser le nombre tout en cherchant à obtenir, par exemple, une efficacité optimale, c'est-à-dire que le nombre de longueurs d'onde nécessaires soit égal à la charge du routage.

Dans le cas de la conversion totale, l'optimisation consiste à choisir un sous-ensemble minimal des nœuds du réseau où installer les convertisseurs. Puisqu'un convertisseur total permet de traiter tous les chemins y passant sans aucune contrainte, placer un convertisseur dans un nœud revient à couper le réseau et les chemins à cet endroit là. Il s'agit donc de trouver des décompositions des réseaux en sous réseaux sur lesquels il soit aisé d'obtenir une affectation de longueurs d'onde d'efficacité optimale (Auletta et coll. 2001; Subramaniam et coll. 1999; Togni 2000). Le cas de la conversion partielle est différent puisque des conversions non contraintes sont disponibles sur tous les sommets mais en nombre limité. Il s'agit alors plutôt de dimensionner chaque équipement pour assurer que tout chemin nécessitant une conversion en trouvera une disponible. Auletta, Caragiannis, Gargano, Kaklamanis, et Persiano (2001) ont retrouvé des problématiques de *graphes à expansion* dans le cas où le réseau est en arbre. Nous montrons au chapitre 5 qu'il y a un lien fort entre ce dimensionnement et l'optimisation des capacités d'un réseau de multiflot.

Dans le cas de la conversion limitée, les problématiques sont plus complexes et mêlent dimensionnement de la largeur du spectre accessible à une longueur d'onde et répartition de ces spectres dans l'ensemble des longueurs d'onde disponibles ;

La section suivante s'intéresse plus spécifiquement aux problématiques d'affectation de longueurs d'onde dans un modèle très épuré des réseaux optique WDM monofibres.

2.2 Les réseaux monofibres

Les *réseaux optiques monofibres* sont les premiers réseaux WDM à avoir été étudiés dans la littérature. Un premier modèle de ces réseaux tire parti d'une grande proximité entre les principales problématiques d'optimisation rencontrées et certaines notions classiques de théorie des graphes.

2.2.1 Les spécificités du modèle

L'interconnexion physique des nœuds d'un réseau optique monofibre est réalisée par des câbles composés de deux fibres optiques orientées de manière symétrique. La modélisation de ces fibres implique souvent une capacité potentiellement infinie en nombre de longueurs d'onde, ce qui est cohérent lorsqu'on cherche à minimiser ce nombre.

Les routeurs qui équipent les nœuds utilisent les technologies tout optiques. Les fonctionnalités en place dans les routeurs que nous considérons sont schématisées en figure 2.5 : il ne s'agit que d'établir des connexions à plusieurs sauts en les manipulant par longueur d'onde et sans conversion.

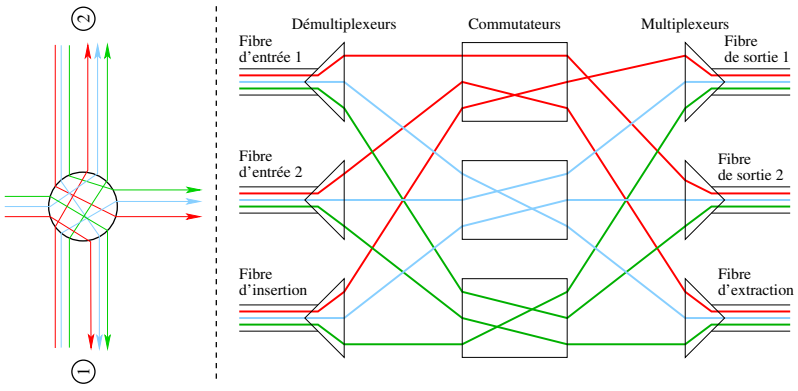


FIG. 2.5: *Schéma de fonctionnement d'un routeur dans un nœud à 2 voisins.*

L'instance de communication du réseau est l'ensemble des *requêtes de communication* que le réseau doit servir. Une telle requête est la donnée d'un nœud source et d'un nœud destination entre lesquels une communication doit être établie.

Une description combinatoire satisfaisante d'un tel réseau s'obtient directement avec un graphe orienté symétrique. Une demande de connexion est alors un couple de sommets et la connexion sera réalisée par un chemin du graphe allant de la source de la demande à la destination. Le multiplexage en longueurs d'onde et la contrainte physique qui en résulte se traduisent par le fait que chaque chemin se voit affecter une longueur d'onde de sorte qu'aucune longueur d'onde ne soit portée par deux chemins qui se croisent sur un lien. La modélisation des réseaux monofibres par des graphes non-orientés a aussi été étudiée dans la littérature pour prendre en compte des contraintes supplémentaires de symétrie. En effet le modèle en graphes non-orienté est équivalent au modèle orienté symétrique contraint à la symétrie de l'instance de communication (s'il y a une requête (u, v), alors il y a la requête (v, u)), du routage (le chemin de la requête (u, v) doit utiliser les mêmes liens que celui de (v, u)), et l'affectation de longueurs d'onde (la longueur d'onde affectée au chemin de (u, v) est la même que celle affectée à celui de (v, u)).

L'affectation de longueurs d'onde aux chemins servant l'instance de communication s'exprime en termes de théorie des graphes exactement comme le problème classique de coloration de chemins dans un graphe avec un minimum de couleurs. Cette quantité correspond en effet à un des facteurs principaux du coût des équipements mis en place dans un réseau réel et son étude est intéressante car l'influence d'une telle fonction de coût, globale à tout le réseau, est très différente de celle d'une fonction telle que la charge du réseau, locale à chaque lien. Alternativement, on peut vouloir maximiser le nombre de requêtes servies avec un nombre donné de longueurs d'onde disponibles, c'est équivalent.

Les applications envisagées à l'optimisation des réseaux WDM, mais aussi au *call scheduling* où il s'agit d'ordonnancer des communications sur un réseau à multiplexage en temps déterministe, ont motivé un nouvel intérêt dans l'étude de la coloration de chemins sur des classes de graphes particulières (Bermond et coll. 1996; Kumar 1998; Erlebach et coll. 1999; Auletta et coll. 2000).

2.2.2 Algorithmique de la coloration de chemins

Récemment ont été publiés de nombreux travaux étudiant la coloration de chemins sur des topologies simples telles les grilles, les anneaux et les arbres (Bermond et coll. 1996; Gargano et coll. 1997; Kumar 1998; Niessen et Kind 1998; Erlebach et coll. 1999; Auletta et coll. 2000). La plupart de ces études s'intéressent aux liens entre la *charge* l de l'ensemble de chemins et le nombre optimal de couleurs nécessaires à une coloration, w. En effet, la charge d'un lien du réseau est le nombre de chemins traversant ce lien et la charge du réseau est la charge du lien le plus chargé. Étant donné que deux chemins qui traversent le même lien ne peuvent avoir la même couleur, la charge du réseau est une borne inférieure au nombre minimum de couleurs nécessaires. Le ratio $\frac{w}{l}$ entre le nombre de couleurs et la charge est une quantité très étudiée car elle permet de mesurer facilement la qualité d'une solution approchée. En effet, il est souvent difficile d'estimer le nombre optimal de couleurs alors que la charge est une quantité immédiatement disponible.

Le cas où le graphe a une topologie en anneau s'appelle le problème de coloration des arcs circulaires. Une première étude de Karapetyan (1980) a donné une $\frac{3}{2}$-approximation. En utilisant un résultat de Tucker (1975) qui réduit la coloration des arcs circulaires à une instance particulière de multiflot entier, Kumar (1998) a proposé une approximation aléatoire de facteur $1+\frac{1}{e} \approx 1.37$. Sa technique, proche de celles utilisées dans nos travaux, consiste principalement à résoudre la relaxation fractionnaire du multiflot et à procéder à un *arrondi aléatoire* de la solution obtenue. Cette technique colore presque tous les chemins avec un nombre de couleurs égal à l'optimal de la relaxation, la faible proportion de chemins restants est coloriée dans une ultime phase utilisant le meilleur algorithme polynomial connu. Cette dernière phase n'engendre qu'une petite perte puisqu'elle ne s'applique qu'à peu de chemins.

Pour les topologies en arbre, une approximation déterministe a été proposée par Erlebach, Jansen, Kaklamanis, Mihail, et Persiano (1999). Cet algorithme, qui colore tout ensemble de chemins de charge l sur un arbre en utilisant au plus $\frac{5l}{3}$ couleurs, est glouton et procède par phases, une par sommet v de l'arbre. L'idée consiste à considérer les sommets dans l'ordre d'un parcours en largeur d'abord. Lorsque le sommet v est traité, l'algorithme fait l'hypothèse qu'il a déjà construit une coloration partielle de tous les

chemins qui *touchent* un sommet antérieur à v et uniquement de ceux-là. On dit qu'un chemin touche un sommet s'il en part, y arrive ou le traverse simplement. Il est aussi fait l'hypothèse que les chemins touchant v utilisent plus de couleurs que la charge, mais que ce surplus est limité à un tiers de la charge. Cette hypothèse et des techniques de recoloriage donnent la liberté de continuer la coloration des chemins touchant v en utilisant en plus un ensemble de couleurs non représentées parmi les chemins déjà coloriés touchant v. La taille de cet ensemble est elle aussi limitée à un tiers de la charge, ce qui amène à un nombre total de couleurs utilisées inférieur à $\frac{5}{3}$ de la charge. Erlebach et coll. (1999) prouvent aussi que cet algorithme est le meilleur algorithme polynomial glouton et déterministe pour ce problème.

Pour améliorer cette approximation, il a donc fallu changer de classe d'algorithme et utiliser une autre approche. Dans le cas des arbres binaires, Auletta, Caragiannis, Kaklamanis, et Persiano (2000) ont proposé un algorithme aléatoire, optimal dans la classe des algorithmes aléatoires, polynomiaux et glouton. Cet algorithme calcule lui aussi une solution en partant de la racine vers les feuilles, mais du fait de l'aléatoire, se comporte comme s'il conservait une distribution des solutions constructibles de cette manière. Avec grande probabilité, chaque élément de cette distribution a la propriété d'être localement *quasiment aléatoire*. Cela signifie que dans chaque étoile centrée en un sommet de l'arbre, tout se passe indépendamment du reste de l'arbre, au prix d'une erreur très faible qui est corrigée *a posteriori*. Cela amène à ce que l'algorithme se comporte comme s'il était toujours dans le cas en moyenne de l'algorithme glouton et déterministe. De la sorte les pires cas sont évités et le facteur d'approximation est amélioré, avec grande probabilité, à $\frac{7}{5} + o(1)$. Il faut toutefois remarquer que les constantes cachées dans les termes d'ordre inférieur, dues à la correction des erreurs faites localement par les choix aléatoires, sont gigantesques et limitent la portée de l'algorithme aux arbres de hauteur petite par rapport à la charge.

2.3 Les réseaux multifibres

Le modèle pour les réseaux monofibres présenté dans la section précédente est très épuré et plus proche de notions de théorie des graphes que de la réalité. En particulier, il

est implicitement fait l'hypothèse que les fibres optiques installées dans le réseau peuvent porter un nombre quasi infini de longueurs d'onde. De plus, on considère qu'entre deux nœuds du réseau une seule fibre est installée dans chaque direction. La première hypothèse est très éloignée de la réalité technologique, la deuxième est de moins en moins réaliste économiquement, en particulier dans les réseaux d'infrastructure.

En effet, le coût prépondérant lors de l'installation d'un réseau est dû à la pose des câbles, en fait aux travaux de terrassement nécessaires à l'enfouissement de ces câbles. En comparaison de ces coûts, le prix d'une fibre dans un câble est négligeable et les opérateurs préfèrent poser des faisceaux de plusieurs fibres pour chaque lien (Porto 2000). De plus la technologie actuelle ne propose que des fibres de capacité bornée en nombre de longueurs d'onde disponibles. Il est donc essentiel de faire évoluer le modèle et de considérer des réseaux *multifibres*, c'est-à-dire où un lien du réseau est constitué d'un faisceau de plusieurs fibres.

Auparavant, seule l'optimisation sur le nombre de longueurs d'onde apparaissait naturellement. Dans le contexte des réseaux multifibres, un autre paramètre est sujet à optimisation : le nombre de fibres sur chaque lien. En effet, même si les opérateurs en installent plusieurs d'un coup, il reste intéressant d'en utiliser le moins possible afin de pouvoir, par exemple, louer les fibres vacantes à d'autres opérateurs ou simplement garder une marge de maneuvre en prévision de croissance du trafic ou de processus ultérieurs de sécurisation du réseau. Par ailleurs, nous verrons que prendre pleinement en compte la multiplicité des fibres sur les liens, notamment la souplesse qu'elle apporte, permet d'exploiter la capacité du réseau de manière plus efficace.

2.3.1 Algorithmes d'affectation de longueurs d'onde

Plusieurs travaux ont étudié des propriétés théoriques des réseaux multifibres sans définir de nouveau modèle combinatoire, un lien constitué de k fibres étant alors modélisé par autant d'arcs parallèles dans le graphe (Margara et Simon 2000; Li et Simha 2001; Choi et coll. 2002).

Dans ce contexte, il a été notamment prouvé qu'augmenter le nombre de fibres présent sur chaque lien du réseau peut radicalement simplifier le problème d'affection de longueurs

d'onde. En particulier, ce problème est $\mathcal{NP}$-complet sur les réseaux monofibres en étoile non orientée mais devient polynomial dès que deux fibres sont disponibles sur chaque lien. En effet, ces fibres supplémentaires modifient structurellement le problème en compensant les contraintes de symétrie du modèle non orienté et rendent le problème équivalent à celui dans les réseaux monofibre en étoile orientée symétrique.

Pour étudier l'influence du degré de liberté apporté par la multiplicité des fibres sur l'efficacité de l'affectation de longueurs d'onde, la charge telle qu'elle est définie dans le contexte des réseaux monofibres n'est plus une notion pertinente. En effet, il est immédiat que le nombre de longueurs d'onde nécessaires est diminué au moins d'un facteur k par rapport à un réseau monofibre s'il y a k fibres par lien : il suffit de considérer un couple (longueur d'onde, numéro de fibre) dans les réseaux multifibres comme une seule longueur d'onde dans les réseaux monofibres. Clairement, s'il existe une affectation de longueurs d'onde aux chemins avec wk longueurs d'onde avec une seule fibre, alors il en existe une avec w longueurs d'onde et k fibres. Cela revient à distribuer les chemins sur chaque fibre et à considérer qu'un chemin garde le même numéro de fibre sur chaque lien comme il garde la même couleur de bout en bout. Tout aussi clairement, le fait de garder le même numéro de fibre de bout en bout est trop contraignant puisqu'en changer à chaque routeur est possible et il doit être possible de faire mieux que ce simple ratio k. Par conséquent, la quantité pertinente dans le cas d'un réseau multifibre est le ratio $\frac{l}{k}$ de la charge sur le nombre de fibres, c'est à dire la charge moyenne de chaque fibre.

La multiplicité des fibres permet une amélioration de l'efficacité de l'affectation de longueurs d'onde bien plus importante que ce simple facteur k. En effet, Margara et Simon (2000) ont montré que pour tout k et w, il existe un réseau et un ensemble de chemins tels qu'il faut au moins w longueurs d'onde avec k fibres par lien, mais une seule longueur d'onde avec $k+1$ fibres. Les mêmes auteurs ont montré que pour tout réseau $\mathcal{N}$donné, il existe un nombre de fibres par lien $k(\mathcal{N})$ à partir duquel tout ensemble de chemins sur $\mathcal{N}$admet une affectation de longueurs d'onde optimale, c'est-à-dire avec autant de longueurs d'onde que la charge moyenne définie ci-dessus (Margara et Simon 2001).

Ce type de résultat constitue une première approche théorique de l'impact de la multiplicité des fibres sur l'affectation de longueurs d'onde, mais ne sont valables que

pour des topologies issues de la théorie, comme dans les travaux de Margara et Simon (2000) et Li et Simha (2001), ou bien ne permettent pas d'exploitation pratique. Par exemple, le nombre de fibres $k(\mathcal{N})$ défini par Margara et Simon (2001) est de l'ordre de $n!$ pour un anneau à n sommets ...

D'autres études se sont intéressées à des situations plus concrètes (Baroni et coll. 1999; Hyytiä et Virtamo 1999; Saad et Luo 2002; Zhang et Qiao 1998; Li et Somani 2000; Kuri et coll. 2002). Afin d'être en capacité de traiter à la fois l'affectation de longueurs d'onde, les conversions en longueurs d'onde et la sécurisation des réseaux, Baroni, Bayvel, Gibbens, et Korothy (1999) considèrent un routage sur des chemins de longueur limitée. Ils proposent une formulation en programme linéaire en nombres entiers qu'ils tentent de résoudre par des heuristiques visant à minimiser le nombre total de fibres dans le réseau. La limitation de la longueur des chemins leur permet de manipuler un moins grand nombre de chemins mais limite aussi les performances des solutions produites. De plus la difficulté de résolution de leur programme linéaire explose littéralement à mesure que l'on autorise des longueurs plus grandes, et la borne sur les longueurs qui donne le meilleur résultat varie grandement avec la topologie du réseau et la structure de l'instance de communication. Il est donc quasiment impossible d'estimer *a priori* la qualité des solutions produites par cette approche. Hyytiä et Virtamo (1999) attaquent le même problème par le biais de métaheuristiques de type recuit simulé ou recherche tabou. Ces deux études mettent en évidence des résultats qui confirment que la multiplicité des fibres accroît l'efficacité du réseau. Saad et Luo (2002) considèrent, eux, le problème associé de maximisation du nombre de chemins qu'un réseau donné peut servir. Ils utilisent des techniques de décomposition lagrangienne pour obtenir des heuristiques de résolution en temps raisonnable de leur formulation en programmation linéaire.

Les réseaux supportant des trafics dynamiques ont aussi été étudiés. Dans ce contexte, les chemins optiques doivent être établis puis retirés dynamiquement. Kuri, Puech, Gagnaire, et Dotaro (2002) traitent d'un modèle déterministe qui permet notamment de modéliser des situations où des clients réservent des demandes de connexion pour une période donnée. Il s'agit alors de dimensionner le réseau en prenant en compte le calendrier de présence des connexions. Le dimensionnement est calculé grâce à une métaheuristique de type recherche tabou. Zhang et Qiao (1998) et Li et Somani (2000) s'intéressent aux

trafics stochastiques qu'ils traitent par des techniques *online*. Ces travaux montrent encore le fait qu'un réseau multifibre est plus efficace qu'un réseau monofibre de même capacité par lien. Ils mettent aussi en évidence que la multiplicité des fibres amène des gains en performance équivalent à ce que peut apporter des équipements de conversion en longueurs d'onde limitée.

Cette dernière constatation est à mettre en cohérence avec la remarque faite au début de cette section. En effet, si l'on considère le couple (longueur d'onde, numéro de fibre) comme une seule longueur d'onde dans un réseau monofibre, alors le fait de pouvoir changer de fibre à chaque routeur se comporte comme un conversion en longueurs d'onde limitée : chaque chemin peut subir une conversion à tous les nœuds mais, à partir d'une couleur (λ, i), ne sont accessibles que les longueurs d'onde de la forme $(\lambda, j),\ j = 1 \dots k$.

2.3.2 De la nécessité d'un nouveau modèle combinatoire

Les méthodes employées dans les travaux cités ci-dessus sont issues de techniques de recherche opérationnelle ou de métaheuristiques, mais même si elles amènent de bons résultats opérationnels, il y a un manque crucial de garantie de performance ou bien elles sont trop spécifiques à des situations théoriques pour être généralisables. Ce manque ne pourra pas être comblé si l'on fait l'économie d'une modélisation plus formelle et pertinente. C'est le but du travail détaillé au chapitre 3 où nous proposons une approche s'appuyant sur la théorie tout en ayant des visées opérationnelles. Ce modèle s'appuie sur une dégradation de l'information contenue dans la définition des chemins. En effet, la notion de conflit paire à paire qui prévaut dans les réseaux monofibres suppose que l'on définisse un chemin dans le réseau comme la suite des fibres qu'il traverse. Nous proposons de ne considérer les chemins dans un réseau multifibre que comme la suite des liens qu'il traverse. Il en résulte la notion de conflit de groupe localisé sur chaque lien qui permet d'envisager l'affectation de longueurs d'onde de manière plus efficace. Ces conflits de groupe se modélisent alors par des hypergraphes et l'information perdue dans la définition d'un chemin, à savoir quelle fibre ce chemin emprunte sur chacun des liens qu'il traverse, se calcule facilement une fois l'affectation de longueurs d'onde terminée.

Chapitre 3

Modélisation combinatoire des réseaux d'infrastructure multifibres

La correspondance directe entre le problème d'affectation de longueurs d'onde dans le modèle de réseau *monofibre* et la coloration de chemins dans les graphes a permis d'obtenir un grand nombre de résultats théoriques et pratiques. Malheureusement, il n'est pas possible d'étendre directement ces études aux réseaux multifibres. En effet, il est très différent d'avoir deux fibres portant w longueurs d'onde sur chaque lien du réseau ou une seule portant $2w$ longueurs d'onde. Le fait d'avoir plusieurs fibres entre deux nœuds modifie la contrainte de coloration, d'un conflit chemin à chemin global au réseau, « `deux chemins non disjoints ne peuvent pas avoir la même couleur` », on passe à une notion de conflit de groupe locale à chaque lien : « `sur un lien à` k `fibres peuvent passer au plus` k `chemins de même couleur` ». Cette nouvelle sorte de conflit ne peut pas être prise en compte par la coloration de chemins dans un graphe.

Les méthodes employées dans les travaux de la littérature sur les réseaux multifibres sont issues de techniques de recherche opérationnelle ou de métaheuristiques, mais même si elles amènent de bons résultats opérationnels, elles pèchent par un manque crucial de garantie de performance. Ce manque ne pourra pas être comblé si l'on fait l'économie d'une modélisation plus formelle et pertinente. C'est le but du travail qui suit où nous proposons une approche s'appuyant sur la théorie tout en ayant des visées opérationnelles. Nous construisons une modélisation des conflits de groupe inhérents aux réseaux multifibres par une nouvelle notion fondée sur la théorie des hypergraphes. Nous validons

ensuite notre modèle en traitant l'optimisation du nombre de fibres par lien et celle du nombre de longueurs d'onde à la fois par des formulations en programmes linéaires et par des algorithmes d'approximation de facteur garanti. Nous présentons aussi les résultats d'expérimentations sur deux réseaux d'infrastructure : le réseau européen COST 239 et le réseau *pan-américain*. Nous présentons ensuite une modélisation algorithmique du routage optique dans les réseaux multifibres par un problème de multiflot dans un graphe auxiliaire. Cette modélisation s'inspire de travaux menés précédemment sur les réseaux monofibres et s'apparente au modèle en hypergraphe précédent dans la manière de considérer les conflits de groupe. Ce modèle sera validé au chapitre 5 où nous expérimentons nos algorithmes d'approximation du multiflot sur les problèmes de routage optique.

Notre contribution principale et l'organisation de ce chapitre sont plus précisément les suivantes. Tout d'abord la section 3.1 donne le cadre de la définition du problème d'affectation de w longueurs d'onde dans les réseaux multifibres avec k fibres par liens, noté le $(k, w)-$WAP. Nous développons aussi une nouvelle modélisation avec la notion d'*hypergraphe des conflits* qui généralise le concept de graphe des conflits défini pour les réseaux monofibres et qui appréhende les interdépendances entre chemins optiques dans les réseaux multifibres de manière pertinente. Grâce à ce modèle, nous pouvons faire le lien entre les résultats de coloration d'hypergraphe de la littérature et le $(k, w)-$WAP.

Forts de ce modèle, nous analysons dans la section 3.1.2 la complexité du problème qui nous occupe. De fait, nous prouvons que le problème de minimisation du nombre de longueurs d'onde est $\mathcal{NP}$-complet et ce même dans le cas où le nombre de fibres est fixé à l'avance. Cela ferme la question de la complexité exacte du problème et nous permet de prouver aussi quelques résultats connexes de bornes inférieures.

Les sections 3.2 et 3.2.3 sont consacrées à l'étude des performances théoriques et pratiques de méthodes exactes et d'approximations fondées sur la coloration d'hypergraphe. Nous validons ces études sur deux réseaux d'infrastructure validés par les opérateurs, un en Europe et l'autre aux États Unis d'Amérique. Par ailleurs nous analysons la faisabilité de la résolution des programmes linéaires en nombres entiers exacts sur des instances réelles du $(k, w)-$WAP avec les programmes de programmation linéaire existants. La méthode approchée que nous utilisons est un algorithme aléatoire combinatoire de facteur

d'approximation garanti de l'ordre du logarithme de la charge du routage. La seconde est un arrondi aléatoire de la relaxation linéaire du problème suivi de techniques de recoloriage. Le facteur d'approximation de ce dernier algorithme est le meilleur qui soit connu dans la littérature théorique, de l'ordre du logarithme de diamètre du routage[1].

La section 3.3 décrit notre modélisation du routage optique par un problème de multiflot entier qui se fonde sur la même notion de conflits de groupe que précédemment et sur l'agrégation des requêtes de communication en *multi-unicast*, ce qui amène à des multiflots plus compacts que ceux proposés dans la littérature pour des problèmes moins généraux. Nous étendons cette modélisation au routage optique dans les réseaux disposant d'équipements de conversion partielle en section 3.3.3.

3.1 L'affectation de longueurs d'onde

Dans la suite, nous définissons formellement le $(k, w)-$WAP, et le modélisons grâce à la notion d'*hypergraphe des conflits* et de quelques concepts connexes. Soit $\mathcal{N}$ un réseau optique, N l'ensemble de ses nœuds et L l'ensemble de ses liens. Nous supposons ici que chaque lien du réseau est un faisceau du même nombre k de fibres. Soit $\mathcal{P}$ un routage sur $\mathcal{N}$, c'est à dire l'ensemble des chemins qui servent les requêtes de communication. L'objectif du $(k, w)-$WAP est de trouver une affectation d'une longueur d'onde parmi w à chaque chemin, de sorte que sur chaque lien du réseau, au plus k chemins partagent la même longueur d'onde. Nous prouvons ensuite que le $(k, w)-$WAP est équivalent à un problème $\mathcal{NP}$-difficile de coloration d'hypergraphes appelé la $(k, c)-$*coloration*.

3.1.1 L'hypergraphe des conflits

Afin de modéliser ces conflits de groupe, nous définissons l'hypergraphe des conflits $H(\mathcal{N}, \mathcal{P})$ du routage $\mathcal{P}$ dans $\mathcal{N}$ comme illustré par la figure 3.1.

Définition 3 *L'*hypergraphe des conflits $H = (V, E)$ *du routage* $\mathcal{P}$ *sur* $\mathcal{N}$ *est l'hypergraphe avec un sommet* $v \in V$ *pour chaque chemin* $p \in \mathcal{P}$*, et un hyperarc* $e \in E$ *pour*

[1] Ce travail a fait l'objet des publications (Rivano 2001), (Ferreira, Pérennes, Richa, Rivano, et Stier 2003a) et (Ferreira, Pérennes, Richa, Rivano, et Stier 2003b)

chaque lien $\ell \in L$. Un hyperarc de H contient les sommets correspondants aux chemins passant par le lien ℓ correspondant. □

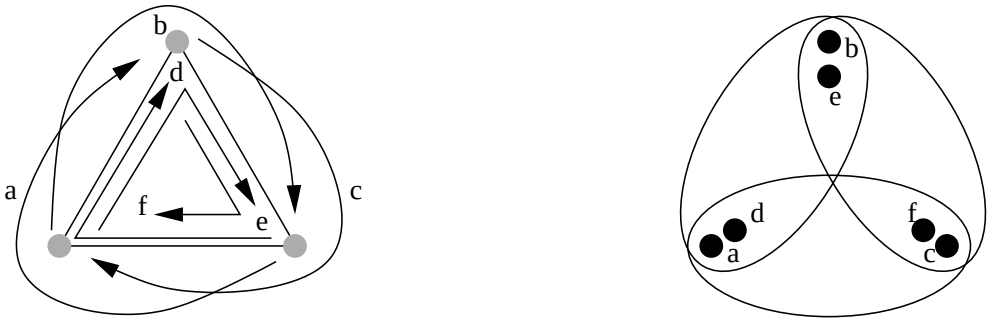


FIG. 3.1: *Un routage sur un réseau en anneau et l'hypergraphe des conflits correspondant.*

Comme pour le graphe des conflits des réseaux monofibres, une coloration des sommets de l'hypergraphe des conflits induit une affectation de longueurs d'onde aux chemins. Cette dernière est valide si et seulement si chaque hyperarc ne contient qu'au plus k sommets de même couleur. On appelle une telle coloration avec c couleurs une $(k, c)-$coloration de l'hypergraphe.

Définition 4 *Étant donné un hypergraphe $H = (V, E)$ et un ensemble de couleurs $\mathcal{C} = \{1 \dots c\}$, une affectation $f : V \to \mathcal{C}$ est une* $(k, c)-$coloration *si et seulement si tout hyperarc ne contient qu'au plus k sommets de même couleur. Autrement dit, $\forall e \in E,\ \forall q \in \mathcal{C},\ |\{v \in e : f(v) = q\}| \leq k$.* □

La modélisation combinatoire des réseaux multifibres par les hypergraphes se continue par l'identification des quatre paramètres principaux d'un hypergraphe des conflits en termes issus des propriétés de $\mathcal{N}$et $\mathcal{P}$.

- le *nombre de sommets* $n \equiv |V| = |\mathcal{P}|$,
- le *nombre d'hyperarcs* $m \equiv |E| = |L|$,
- le *rang* $t \equiv \max_{\ell \in L} |\{P \in \mathcal{P} : \ell \in P\}| \equiv$ la charge de $\mathcal{P}$,
- le *degré maximum* $\Delta \equiv \max_{v \in V} |\{e \in E : e \ni v\}|$.

  Notons que $\Delta \leq max_{p \in \mathcal{P}}$ longueur(p), le diamètre du routage $\mathcal{P}$.

Des définitions 3 et 4 il est trivial de montrer que toute $(k, c)-$coloration de l'hypergraphe des conflits de $\mathcal{P}$ donne une affectation valide de longueurs d'onde à ces chemins.

De plus, comme la construction de l'hypergraphe des conflits de $\mathcal{P}$ dans $\mathcal{N}$est polynomiale en espace et en temps, cela donne une réduction polynomiale du $(k,w)-$WAP à la $(k,c)-$coloration. La $(k,c)-$coloration des hypergraphes est donc au moins aussi difficile que le $(k,w)-$WAP. À première vue la $(k,c)-$coloration des hypergraphes peut sembler être un problème plus général et donc plus difficile que l'autre, mais ce n'est pas le cas : de la même manière que pour le graphe des conflits, il existe une réduction polynomiale inverse qui montre que ces deux problèmes sont équivalents. Cette réduction est l'argument principal des preuves de complexité de la section suivante. Nous anticipons sur la preuve de l'équivalence des deux problèmes et utilisons indifféremment dès maintenant les termes de couleurs et de longueurs d'onde.

3.1.2 Étude de complexité

Nous prouvons ici l'équivalence entre le $(k,w)-$WAP et le problème de $(k,c)-$coloration en montrant que tout hypergraphe est l'hypergraphe des conflits d'un routage sur un réseau multifibre. Nous utilisons cette équivalence pour montrer que le $(k,w)-$WAP est $\mathcal{NP}$-complet même dans le cas où k est donné à l'avance. Nous prouvons aussi une borne inférieure sur le nombre de couleurs nécessaires à la $(k,c)-$coloration d'une *hyperclique*.

Théorème 7 *Le problème de $(k,c)-$coloration des hypergraphes est équivalent au sens de la réduction polynomiale au $(k,w)-$*WAP *dans les réseaux multifibres.*

Preuve : Il a déjà été prouvé en section 3.1 que le $(k,w)-$WAP est équivalent à un sous-ensemble du problème de la $(k,c)-$coloration des hypergraphes, celui de la coloration des hypergraphes des conflits. Afin de prouver le théorème, il suffit de montrer que tout hypergraphe H est l'hypergraphe des conflits d'un routage $\mathcal{P}$ sur un réseau $\mathcal{N}$, tous deux de taille polynomiale en la taille de H.

Soit $H = (\{v_1, \ldots, v_n\}, \{e_1, \ldots, e_m\})$ un hypergraphe quelconque avec n sommets et m hyperarcs.

Nous construisons un réseau $\mathcal{N}$et un routage $\mathcal{P}$ de n chemins de la façon suivante, illustrée par la figure 3.2.

Tout d'abord, on place une colonne de n nœuds $x_j^0,\ j = 1, \ldots, n$. Ces nœuds sont les points de départ des chemins.

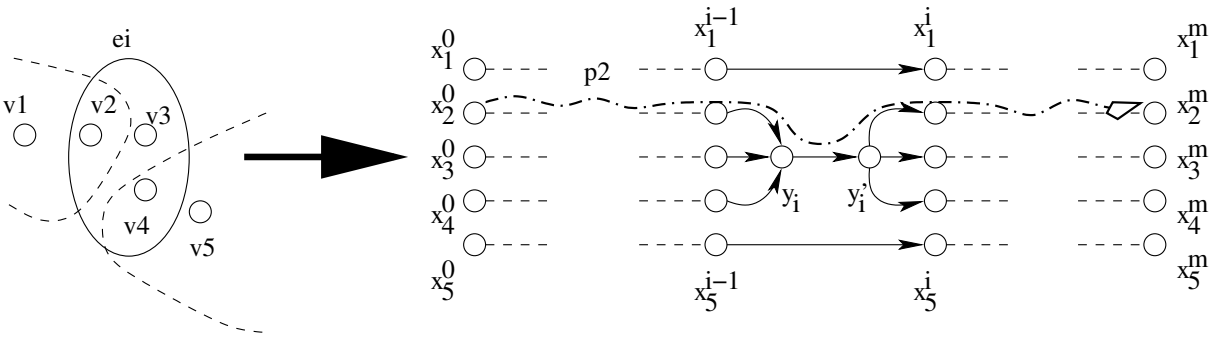


FIG. 3.2: *Un hypergraphe H et le réseau $\mathcal{N}$ du théorème 7.*

Pour chaque hyperarc e_i, on place une couche faite de 2 nœuds y_i et y'_i avec un lien entre les deux et une nouvelle colonne de n nœuds $x^i_j, j = 1 \ldots, n$.

Pour chaque sommet v_j dans e_i, on place un lien entre x^{i-1}_j et y_i, et un entre y'_i et x^i_j.

Pour chaque sommet v_k à l'extérieur de e_i, on place un lien entre x^{i-1}_j et x^i_j.

Pour chaque sommet v_j, on prend le chemin p_j allant de x^0_j à x^m_j qui passe par les nœuds x^i_j et y_i/y'_i si nécessaire.

Par construction, un conflit entre deux chemins p_j et p_k ne peut avoir lieu que sur un lien $y_i \rightarrow y'_i$, ce qui force v_j et v_k à appartenir tous les deux à e_i. Ainsi, H est exactement l'hypergraphe des conflits de $\mathcal{P}$ dans $\mathcal{N}$.

De plus, $|\mathcal{P}|$ vaut n et $|\mathcal{N}|$ est de l'ordre de mn, ce qui donne le théorème.

□

La $(k, c)-$coloration est trivialement $\mathcal{NP}$-complète pour k quelconque puisqu'elle généralise la coloration classique de graphe quand $k = 1$. L'équivalence prouvée ci-dessus donne directement le corollaire suivant.

Corollaire 8 *Le $(k, w)-$WAP des réseaux multifibres est $\mathcal{NP}$-complet pour k quelconque.*

Le résultat suivant n'est pas particulièrement surprenant mais clos toute question sur la complexité de notre problème : il est $\mathcal{NP}$-complet même lorsque k est fixé.

Théorème 9 *La $(k, c)-$coloration des hypergraphes est $\mathcal{NP}$-complète pour tout k fixé.*

Conséquemment, le $(k, w)-$WAP des réseaux multifibres est $\mathcal{NP}$-complet pour tout k fixé.

Preuve : La preuve du théorème se fonde sur une réduction de la $(k, c)-$coloration avec k fixé au problème de décision de la coloration classique des sommets d'un graphe qui consiste, étant donné un graphe G, à répondre à la question « `Les sommets de` G `peuvent ils être coloriés avec` c `couleurs de telle sorte que deux sommets adjacents n'aient pas la même couleur ?` ». Pour prouver que la $(k, c)-$coloration est $\mathcal{NP}$-complète, nous répondons à la question précédente pour tout graphe en appelant l'oracle de $(k, c)-$coloration. Nous pouvons dès à présent supposer que $c < n$, dans les autres cas la réponse est trivialement positive.

Nous prenons donc un graphe G, avec n sommets et m arcs, que nous devons colorier avec c couleurs. Pour appeler l'oracle de $(k, c)-$coloration, nous étendons G en un hypergraphe H de la façon suivante.

- Le but est de construire un hypergraphe pour lequel une seule $(k, c)-$coloration existe (à permutation des couleurs près), chacune des c couleurs étant répétée sur exactement $k + 1$ sommets :
  - Nous définissons l'*hyperclique* $K_{\kappa,\tau}$ comme l'hypergraphe avec κ sommets qui contient tous les hyperarcs possibles de rang τ. Nous plaçons dans H un $K_{ck,k+1}$ qui peut trivialement se $(k, c)-$colorier.
  - Nous fixons une des colorations possibles de $K_{ck,k+1}$.
  - Pour forcer cette coloration à être la seule possible, nous ajoutons c sommets à H, chacun avec une couleur différente et tous les hyperarcs de rang $(k + 1)$ qui joignent au plus k sommets de même couleur.

    Par construction la coloration que nous avons fixée est faisable, mais c'est forcément la seule. En effet si la cardinalité de l'ensemble des sommets ayant la même couleur donnée est modifiée ou si deux couleurs sont permutées, il y aura un hyperarc de rang $k + 1$ qui contiendra plus de k sommets de même couleur.

    La construction permet aussi d'affirmer que H contient $k + 1$ sommets dans chaque couleur.
- Ce bloc central va nous permettre d'ajouter des sommets représentant ceux de G et des hyperarcs contraignant l'oracle de $(k, c)-$coloration à construire une coloration

valide des sommets de G. Ainsi, pour chaque sommet de G, on place un sommet dans H. Pour chaque arc e de G et chaque couleur χ, on place un hyperarc contenant $k-1$ sommets du bloc central coloriés avec χ et les deux sommets correspondants aux extrémités de e. La $(k,c)-$coloration de cet ensemble d'hyperarcs force que les deux extrémités n'aient pas la même couleur.

- Par conséquent, s'il existe une $(k,c)-$coloration de H, elle induit une coloration valide des sommets de G avec au plus c couleurs.

Il ne reste plus qu'à montrer que la construction de H est polynomiale en n, m et c. Puisque k est fixé, il n'est pas nécessaire que la dépendance à k soit polynomiale. Le nombre total de sommets de H est $c(k+1)+n$. Nous avons ensuite ajouté une partie de l'ensemble des hyperarcs de rang exactement $k+1$, c'est à dire moins que

$$\binom{c(k+1)}{k+1} \leq \frac{c^{k+1}(k+1)^{k+1}}{(k+1)!}$$

Cette quantité est assurément polynomiale en c, même si elle ne l'est bien sûr pas en k. Nous avons ensuite ajouté mc hyperarcs pour assurer d'avoir une coloration valide des sommets de G. Puisque nous avons supposé que $c < n$, le nombre total d'hyperarcs de H est $O(n^{k+1}+mn)$, ce qui termine la preuve.

□

Une borne inférieure

En étendant la notion des cliques d'un graphe aux hypercliques dans les hypergraphes, nous donnons une borne inférieure sur le nombre de couleurs nécessaires à une $(k,c)-$coloration. Rappelons que l'hyperclique $K_{\kappa,\tau}$ est l'hypergraphe à κ sommets qui contient tous les hyperarcs de rang τ.

Lemme 10 *Il existe une $(k,c)-$coloration de $K_{\kappa,\tau}$ si et seulement si*

$$c \geq \begin{cases} \lceil \frac{\kappa}{k} \rceil & si\ \tau > k, \\ 1 & sinon. \end{cases}$$

Preuve : Le cas $\tau \leq k$ est trivial puisque les hyperarcs sont de rang inférieur à k : il n'y a aucune contrainte de coloration et une unique couleur suffit.

Dans le cas où $\tau > k$, faisons l'hypothèse que $K_{\kappa,\tau}$ peut être colorié avec c couleurs. Puisqu'il y a κ sommets, il existe au moins une couleur χ qui est présente sur $\lceil \frac{\kappa}{c} \rceil$ sommets. Puisque tous les hyperarcs de rang τ sont présents, il faut $\lceil \frac{\kappa}{c} \rceil \leq \tau$ et il y a au moins un hyperarc contenant $\lceil \frac{\kappa}{c} \rceil$ sommets avec la couleur χ. Il en suit qu'il faut que $\lceil \frac{\tau}{c} \rceil \leq k$. Le minimum pour lequel le nombre de couleurs satisfait cette condition est $\lceil \frac{\kappa}{k} \rceil$.

Par ailleurs, en considérant les propriétés de symétrie de l'hyperclique, on peut montrer qu'une $\left(k, \lceil \frac{\kappa}{k} \rceil\right)$ −coloration de $K_{\kappa,\tau}$ est une distribution uniforme des couleurs sur les sommets. □

Le lemme précédent donne une borne directe sur le nombre de couleurs nécessaire à la (k,c)−coloration de tout hypergraphe contenant une hyperclique $K_{\kappa,\tau}$. Ce résultat généralise le résultat sur les graphes qui affirme que le nombre chromatique d'un graphe est au moins la taille de sa clique maximale, il suffit de prendre $\tau = 2$ et $k = 1$.

Corollaire 11 *Soit H un hypergraphe contenant $K_{\kappa,\tau}$. Si H peut être (k,c)−colorié avec $k < \tau$, alors $c \geq \lceil \kappa/k \rceil$.*

La section suivante traite de la résolution pratique du (k,w)−WAP des réseaux multifibres. Les techniques employées se fondent sur l'équivalence avec la (k,c)−coloration prouvée précédemment et les résultats de complexité que nous venons de démontrer nous poussent à rapidement se concentrer sur des algorithmes d'approximations.

3.2 Dimensionnement des réseaux multifibres

Dans cette section nous présentons deux scénari différents pour le dimensionnement des réseaux multifibres. Puisque le nombre de fibres et le nombre de longueurs d'onde sont deux paramètres d'optimisation concurrents, et qu'il est difficile de comparer le coût d'un des paramètres par rapport à l'autre, nous considérons les optimisations mono-critères où l'un des paramètres est fixé. Le premier scénario se rapproche de la situation qu'un opérateur rencontre quand il doit installer un réseau : il sait quel type d'équipement est à sa disposition et tente de minimiser le coût des liens du réseau. Le deuxième considère

plutôt que le réseau est déjà présent et que l'opérateur veut mettre à jour les équipements sans toucher aux liens qui sont enterrés. L'équivalence entre le WAP et la coloration d'hypergraphe nous permet d'utiliser des approches développées pour le deuxième problème.

3.2.1 Minimiser le nombre de fibres

Considérons tout d'abord le problème de minimiser le nombre de fibres quand le nombre de couleurs est donné. Cette optimisation peut, toutes proportions gardées, se comparer à la situation d'un opérateur installant un nouveau réseau. Si l'on imagine que les routeurs optiques sont déjà choisis, le nombre de longueurs d'onde disponibles sur le réseau est fixé et l'opérateur va tenter de minimiser le coût des liens qu'il doit installer. Bien entendu la situation réelle est beaucoup plus complexe, mais ce genre d'optimisation peut au moins aider à la décision de l'opérateur. Un autre objectif peut être poursuivi par l'opérateur dans le cas où son réseau est en place. En effet, il peut voir son intérêt à minimiser le nombre de fibres utilisées par son affectation de fréquences, afin d'en réserver un maximum pour la location à des clients particuliers, ou même pour assurer une certaine résistance aux pannes des équipements présents dans les routeurs.

En exprimant le problème en termes d'hypergraphes, on peut utiliser la formulation en programme linéaire en nombres entiers *min-max* donnée par Srinivasan (1996). Cette formulation consiste à définir une variable $x_{i,c} \in \{0,1\}$ pour tout sommet $i \in V$ et couleur $c \in \{1 \dots c\}$ telle que $x_{i,c} = 1$ si et seulement si le sommet i a la couleur c et 0 sinon. Les contraintes de coloration des hyperarcs s'expriment alors comme des sommes de ces variables et le nombre optimal de fibres se calcule en résolvant le programme linéaire en nombres entiers 8.

Programme linéaire 8

$$
\begin{array}{lrcll}
\textit{minimiser} & k & & \textit{(nombre de fibres)} \\
\textit{t.q.} & \displaystyle\sum_{c} x_{i,c} & = & 1 & \forall\ i \textit{ un sommet} \\
& \displaystyle\sum_{i \in \ell} x_{i,c} & \leq & k & \forall\ c \textit{ une couleur}, \forall\ \ell \textit{ un hyperarc} \\
& k \geq 0,\ x_{i,c} & \in & \{0,1\} & \forall\ c \textit{ une couleur}, \forall\ i \textit{ un sommet.}
\end{array}
$$

Srinivasan (1996) a montré que l'arrondi aléatoire d'une solution optimale à la relaxation linéaire du programme linéaire 8 produit, avec probabilité positive, une solution valide et proche de l'optimum à un facteur de l'ordre du logarithme du degré de l'hypergraphe près. Par ailleurs, dans notre cas, le degré de l'hypergraphe des conflits est égal au diamètre du routage qu'il représente. Ce résultat se fonde sur une extension du lemme local de Lovász donnée dans le même article. Inspiré par ce résultat existentiel, Lu (1998) a proposé un algorithme aléatoire combinatoire qui calcule une solution approximant l'optimum, mais celui-ci ne garantit qu'un facteur moins bon que celui donné par Srinivasan (1996). Cet algorithme commence par affecter aléatoirement à chaque sommet une couleur parmi le tiers des couleurs disponibles. Il détecte ensuite les ensembles de sommets qui sont *mal coloriés*, recolore les sommets concernés aléatoirement avec un deuxième tiers, puis refait une phase de détection des sommets mal coloriés. Lu (1998) prouve alors qu'avec grande probabilité, les sommets restant à re-colorier sont suffisamment peu nombreux pour qu'une coloration exacte, donc exponentielle, soit faite avec le dernier tiers de couleurs restant. L'analyse de l'algorithme montre que le facteur d'approximation de son algorithme est de l'ordre du logarithme de la charge du routage.

Une amélioration de cet algorithme a été récemment proposée par Leighton, Lu, Rao, et Srinivasan (2001). L'idée est que l'algorithme de Lu (1998) fonctionne comme s'il partait d'une solution triviale du programme linéaire, et qu'il faisait un arrondi aléatoire en ne considérant les couleurs que par tiers. Ce faisant, il s'empêche intrinsèquement d'exploiter pleinement la plage de couleurs disponibles, ce qui l'amène à des solutions moins bonnes que ce que l'analyse théorique de Srinivasan (1996) promet. Leighton et coll. (2001) prennent le parti de construire la solution dont l'existence est prouvée par Srinivasan (1996). Ils commencent donc par un arrondi aléatoire de la relaxation linéaire du programme linéaire 8. Pour passer d'une simple assurance de l'existence d'une bonne solution à l'assurance de sa construction, l'arrondi aléatoire est suivi d'une phase de recoloration comparable à celles employées dans l'algorithme de Lu (1998). L'emploi de la programmation linéaire pour obtenir un bon guide à l'arrondi aléatoire est coûteux en temps de calcul, mais il permet d'atteindre le facteur d'approximation théorique, le logarithme du diamètre du routage. Nous verrons de plus, en section 3.2.3, que les techniques de recoloration ne sont jamais utilisées en pratique, car l'arrondi aléatoire suffit pour

générer des solutions proches de l'optimum. Cette situation se retrouve dans le chapitre suivant où nous verrons aussi que les analyses des algorithmes fondés sur de l'arrondi aléatoire sont souvent très pessimistes en pratique. Il est d'ailleurs intéressant de noter que les algorithmes de Lu (1998) et de Leighton et coll. (2001) peuvent être *déterminisés* en utilisant les techniques classiques d'*estimateurs pessimistes*. Cette technique consiste à remplacer les décisions aléatoires par des choix déterministes guidés par une évaluation pessimiste de l'impact d'un tel choix. Cette évaluation peut se faire, par exemple, à l'aide de probabilités conditionnelles fondées sur l'analyse de la version aléatoire de l'algorithme. Le fait même que l'on puisse employer ce genre de techniques montre que la version aléatoire peut se comporter bien mieux que ce que l'analyse garantit.

3.2.2 Minimiser le nombre de longueurs d'onde

Étant donné le nombre k de fibres présentes sur chaque lien du réseau, il est intéressant de trouver le nombre minimal de longueurs d'onde nécessaires au WAP. Cette optimisation peut représenter l'intérêt d'un opérateur disposant d'un réseau multifibre installé pour lequel il décide de modifier les équipements en place dans les routeurs à moindre coût, ou bien en gardant un maximum de longueurs d'onde disponibles pour d'autres usages.

En termes de coloration d'hypergraphes, cela signifie minimiser le nombre de couleurs c tel qu'une $(k, c)-$coloration valide de l'hypergraphe existe, ce qui s'écrit comme le programme linéaire en nombres entiers 9 de la façon suivante. Nous définissons une variable $x_{i,c} \in \{0, 1\}$ pour chaque sommet i et couleur c telle que $x_{i,c} = 1$ si le sommet i prend la couleur c, 0 sinon. Nous avons vu précédemment que le nombre de couleurs est au plus $\lceil n/k \rceil$ s'il y a n sommets, avec égalité quand le graphe est une hyperclique. Nous définissons aussi pour chaque couleur c une variable $y_c \in \{0, 1\}$ qui vaut 1 si un sommet prend la couleur c. Ces variables permettent de compter le nombre de couleurs effectivement utilisées.

Programme linéaire 9

$$
\begin{aligned}
\min \quad & \sum_c y_c && && \textit{(nombre de couleurs)} \\
& \sum_c x_{i,c} && = \quad 1 && \forall\ i \textit{ un sommet} \\
& \sum_{i \in \ell} x_{i,c} && \leq \quad k && \forall\ c \textit{ une couleur, } \ell \textit{ un hyperarc} \\
& x_{i,c} && \leq \quad y_c && \forall\ c \textit{ une couleur, } i \textit{ un sommet} \\
& x_{i,c}, y_c && \in \quad \{0,1\} && \forall\ c \textit{ une couleur, } i \textit{ un sommet}
\end{aligned}
$$

Il est une objection que l'on peut faire au programme linéaire précédent : il n'est pas très intéressant d'un point de vue pratique. En effet, d'une part le programme est volumineux puisqu'il faut $O(n^2)$ variables et $O(n^2 m)$ contraintes. D'autre part, la symétrie de la formulation est critique : Mehrotra et Trick (1996) ont montré que les techniques de *Branch-and-Bound* utilisées pour sa résolution vont perdre beaucoup de temps à parcourir un grand nombre de solutions équivalentes. Cette difficulté est due au fait qu'après avoir contraint une variable, une permutation des autres variables peut toujours amener à une solution valide et de même coût sans que ce soit facilement détectable par l'algorithme de résolution à cause des variables y_c. Toutefois, Margot (2001) a développé des techniques d'élagage automatique de l'arbre de *Branch-and-Bound* qui peuvent s'appliquer à ce genre de situations.

3.2.3 Implémentation et évaluation de performances

Afin de valider l'analyse des algorithmes présentés précédemment et la pertinence de notre modélisation, nous avons implémenté les algorithmes aléatoires de Lu (1998) et Leighton et coll. (2001) que nous comparons avec les formulations en programmes linéaires en nombres entiers. L'idée est principalement d'évaluer le compromis entre la qualité et le temps de calcul des solutions. Nous montrons aussi les difficultés évoquées plus haut pour l'optimisation du nombre de longueurs d'onde. Nous avons testé ces approches dans plusieurs situations, et nous présentons les résultats obtenus sur deux réseaux d'infrastructure continentaux.

Les instances

Le premier réseau que nous avons utilisé est illustré en figure 3.3. COST 239 interconnecte 11 capitales européennes par 24 liens multifibres. Les requêtes de communications, donnée publique de France Télécom (Porto 2000), sont au nombre de 176 et il y a en a au moins une par paire de villes. Le routage de ces requêtes induit une charge maximum de 58, ce qui nous donne une borne inférieure sur le nombre de longueurs d'onde nécessaires.

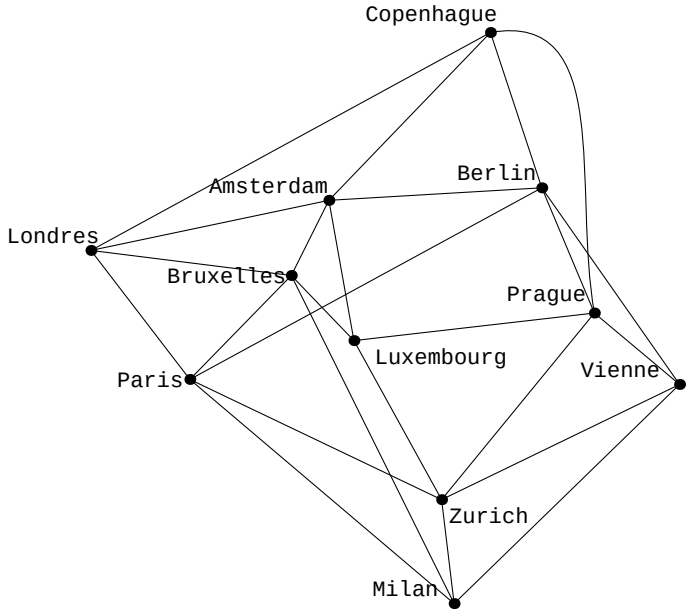


FIG. 3.3: *Le réseau européen* COST 239.

Le second réseau, illustré en figure 3.4, couvre les États Unis d'Amérique en interconnectant 78 des principales villes avec 102 liens. L'ensemble des requêtes de communication a été généré par le modèle gravitationnel classique. Nous avons choisi de pondérer les villes par leur importance démographique et leur distance à 5 des principaux bassins de population du pays. Enfin, le nombre de requêtes entre deux villes est proportionnel au produit des poids des deux villes. Différentes instances ont été générées selon la façon de fixer les poids. Nous avons ensuite choisi de prendre en compte des considérations de résistance aux pannes du réseau en utilisant une politique de routage centrée autour du problème des *deux chemins disjoints de coût minimum*. Ainsi, pour chaque paire origine-destination, nous calculons les deux chemins disjoints dont la somme des longueurs est minimale et distribuons aléatoirement les requêtes sur ces deux chemins. Nous présentons

ici les résultats obtenus sur une instance assez importante de 2002 requêtes induisant une charge maximale de 520.

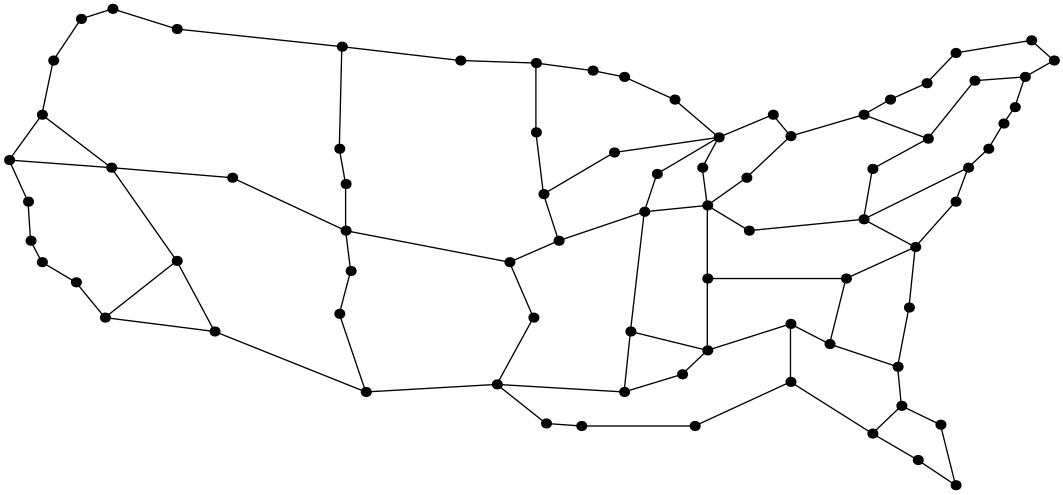


FIG. 3.4: *Le réseau pan-américain.*

Les résultats

Les tests que nous avons réalisés nous ont permis de valider les analyses théoriques des algorithmes et certaines intuitions sur leur comportement pratique. Toutefois, il convient d'être prudent devant les résultats numériques que nous présentons. Même s'ils sont représentatifs de l'ensemble des tests que nous avons pu réaliser, rien n'indique qu'ils le soient de la réalité sur le terrain. En effet, l'ensemble des requêtes de communication qu'un opérateur doit servir est une donnée extrêmement stratégique que l'opérateur ne diffuse pas. Par exemple, les données disponibles pour le réseau COST 239 sont une distortion suffisamment importante de la réalité pour pouvoir les rendre publiques sans risque pour un opérateur. Après tout, il nous importe peu d'avoir des données réelles puisque nos algorithmes sont génériques et leurs performances sont garanties indépendamment des instances qu'ils traitent. Par contre, il est important que l'on puisse compter sur le réalisme de ces données si l'on veut aller plus loin que la simple validation des analyses théoriques. Malheureusement, personne ne sait aujourd'hui dire en quoi un ensemble de requêtes de communication est réaliste, ni même quelles sont les caractéristiques pour lesquelles le réalisme est le plus fondamental.

L'instance de communication que nous avons utilisée sur le réseau d'infrastructure américain est générée en utilisant le modèle gravitationnel classique. Il est certain que ce modèle transcrit une certaine réalité sociologique, mais nous ne savons si cela suffit à rendre une instance réaliste pour un réseau optique d'infrastructure. Enfin, les stratégies de routage que nous avons employées tentent de prendre en compte les contraintes de résistance aux pannes des réseaux. Cependant l'étude des routages robustes est loin d'être achevée et nous ne pouvons pas certifier avoir utilisé une stratégie qui sera employée en pratique. Il faut donc relativiser la portée de nos conclusions qui ne peuvent s'attacher qu'à des considérations générales et des comparaisons entre nos différentes approches.

Résolution exacte

Concernant la résolution du programme linéaire 8 pour la minimisation du nombre de fibres, il semble acquis que les programmes de résolution sont capables de traiter des instances de petite taille en temps raisonnable. En effet, que ce soit sur le réseau européen COST 239 ou le réseau américain avec peu de longueurs d'onde disponibles, le programme de résolution, CPLEX v7.5, est toujours arrivé à prouver l'optimalité de ses solutions. Bien entendu, puisque le problème sous-jacent est $\mathcal{NP}$-difficile, le temps de calcul et l'espace en mémoire explosent rapidement à mesure que la taille des instances augmentent. Le fait que CPLEX arrive à optimiser les instances de taille raisonnable peut être toutefois dû à une certaine facilité des instances qui est récurrente dès que l'on traite de réseaux réels. Il est par contre assuré qu'il est difficile d'envisager l'optimisation de réseaux déployant plus de quelques dizaines de longueurs d'onde, forçant l'utilisation de méthodes approchées dès que les technologies CWDM (*coarse* WDM) ou DWDM (*dense* WDM) sont déployées et offrent de quelques centaines à plusieurs milliers de longueurs d'onde.

Résolution approchées

Nos tests ont aussi apporté des précisions sur le fonctionnement des algorithmes d'approximation que nous avons implémenté. L'algorithme de Lu (1998) produit toujours des solutions approchées à un facteur proche de celui donné par l'analyse théorique. De plus, la stratégie qui consiste à diviser l'ensemble des longueurs d'onde disponibles en trois paquets qui sont distribués l'un après l'autre a des conséquences directes sur le fonc-

tionnement même de l'algorithme. L'expérience montre donc, et c'est assez intuitif, que l'algorithme se comporte bien mieux lorsque le nombre de longueurs d'onde disponibles est un multiple de 3. Ce comportement se voit aussi dans la valeur du nombre de fibres que calcule l'algorithme et qui décroît par palier à mesure que le nombre de longueurs d'onde augmente. Le facteur d'approximation réel des solutions varie donc très fortement.

L'algorithme de Leighton et coll. (2001) qui se fonde sur un arrondi aléatoire est bien sur plus performant, puisqu'il ne fait pas cette division arbitraire de l'ensemble des couleurs, et qu'il utilise une solution optimale de la relaxation linéaire. L'expérience montre que l'algorithme n'utilise en pratique jamais les techniques de recoloriage qui sont prévues pour améliorer la qualité des solutions produites par l'arrondi aléatoire en cas de trop grande déviation. En effet, et comme cela est illustré aussi au chapitre suivant, l'arrondi aléatoire est un processus qui produit de très bonnes solutions même s'il peut théoriquement dévier bien plus de l'optimal.

Dimensionnement du réseau COST 239

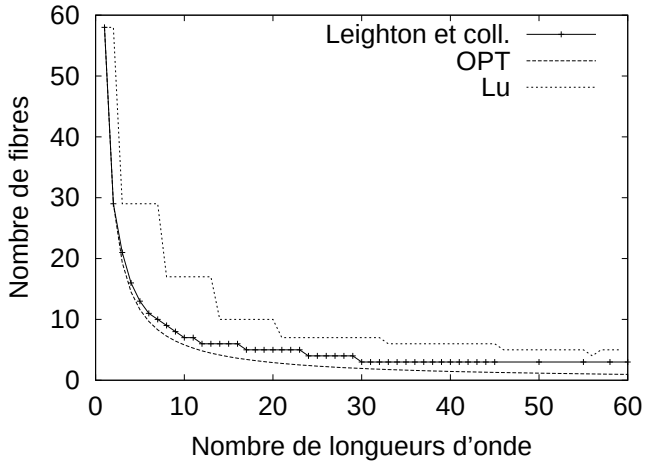


FIG. 3.5: *Optimisation du réseau* COST 239.

La figure 3.5 présente les résultats des différentes méthodes d'optimisation du nombre de fibres en fonction du nombre de longueurs d'onde disponibles sur le réseau COST 239. La courbe OPT montre le dimensionnement optimal donné par le programme linéaire 8, les deux autres courbes montrent les résultats des deux algorithmes approchés.

Nous l'avons précisé plus haut, le programme linéaire 8 a pu être optimisé même avec un nombre conséquent de longueurs d'onde grâce à la petite taille de l'instance, ce qui nous permet de comparer les autres algorithmes avec précision. La petite taille de l'instance rend par contre difficile la comparaison des temps de calculs : les deux algorithmes approchés terminent quasiment immédiatement. Il n'y a donc pas de courbe pour ces temps, ils seront étudiés sur le réseau américain.

Comme attendu, l'algorithme de Lu (1998) donne les solutions les plus lointaines de l'optimal et la dépendance en escalier du nombre de fibres par rapport au nombre de longueurs d'onde est très visible. L'algorithme de Leighton et coll. (2001) produit des solutions quasi-optimales à une constante additive près, 5 en l'occurrence.

Dimensionnement du réseau pan-américain

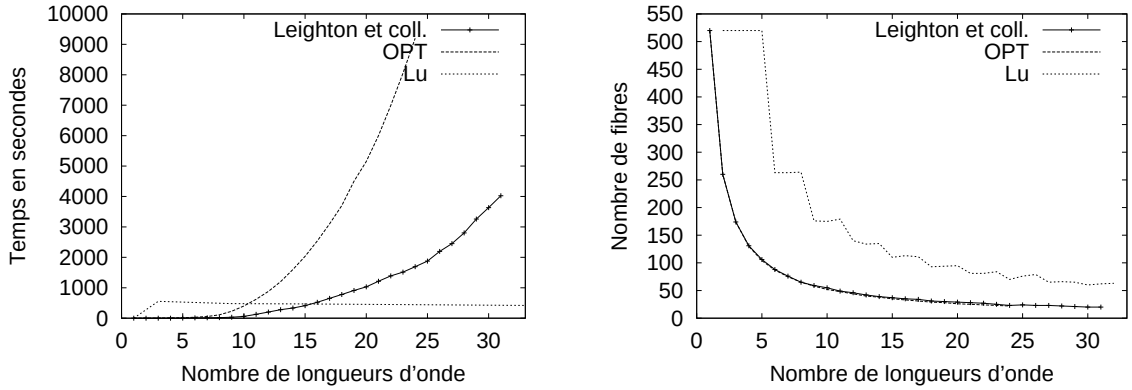


FIG. 3.6: *Temps de calcul et résultats de l'optimisation du réseau américain.*

La taille importante du réseau américain permet de mettre en lumière les tendances lourdes du comportement des algorithmes. Ainsi les temps de calcul des différentes méthodes de résolution, représentés par les courbes de gauche de la figure 3.6, indiquent clairement que la résolution du programme linéaire en nombres entiers prend un temps exponentiel en le nombre de longueurs d'onde disponibles, et qu'elle devient impossible à partir d'un certain seuil, 25 longueurs d'onde sur nos ordinateurs (PIV 1GHz, 512Mo de RAM). Par ailleurs cette impossibilité n'est pas le seul fait d'un temps de calcul prohibitif, l'espace mémoire nécessaire au *Branch-and-Bound* dépasse largement les capacités de la machine. Le caractère exponentiel de cette croissance ne permet pas d'espérer aller

beaucoup plus loin avec des ordinateurs plus puissants, et il semble bien que l'optimisation de réseaux avec quelques centaines de longueurs d'onde ne peut pas être réalisé par cette méthode. L'algorithme de Leighton et coll. (2001) prend un temps polynomial en le nombre de longueurs d'onde qui laisse présager des difficultés sur des réseaux de grande capacité, mais pas de véritable impossibilité.

Ce qui est plus surprenant, c'est que l'algorithme de Lu (1998) prend, lui, un temps quasi constant, qui décroît même lentement à mesure que le nombre de longueurs d'onde augmente. Cela parce que, contrairement aux autres méthodes, l'augmentation du nombre de couleurs ne fait pas augmenter la complexité combinatoire de l'optimisation, et la fait même diminuer, puisque le nombre de sommets devant être recolorés lors de la phase exhaustive a tendance à diminuer quand le nombre de longueurs d'onde augmente.

Bien entendu, des temps de calcul inférieurs se paient sur la qualité du dimensionnement. Du fait de la grande taille de l'instance, la loi des grands nombres joue en faveur de l'arrondi aléatoire qui est collé à la courbe optimale et ne s'en écarte que rarement de quelques unités, 4 au pire.

Remarques sur les performances de l'arrondi aléatoire

Cette précision importante contraste avec l'analyse théorique des arrondis aléatoires qui est un peu moins optimiste. Pourtant ce comportement se retrouve dans d'autres applications comme nous le verrons dans le chapitre suivant. Plusieurs hypothèses permettraient d'expliquer ce phénomène. D'une part, il est possible que l'analyse probabiliste emploie des résultats de probabilités peu adaptés à ce type d'algorithme. D'autre part il n'est pas exclu que la nature des programmes linéaires ait un impact. En effet, il s'agit tout le temps de relaxations linéaires de problèmes combinatoires où tous les paramètres sont entiers. Comme la solution optimale de laquelle on part est un point extrémal du polytope, des propriétés parasites peuvent apparaître qui ne sont pas prises en compte dans les analyses. En particulier, les solutions sont souvent quasi entières. Enfin, peut être les choix de programmation faits lors du développement de CPLEX lui font-ils privilégier des solutions qui ont la propriété de bien s'arrondir. Toujours est-il que les arrondis aléatoires se comportent bien mieux en pratique qu'en théorie, ce qui implique dans notre cas que les techniques de recoloration prévues dans cet algorithme ne sont jamais mises

en oeuvre.

En ce qui concerne l'algorithme de Lu (1998), la taille de l'instance joue finalement en sa défaveur puisqu'il est toujours proche de ses performances théoriques. Ainsi, il produit une solution qui approxime l'optimum à un facteur multiplicatif proche de 3. On observe toujours l'évolution par palier des solutions.

Compromis temps/qualité

La conjonction de ces deux mesures de performance permet de situer toutes les approches dans un compromis entre temps de calcul et qualité du dimensionnement. Ainsi sur des petites instances, il semble naturel de rechercher le dimensionnement optimal. Par contre, dès que les conditions deviennent plus difficiles, il faut se tourner vers les algorithmes d'approximation. Tout dépend alors du contexte et des conditions d'exploitation. Si le temps de calcul n'est pas une ressource critique, par exemple s'il s'agit de décider quelles sont les modifications à apporter au réseau, l'algorithme de Leighton et coll. (2001) est la bonne stratégie puisqu'elle donne des solutions quasiment optimales. Par contre, si le temps de calcul est plus critique que la qualité de la solution, par exemple si le calcul a vocation à être intégré dans un procédé automatique de réaction aux modifications du trafic, l'algorithme de Lu (1998) permet d'obtenir des solutions de qualité même si elles ne sont pas optimales, et surtout en temps très raisonnable et en utilisant peu de mémoire. Il serait d'ailleurs intéressant de mettre au point des solutions intermédiaires.

Minimisation du nombre de longueurs d'onde

Concernant l'optimisation du nombre de longueurs d'onde en fonction du nombre de fibres, l'étude est moins poussée. La principale dépendance du temps de calcul du programme linéaire 9 est en le nombre de variables représentant les couleurs. Dans un premier temps, nous mettions autant de couleurs que de requêtes puisque c'est une borne supérieure évidente. Évidemment, la résolution de ce programme présentait d'importantes difficultés de passage à l'échelle lorsque le nombre de requêtes de communication augmentait et n'arrivait à rien sur des instances de taille réaliste. En effet le trop grand nombre de variables fait passer énormément de temps à CPLEX pour résoudre ne serait-ce que le premier sommet de l'arbre de *Branch-and-Bound*. Nous avons ensuite décidé de

faire une recherche dichotomique pour borner supérieurement le nombre de couleurs de manière plus fine. Cette stratégie s'appuie sur l'observation que, lorsqu'il y a trop peu de couleurs autorisées, la relaxation linéaire montre une impossibilité rapidement. Nous avons ainsi pu obtenir des programmes avec un nombre réduit de variables, que CPLEX arrive à traiter, même si, comme il est prévisible, la symétrie du problème empêche que l'énumération des sommets de l'arbre de *Branch-and-Bound* puisse être exhaustive, et donc l'optimalité d'une solution est difficile à prouver directement. Nous avons toutefois pu obtenir des preuves d'optimalité des solutions à cause d'une propriété parasite de nos instances. Le nombre de couleurs nécessaires est, ici, égal à la borne inférieure. Par conséquent, si l'on diminue le nombre de couleurs disponibles, la relaxation linéaire est elle même infaisable. Cependant, ce comportement ne peut pas être considéré comme acquis, puisqu'il existe des instances issues de la théorie où ce n'est pas le cas. Nous suspectons d'ailleurs que ce comportement parasite disparaîtra si l'on utilise un autre routage, ou si l'on opère un dimensionnement non uniforme du réseau, c'est-à-dire si l'on met plus de fibres sur les liens les plus chargés comme par exemple les 3 liens qui forment l'essentiel de la connectivité du réseau américain entre les côtes est et ouest.

3.3 Le routage optique

À l'instar de la plupart des problèmes d'optimisation des télécommunications, le routage optique (RWA) peut s'écrire de multiples façons comme un programme linéaire en nombres entiers. Toutefois, ces écritures ne permettent souvent pas de bonnes relaxations et ne sont utiles que grâce à l'existence de solveurs efficaces. Le RWA aurait pu être un cas de plus sans une meilleure compréhension de la nature même du problème.

Dans la suite, nous montrons que le RWA est un problème de multiflot dans un graphe auxiliaire. Nous nous inspirons des travaux de Beauquier, Hell, et Perennes (1998) qui ont étudié le routage d'instances *multi-unicast* dans les réseaux WDM monofibres sans conversion. Récemment, une approche très similaire a été appliquée à des instances générales par Krishnaswamy et Sivarajan (2001), mais toujours dans les réseaux monofibres sans conversion. Nous montrons dans la section suivante comment les instances multi-unicast

se modélisent par un simple flot et les instances générales par un multiflot dans un graphe auxiliaire. Nous montrons aussi que ce multiflot est plus efficace que celui proposé par Krishnaswamy et Sivarajan (2001) pour un problème moins général, notamment grâce à l'agrégation des requêtes du routage optique en multi-unicast[2].

3.3.1 Multi-unicast et flot

L'entité élémentaire des instances de communication est la requête entre deux nœuds à laquelle l'on affecte un chemin optique. Il peut donc y avoir un nombre quadratique de requêtes différentes dans un réseau, une par couple de sommets. Il est toutefois possible d'agréger ces requêtes afin de manipuler une instance formée d'entités de plus haut niveau.

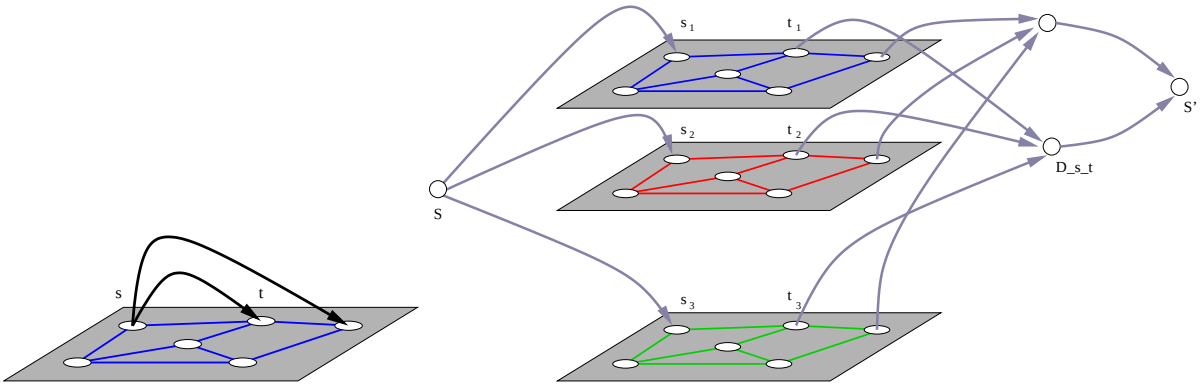
En effet, une instance I peut être considérée à une granularité moins fine comme l'ensemble des multi-unicasts $M_s = \{\{(s, y)|(s, y) \in I\}\}$ regroupant toutes les requêtes provenant de la source s. Cette agrégation permet de ne plus considérer qu'une seule entité par source de trafic. De plus, les remarques suivantes permettent d'écrire le RWA des multi-unicast comme un problème de flot :

1. La capacité utilisée par n unités de flot depuis un nœud s vers un nœud t dans un réseau de flot, peut être décomposée en n chemins simples de capacité 1, allant de s à t.
2. L'ensemble des chemins composant un flot utilise sur l'arc e une charge égale à la quantité de flot sur e.
3. Dans un réseau WDM, la charge sur un lien e (i.e. : le nombre de chemins passant par e) d'un ensemble de chemins pouvant utiliser la même longueur d'onde est au plus $k(e)$, le nombre de fibres sur ce lien.

Étant donné un réseau G, une instance de multi-unicast M enracinée en s et les ressources du réseau (w, $k(e)$ et $c(e)$), nous construisons un réseau de flot tel qu'il y ait un flot de $|M|$ unités entre la source et le puits si et seulement s'il existe un chemin optique entre s et chacune de ses destinations.

[2]Ce travail a fait l'objet de la publication (Coudert et Rivano 2002)

La figure 3.7(b) décrit le réseau de flot permettant de résoudre le RWA pour le réseau et l'instance de multi-unicast de la figure 3.7(a) avec 3 longueurs d'onde.



(a) Une requête de multi-unicast.

(b) Réseaux de flot associé avec $w = 3$.

FIG. 3.7: *Requête de multi-unicast et réseau de flot associé pour* $w = 3$.

Les idées 2) et 3) nous amènent à considérer un réseau de flot $\mathcal{N}$ avec w copies de G et capacité $k(e)$ sur chaque copie de l'arc e. Un faisceau de longueur d'onde i dans G se modélise par une unité de flot dans la i^{e} copie de G dans $\mathcal{N}$. Un ensemble de faisceaux utilisant w longueurs d'onde et $k(e)$ fibres sur chaque lien e peut donc se modéliser par un ensemble d'unités de flot respectant les capacités de $\mathcal{N}$.

Afin de trouver les chemins optiques entre les nœuds s et t dans G, nous ajoutons un nœud S, un arc de S vers chaque s_i, un nœud D_s_t, et un arc de chaque t_i vers D_s_t. À chaque chemin optique de s vers t nous pouvons associer une unité de flot de S vers D_s_t. La réciproque est dûe à l'idée 1). D'où le lemme suivant :

Lemme 12 *Transmettre d unités de flot de S à D_s_t dans $\mathcal{N}$ est équivalent à trouver d chemins optiques de s à t dans G.*

Étant donné une instance de multi-unicast enracinée en s, $\mathcal{N}$ est le réseau de flot où il y a un nœud D_s_t pour chaque $t \in M$ comme préalablement décrit, et un nœud S' relié à chaque nœud D_s_t par un arc de capacité $d(t)$, le nombre de chemins optiques à établir de s à t.

Théorème 13 *Étant donné un réseau G, une instance de multi-unicast M enracinée en s et les ressources du réseau (w et $k(e)$, $\forall\ e$), résoudre le* RWA *est équivalent à trouver un flot de taille $\sum_M d(t)$ depuis S vers S' dans le réseau auxiliaire $\mathcal{N}$.*

Preuve : Immédiat des chemins optiques vers le flot.

Le passage du flot aux chemins optiques est dû au fait que les seuls liens atteignant S' proviennent de D_s_t avec capacité $d(t)$, $t \in M$. En conséquence, s'il y a un flot de valeur $\sum_M d(t)$ de S vers S', alors exactement $d(t)$ unités de flot traversent le lien de D_s_t à S'. Donc, pour chaque $t \in M$, il y a $d(t)$ unités de flot de S à D_s_t. Le lemme 12 permet de conclure. □

Après quelques simplifications mineures, nous pouvons écrire le RWA comme l'ILP suivant :

Programme linéaire 10

Lois de Kirchoff :

$$\forall u \in V \backslash V_s,\ \forall \omega < w,\quad \sum_{\Gamma^+(u)} f_\omega(e) - \sum_{\Gamma^-(u)} f_\omega(e) = 0$$

$$\forall u \in M_s,\quad \sum_{\omega<w} Out_\omega(u) - d(u) = 0$$

$$\sum_{\omega<w} In_\omega - \sum_{u\in M_s} d(u) = 0$$

$$\forall u \in M_s,\ \forall \omega < w,\quad \sum_{\Gamma^+(u)} f_\omega(e) + Out_\omega(u) - \sum_{\Gamma^-(u)} f_\omega(e) = 0$$

$$\sum_{\Gamma^+(s)} f_\omega(e) - \sum_{\Gamma^-(s)} f_\omega(s) - In_\omega = 0$$

Contraintes de capacités :

$$\forall e \in E,\ \forall \omega < w,\quad f_\omega(e) \le k(e)$$

3.3.2 Cas général et multiflot

Nous nous intéressons maintenant aux instances générales $I = \bigcup_{s\in E} M_s$. Pour cela, nous étendons le réseau précédent, $\mathcal{N}$, en ajoutant un ensemble de sommets $\{S,\ D\_s\_t,\ S'\}$ pour chaque multi-unicast M_s. Le RWA est alors équivalent à trouver un flot entier de valeur $\sum_{t\in M_s} d(s,t)$ entre S et S' pour chaque $s \in E$. La figure 3.8 donne l'allure de $\mathcal{N}$dans le cas de deux sources différentes.

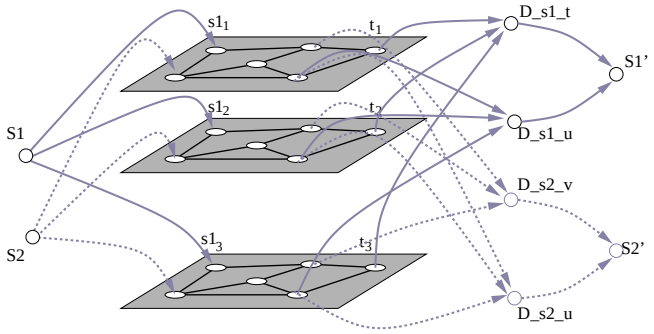


FIG. 3.8: *Le graphe auxiliaire pour 4 requêtes et 2 sources,* $w = 3$

Le programme linéaire final s'écrit avec $O(|V|.|E|.w)$ variables et $O(|V|^2.w)$ contraintes. C'est inférieur de $O(|V|^2)$ aux $O(|V|^2.|E|.w)$ et $(|V|^3.w)$ du programme linéaire de Krishnaswamy et Sivarajan (2001). Celui-ci ne prend pourtant pas en compte la multiplicité des fibres.

Cette réduction drastique de la taille de la formulation, obtenue par l'agrégation de l'information des requêtes du routage optique en multi-unicasts, a des conséquences fortes sur la capacité à résoudre le programme linéaire et nos algorithmes d'approximations fondés sur l'*arrondi aléatoire* de la relaxation linéaire que nous décrivons dans la section suivante. Cela nous permet notamment de manipuler des réseaux et des instances plus importants.

Nous montrons de plus dans la section suivante qu'il est facile d'intégrer à cette formulation le dimensionnement d'équipements de conversion partielle sans pour autant changer l'ordre de grandeur de la taille du programme linéaire.

3.3.3 Routage optique multifibre avec conversion

La définition d'un chemin optique dans un réseau optique sans conversion est très naturelle puisqu'il s'agit d'un chemin "classique" associé à une longueur d'onde. Dans le cas où des conversions sont possibles, la correspondance n'est plus aussi simple. En fait, lorsqu'un chemin optique traverse un convertisseur, la contrainte de conservation de longueur d'onde entre l'entrée du routeur et la sortie est relâchée. Dans le cas des conversions totales et partielles, elle l'est totalement. Cela revient à avoir coupé le chemin au niveau du routeur en deux parties indépendantes. Un chemin optique avec conversion

peut alors être considéré comme une suite de chemins optiques sans conversion allant d'un convertisseur à l'autre.

Définition 5 (Chemin optique w-coloré)

Soit un réseau $G = (V, E)$, le nombre w de longueurs d'onde par fibre et deux nœuds u et v de G.

Un chemin optique w-coloré de u à v dans G est une séquence P de $c_P + 1$ couples $(p_i, w_i)_{i \leq c_P}$ tels que : p_i est un chemin sur G, p_0 commence en u, p_{c_P} termine en v, et $\forall i \in [1...c_P]$, p_i commence là où p_{i-1} s'arrête ; w_i est la longueur d'onde du faisceau laser correspondant à p_i.

On dit que P est « converti en longueur d'onde *» (ou converti) à chaque nœud où termine un p_i, $i < c_P$. Aussi, c_P est le nombre de conversions effectuées sur P.*

Lorsque les routeurs sont équipés de convertisseurs partiels, la quantité importante est le nombre de conversions effectuées à chaque sommet et le problème du routage optique avec conversion partielle peut s'écrire comme suit :

« `Étant donné un réseau, une instance de communications, et un ensemble de ressources (fibres, longueurs d'onde, convertisseurs), trouver un chemin optique` w-`coloré pour chaque requête sous la contrainte que deux chemins ne peuvent pas utiliser la même longueur d'onde sur la même fibre.` »

L'écriture du problème de décision associé dépend alors de la façon dont les ressources sont comptées.

Définition 6 (Problème du routage optique)

Entrées : *Un réseau $G = (V, E)$, une instance I de requêtes sur G, le nombre w de longueurs d'onde, une fonction $k : E \to \mathbb{N}^*$ donnant le nombre de fibres, et soit (a) une fonction $c : V \to \mathbb{N}$ donnant le nombre de conversions disponibles à chaque nœud de G, soit (b) c_{max}, le nombre maximal de conversions autorisées dans G.*

Question : *Existe-t-il pour chaque requête $(x, y) \in I$ un chemin optique w-coloré de x à y tel que :*

1. *au plus $k(e)$ chemins optiques utilisent la même longueur d'onde sur un lien $e \in E$:*

$$\forall i, \forall e, |\{P \ni (e, i)\}| \leq k(e) \, ;$$

2. (a) *pour tout nœud $u \in V$, le nombre de conversions effectuées en u est au plus $c(u)$;*

   (b) *ou bien, le nombre total de conversions ($\sum_P c_P$) est au plus c_{max} ?*

Ces deux façons de compter les conversions amènent des versions très différentes du problème comme nous le montrons dans la remarque suivante.

Remarque 3.3.1 *Dans le cas (a) de la définition 6, où l'on tient compte précisément des conversions utilisées en chaque nœud, une solution optimale peut contenir un chemin w-coloré suivant un chemin non-simple dans le réseau, même dans le cas où tous les sous-chemins p_i sont contraints à suivre des trajets simples entre les convertisseurs. Dans l'exemple illustré en figure 3.9, lorsqu'une seule conversion par sommet est autorisée, une des trois connexions de F à C emprunte un chemin non-simple. La boucle entre D et G permet d'«aller chercher» la conversion disponible en G. Cette solution est une solution optimale du multiflot donné ensuite calculée par* CPLEX.

Une telle situation est évidemment absurde dans un réseau sans conversion ou lorsque l'on cherche à minimiser le nombre total de conversions dans le réseau (cas (b) de la définition 6) : puisque le placement d'une conversion ne change rien au compte final, les détours sont inutiles.

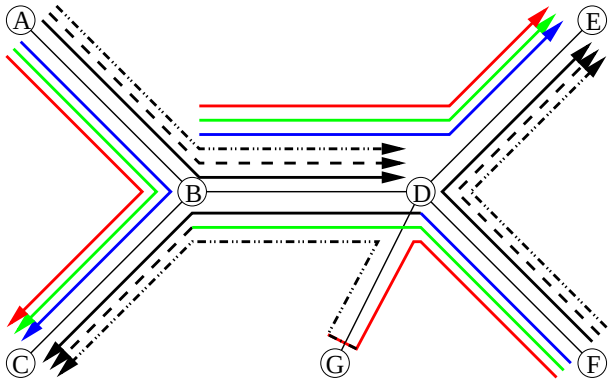


FIG. 3.9: *Chemin non-simple de F à C dans un réseau avec 1 fibre, 6 longueurs d'onde, 1 conversion par sommet.*

Dans le réseau de flot défini précédemment, où il y a une copie du réseau par longueur d'onde, le fait qu'un chemin optique w-coloré subisse une conversion partielle se traduit par le fait que l'unité de flot correspondante passe d'une copie du nœud où a lieu la

conversion à une autre. Il convient aussi de pouvoir mesurer le nombre de conversions ayant lieu à chaque sommet.

Afin de permettre ces passages d'une copie à l'autre, nous ajoutons au réseau de flot un *motif de conversion* pour tout nœud u de G. Ce motif est composé de 2 nœuds, $\texttt{trans}_u^+$ et $\texttt{trans}_u^-$, reliés par un arc de capacité $c(u)$, et $2w$ arcs sans capacité, 1 de chaque u_i vers $\texttt{trans}_u^+$ et 1 de $\texttt{trans}_u^-$ vers chaque u_i. Un tel motif de conversion est représenté sur le réseau dans la figure 3.10.

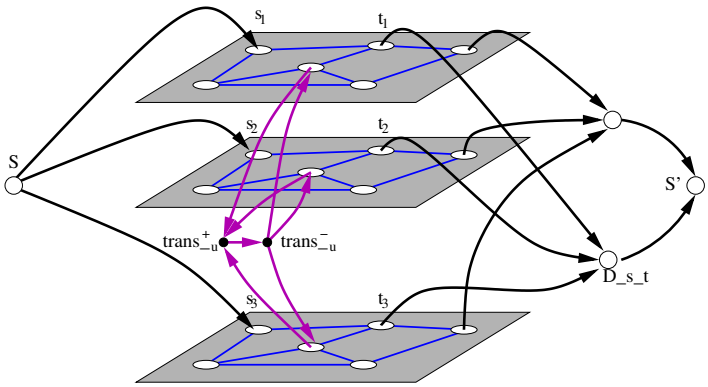


FIG. 3.10: *Routage optique avec conversion partielle. Un seul motif représenté.*

Lorsqu'un chemin optique est converti de la longueur d'onde w_1 à w_2 au nœud u, nous ajoutons à sa modélisation une unité de flot de u_{w_1} vers u_{w_2} via $\texttt{trans}_u^+$ et $\texttt{trans}_u^-$. Dès lors, si un ensemble de chemins optiques utilise moins de $c(u)$ conversions par nœud, alors il est modélisé par un ensemble d'unités de flot qui respecte les capacités de $\mathcal{N}$.

Ces motifs de conversion ajoutent $O(|V|^2 w)$ variables et $(|V|)$ contraintes supplémentaires au programme de multiflot de la section 3.3.2, ce qui ne change pas son ordre de grandeur.

Observations heuristiques : Il est intéressant de noter que lors de nos tests pratiques, la conversion n'a jamais été nécessaire pour obtenir une efficacité optimale de l'affectation de longueurs d'onde. Une seule fois une conversion est apparue nécessaire pour les algorithmes d'approximation que nous présentons au chapitre 5 lors de tests sur un réseau en anneau. Il semble probable que ce phénomène est dû à des propriétés parasites

de nos instances, puisque nous avons pu construire le cas présenté en figure 3.9 où des conversions sont nécessaires. Il est aussi possible que l'utilisation de plusieurs fibres par lien, qui peut s'interpréter comme une forme de conversion en longueurs d'onde dans un réseau monofibre (c.f. section 2.3.1), rende la présence de convertisseurs inutiles lors du dimensionnement d'un réseau. Ces équipements ne seraient donc utiles que dans le cas d'une remise à niveau d'un réseau existant conséquente à une augmentation du traffic et qui exclurait la modification des liens physiques. Ces deux hypothèses mériteraient d'être vérifiées ou invalidées.

Chapitre 4

Algorithmique pour la coloration fractionnaire de chemins

L'approche suivie dans ce chapitre est issue de l'interprétation, détaillée en section 4.1, du problème de coloration des sommets d'un graphe comme un problème de couverture des sommets par des stables. Il s'agit donc de choisir certains de ces stables de sorte que tout sommet appartienne à un stable choisi et en minimisant le nombre de ces ensembles.

Vue sous cet angle, la solution de la relaxation linéaire du multiflot utilisée par Kumar (1998) est en fait une *coloration fractionnaire* optimale des chemins de l'anneau dont une définition, directement transposée de celle de la coloration fractionnaire des sommets d'un graphe, est donnée en section 4.1. Bien que cette relaxation du problème de coloration soit non-approximable dans les graphes généraux (Lovász 1975), le résultat de Kumar (1998) montre qu'elle est calculable en temps polynomial lorsque le graphe est un anneau.

Notre travail s'intéresse au cas où le graphe est un arbre et les observations précédentes nous permettent de prouver les résultats suivants à propos des colorations fractionnaires ou entières de chemins[1].

Grâce à des arguments existentiels, nous montrons en section 4.1 que la coloration fractionnaire de chemins est polynomiale dans les arbres de degré borné. Nous rendons ensuite ce résultat constructif en section 4.2 en développant un algorithme efficace de calcul d'une coloration fractionnaire optimale. Pour cela, nous introduisons le concept de

[1]Ce travail a fait l'objet de la publication (Caragiannis, Ferreira, Kaklamanis, Pérennes, et Rivano 2001)

trace d'une coloration qui encode un aperçu *local et agrégé* de la coloration sur chacun des arcs de l'arbre. Notre algorithme se fonde sur la construction récursive d'un programme linéaire de taille polynomiale décrivant le polytope des traces possibles.

Cet algorithme de coloration fractionnaire pour les arbres peut être facilement adapté à tous les arbres de degré et largeur arborescente bornés comme décrit en section 4.2.5.

Nous montrons que le nombre chromatique fractionnaire vaut, au plus, $\frac{7l}{5}$ pour tout ensemble de chemins de charge l sur un arbre binaire en section 4.3. Ce résultat est étroitement lié à celui présenté par Auletta, Caragiannis, Kaklamanis, et Persiano (2000). En effet leur algorithme aléatoire peut être vu comme une tentative d'obtenir une coloration la plus proche possible d'une *coloration fractionnaire équilibrée*. De la même manière, une coloration fractionnaire équilibrée peut être considérée comme le centre de la distribution des solutions produites par leur algorithme. Toutefois, l'analyse de notre algorithme est beaucoup plus simple, notamment parce que notre algorithme est déterministe, et notre borne est plus serrée.

La section 4.4 s'intéresse à la coloration entière des chemins. En utilisant les techniques de Kumar (1998), nous donnons un algorithme aléatoire d'approximation de la coloration des chemins dans un arbre de degré borné, de facteur $1.61 + o(1)$. L'algorithme proposé opère un arrondi aléatoire sur la coloration fractionnaire produite par l'algorithme présenté en section 4.2. L'analyse du facteur d'approximation se fonde sur le fait que le nombre chromatique fractionnaire est une borne inférieure au nombre chromatique entier.

Dans la suite nous rappelons les définitions formelles des colorations (fractionnaires) de sommets et de chemins.

4.1 Coloration fractionnaire de chemins : définition par extension

Nous avons vu en section 1.4 que les problèmes de coloration de sommets et de chemins sont équivalents et qu'ils pouvaient s'écrire comme le programme linéaire en nombres entiers 1. Malheureusement nous avons aussi vu que la relaxation linéaire de ce programme

ne donne aucune indication sur les solutions entières, la raison principale étant que cette écriture place une grande partie de la structure du problème dans la binarité des variables. Nous avons aussi vu qu'une meilleure approche consiste en l'écriture du problème de coloration de graphe comme un problème de couverture des sommets par les stables, ce qui s'exprime comme le programme linéaire en nombre entier 2. Bien que cette écriture ait un nombre exponentiel de variables, elle est intéressante de par sa relaxation linéaire, la coloration fractionnaire dont le problème de séparation est le calcul du stable de poids maximum.

Le problème qui nous intéresse réellement est la coloration de chemins et non pas de sommets. Dans la section suivante nous utilisons l'équivalence entre ces deux problèmes exposée au chapitre 1 pour définir la *coloration fractionnaire de chemins* et transposer les résultats précédents. Nous utilisons aussi le lemme 5 (cf page 30) pour prouver que le calcul du *nombre chromatique fractionnaire de chemins* dans un arbre de degré borné peut se faire en temps polynomial.

Comme montré en section 1.4.3, la coloration de sommets d'un graphe et la coloration de chemins sont le même problème exprimé différemment. En utilisant la correspondance directe entre sommets et chemins faite dans la définition du graphe des conflits (définition 2), nous transposons dans la suite tous les résultats de la section précédente à la coloration de chemins.

Ainsi, étant donné un ensemble $\mathcal{P}$ de chemins dans un graphe $\mathcal{G}$, on définit un *stable de chemins* comme un ensemble de chemins deux à deux arc-disjoints, c'est à dire un ensemble de chemins dont les sommets correspondants forment un stable du graphe des conflits. La *coloration fractionnaire de chemins* se définit alors de manière analogue à celle des sommets.

Définition 7 *Étant donné un ensemble $\mathcal{P}$ de chemins sur un graphe $\mathcal{G}$, une coloration fractionnaire optimale de $\mathcal{P}$ est une couverture fractionnaire de coût minimal des chemins par des stables de chemins de coût unitaire.*

La coloration fractionnaire de chemins s'écrit de la même manière que celles des sommets comme le programme linéaire 3 et le lemme 5 s'applique toujours. On notera $\chi(\mathcal{G},\mathcal{P})$ ($\chi_f(\mathcal{G},\mathcal{P})$) le nombre chromatique (fractionnaire) des chemins de $\mathcal{P}$ dans $\mathcal{G}$.

On notera aussi $l(\mathcal{G}, \mathcal{P})$ la *charge* de $\mathcal{P}$ dans $\mathcal{G}$, c'est à dire le maximum sur les arcs de $\mathcal{G}$ du nombre de chemins de $\mathcal{P}$ qui passe sur un même arc.

Concernant la relation entre χ et χ_f, le résultat suivant est implicite dans le travail de Kumar (1998) qui s'intéresse au cas où $\mathcal{G}$ est C_n, l'anneau à n sommets[2] :

$$\chi(C_n, \mathcal{P}) \leq \chi_f(C_n, \mathcal{P}) + \frac{l(C_n, \mathcal{P})}{e} + o(l(C_n, \mathcal{P})).$$

Dans le cas particulier où le graphe des conflits est un graphe *arc-circulaire propre*, Niessen et Kind (1998) ont montré que $\chi = \lceil \chi_f \rceil$.

4.2 Construction d'un algorithme polynomial pour les arbres de degré borné

Cette section est consacrée au développement d'un algorithme polynomial construisant une coloration fractionnaire optimale d'un ensemble de chemins dans un arbre de degré borné. Notre algorithme consiste en une approche de type programmation dynamique où des colorations fractionnaires calculées sur deux sous-arbres sont fusionnées en une pour l'arbre complet.

Nous commençons par donner une preuve d'existence d'un tel algorithme.

4.2.1 Preuve existentielle

Dans la suite nous montrons qu'il existe un algorithme polynomial de calcul du nombre chromatique fractionnaire de chemins dans un arbre de degré borné. Ce résultat se fonde sur le lemme 5 et sur un algorithme de calcul du stable de chemins de poids maximum donné par Garg (1994).

Théorème 14 *Étant donné $\mathcal{P}$ un ensemble de chemins sur un arbre de degré borné $\mathcal{T}$, le calcul du nombre chromatique fractionnaire de $\mathcal{P}$ dans $\mathcal{T}$, $\chi_f(\mathcal{T}, \mathcal{P})$, peut se faire en temps polynomial.*

Preuve : Si l'on enracine $\mathcal{T}$ en un sommet r arbitrairement choisi, le stable de poids maximum de $\mathcal{P}$ dans $\mathcal{T}$ est donné par un appel à l'algorithme 1 proposé par Garg (1994).

[2]Ici e représente la base de l'exponentielle, 2.71828...

Algorithme 1 SPM de $\mathcal{P}$ dans un arbre $\mathcal{T}$

Entrée: $\mathcal{T}_i$ sous-arbre de $\mathcal{T}$ enraciné en i, d_i nombre de fils de i, r racine de $\mathcal{T}$

Entrée: Sur chaque lien, 2 chemins "*vides*" de longueur 1 : $p_\emptyset, q_\emptyset$

Sortie: $\forall i, p, q,\ I[i,p,q]$ SPM des chemins de $\mathcal{T}_i$ avec p, q touchant i en sens opposé.

Sortie: $w[i,p,q]$ poids de $I[i,p,q]$ *sans compter* $w(p)$ *ni* $w(q)$

1: **pour tout** i sommet dont les fils ont été traités **faire**

2: **pour tout** p, q chemins touchant i en sens opposé **faire**

3: soit $s_j,\ j = 1 \dots d_i$, le j^{eme} fils de i

4: $I[i,p,q] \leftarrow \emptyset$

5: $w[i,p,q] \leftarrow 0$

6: **pour tout** ensemble $P = \{p_j, q_j, j = 1 \dots d_i\}$ tel que $\forall j,\ p_j, q_j$ aillent entre i et s_j si p (resp. q) touche s_j alors $p_j = p$ (resp. $q_j = q$) **faire**

7: $w(P) \leftarrow \sum_{\rho \in P \setminus \{p,q\}} w(\rho)$ {p et q ne sont pas comptés}

8: **si** $w(P) + \sum_j w[j, p_j, q_j] > w[i,p,q]$ **alors**

9: $I[i,p,q] \leftarrow \cup_j I[j, p_j, q_j]$

10: $w[i,p,q] \leftarrow w(P) + \sum_j w[j, p_j, q_j]$

11: marquer i "traité"

12: **Return** $I[r, p_\emptyset, q_\emptyset], w[r, p_\emptyset, q_\emptyset]$

Cet algorithme de programmation dynamique calcule le stable de poids maximum de $\mathcal{P}$ dans $\mathcal{T}$ en temps polynomial dès que d, le degré de $\mathcal{T}$, est borné. En effet, l'algorithme commence par traiter les feuilles (les seuls sommets dont tous les fils sont traités au départ puis qu'elles n'en ont pas, ligne 1) et remonte vers la racine en ne passant qu'une fois par sommet. À chaque sommet, l'algorithme fait $O(l^{2d})$ accès aux tables I et w (lignes 6 à 10) dont la taille est $O(nl^2)$ (lignes 2 à 5).

Le lemme 5 s'applique alors et donne l'équivalence au sens de la réduction polynomiale entre le calcul de le stable de poids maximum des chemins et celui du nombre chromatique fractionnaire, donc le théorème. □

Bien que ce dernier résultat ne soit qu'existentiel, il indique qu'il est raisonnable de chercher un algorithme de coloration fractionnaire de chemins dans les arbres de degré borné. Le fonctionnement de l'algorithme de Garg (1994) est très similaire à celui de l'algorithme présenté dans la suite qui calcule une coloration fractionnaire en se fondant

sur un programme linéaire construit récursivement. La méthode utilisée pour construire ce programme est en effet comparable à une approche par programmation dynamique.

L'idée principale de l'algorithme que nous proposons est de construire un programme linéaire de taille raisonnable. Dans la suite nous montrons que deux colorations fractionnaires partielles des chemins dans deux sous-arbres peuvent être *fusionnées* en une pour l'arbre entier si et seulement si elles coïncident sur l'arc où elles se superposent. Cette propriété donne l'essentiel de la méthode de construction récursive du programme linéaire, mais n'élimine pas l'écueil d'un nombre exponentiel de configurations possibles des colorations fractionnaires sur un arc donné. Pour contourner cette difficulté, nous définissons un encodage agrégé du "*comportement*" d'une coloration fractionnaire sur un arc que nous appelons la *trace* de cette coloration.

4.2.2 Trace d'une coloration fractionnaire

Étant donné un arbre $\mathcal{T}$ et une coloration fractionnaire d'un ensemble $\mathcal{P}$ de chemins sur $\mathcal{T}$, cette dernière est une pondération $\bar{x}$ des stables de chemins de $\mathcal{P}$. Nous nommons "*trace de la coloration fractionnaire $\bar{x}$ sur l'arc e*" la fonction $X^e : \mathcal{P}^2 \to \mathbb{R}$ définie ci-dessous. Pour simplifier les notations, nous considérons des chemins "*virtuels*" ou vides sur un arc quelconque que nous notons $p_\emptyset$ ou $q_\emptyset$ selon les cas et définissons les ensembles de stables contenant certains chemins comme suit :

1. Pour tout chemin p, $\mathcal{I}(p)$ est l'ensemble des stables auxquels p appartient, et $\mathcal{I}(p,q) \;=\; \mathcal{I}(p) \cap \mathcal{I}(q)$.
2. Pour tout chemin p passant par l'arc e, $\mathcal{I}^e(p, q_\emptyset)$ est l'ensemble des stables où seul p passe par e.
3. $\mathcal{I}^e(p_\emptyset, q_\emptyset)$ est l'ensemble des stables ne contenant aucun chemin passant par e.

Pour toute paire (p, q) de chemins passant par e en sens opposé, p et q pouvant être les chemins "virtuels", la trace de $\bar{x}$ sur e vaut $X^e(p,q) = \sum_{I \in \mathcal{I}(p,q)} \bar{x}(I)$.

La fonction de trace encode un aperçu *local* et *agrégé* de la coloration fractionnaire. Local parce que seuls les chemins traversant l'arc considéré sont pris en compte. Par conséquent, deux colorations peuvent avoir la même trace si leurs différences n'affectent

pas la coloration de ces chemins. Agrégé enfin, puisqu'il ne s'agit que de sommes partielles sur les poids des stables. Là encore deux colorations différentes peuvent avoir des traces identiques. Ces deux caractéristiques induisent une perte d'information par rapport à la pondération des stables, ce qui permet d'avoir un codage de cette information agrégée qui soit polynômial en la charge. Nous verrons toutefois dans la suite que cette perte d'information n'est pas critique pour l'obtention d'une coloration fractionnaire optimale.

La *trace* $\mathbb{X}$ *d'une coloration fractionnaire* $\bar{x}$ est l'ensemble des fonctions de traces sur tous les arcs de $\mathcal{T}$: $\mathbb{X} = \{X^e, \forall e \in T\}$.

Étant donné une instance du problème de coloration fractionnaire de chemins $(\mathcal{T}, \mathcal{P})$ et une constante c, on notera dans la suite $Tr_c(\mathcal{T}, \mathcal{P})$ l'ensemble de toutes les colorations fractionnaires de $(\mathcal{T}, \mathcal{P})$ de coût au plus c.

4.2.3 Fusion et fission

Notre algorithme construit récursivement un programme linéaire de taille polynomiale dont la solution permet d'obtenir une coloration fractionnaire optimale. Notre induction se fonde sur des procédure de fission (algorithme 2) et de fusion (algorithme 3) d'une instance de coloration fractionnaire. Ces procédures permettent de construire toute instance de coloration fractionnaire de chemins à partir d'une collection d'instances sur des étoiles en les fusionnant pas à pas, ou de faire le chemin inverse par des séries de fissions. Rappelons qu'une étoile est un arbre de hauteur 1, c'est-à-dire tel qu'un seul sommet a un degré supérieur à 1.

Algorithme 2 FISSION de $(\mathcal{T}, \mathcal{P})$ sur e en $(\mathcal{T}_1, \mathcal{P}_1)$ et $(\mathcal{T}_2, \mathcal{P}_2)$

Entrée: $e = \{u_1, u_2\}$ un arc non-terminal de $\mathcal{T}$

$U_i \leftarrow$ les composantes connexes de $\mathcal{T} \setminus e \ni u_i,\ i = 1, 2$

$\mathcal{T}_i \leftarrow U_i \cup \{u_j, e\},\ i = 1, 2,\ j \neq i$

pour tout $p \in \mathcal{P}$ **faire**

si $\exists i,\ p \in U_i$ **alors**

$\mathcal{P}_i \leftarrow p$

sinon si $\exists i,\ p = (a \rightsquigarrow u_i \rightarrow u_j \rightsquigarrow b),\ a \in U_i, b \in U_j,\ j \neq i$ **alors**

$\mathcal{P}_i \leftarrow p = (a \rightsquigarrow u_j)$

$\mathcal{P}_j \leftarrow p = (u_i \rightsquigarrow b)$

Si l'on se donne une coloration fractionnaire de $(\mathcal{T}, \mathcal{P})$ où $\mathcal{T}$ n'est pas une étoile et un arc non-terminal $e = (u, v)$ de $\mathcal{T}$, c'est-à-dire un arc tel que ni u ni v n'est une feuille. La fission de $\mathcal{T}$ sur l'arc e induit deux instances plus petites de coloration fractionnaire, notées $(\mathcal{T}_1, \mathcal{P}_1)$ et $(\mathcal{T}_2, \mathcal{P}_2)$, comme illustré par la figure 4.1. Notons que la procédure de fission de toute coloration fractionnaire de $\mathcal{T}$ de coût c induit des colorations fractionnaires de $(\mathcal{T}_1, \mathcal{P}_1)$ et $(\mathcal{T}_2, \mathcal{P}_2)$ de coût également inférieur à c. Par construction de la procédure de fission, l'algorithme 2, les deux colorations induites ont la même trace sur e.

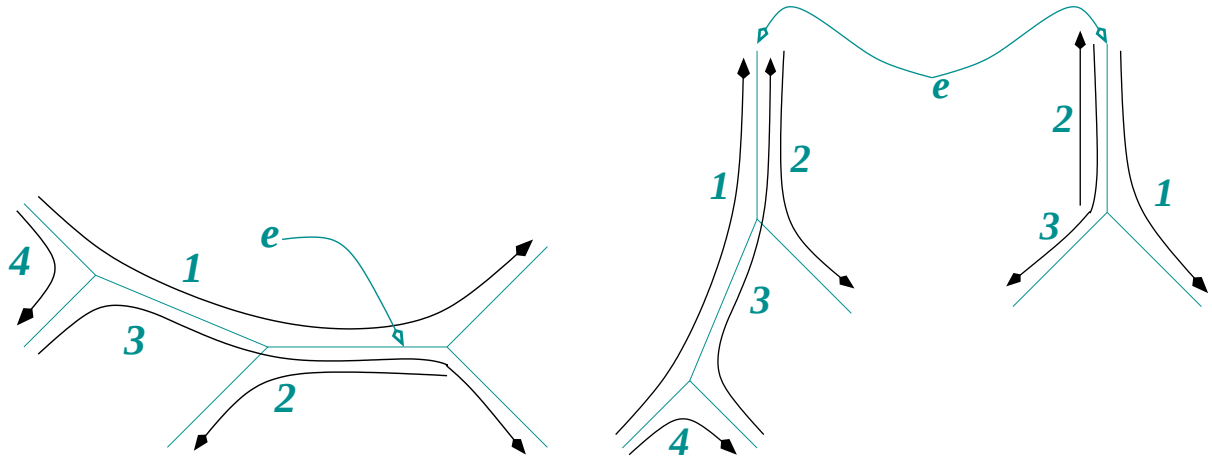


FIG. 4.1: *Fission d'un arbre sur l'arc e.*

Algorithme 3 FUSION $\bar{x}_1 \in Tr_c(\mathcal{T}_1, \mathcal{P}_1),\ \bar{x}_2 \in Tr_c(\mathcal{T}_2, \mathcal{P}_2)$ into $\bar{x} \in Tr_c(\mathcal{T}, \mathcal{P})$

Entrée: Égalité des traces sur l'arc $e = \{u_1, u_2\} : X_1^e = X_2^e$

Sortie: $X^e = X_1^e = X_2^e$

$\forall\ I \in \mathcal{I},\ \bar{x}(I) \leftarrow 0$

tant que $\exists p, q \in \mathcal{P}_1 \cup \{p_\emptyset, q_\emptyset\} | X_1^e(p,q) \neq 0$ **faire**

$\{$*L'égalité des traces assure que* $p, q \in \mathcal{P}_2$ *et* $X_2^e(p,q) > 0\}$

choisir $I_1 \in \mathcal{I}_1(p,q)$ tel que $\bar{x}_1(I_1) > 0$

choisir $I_2 \in \mathcal{I}_2(p,q)$ tel que $\bar{x}_2(I_2) > 0$

$\bar{x}_{min} \leftarrow \min(\bar{x}_1(I_1), \bar{x}_2(I_2))$

$I \leftarrow I_1 \setminus \{p,q\}_{\mathcal{P}_1} \cup I_2 \setminus \{p,q\}_{\mathcal{P}_2} \cup \{p,q\}_{\mathcal{P}}$ $\{$*Par construction,* $I \in \mathcal{I}\}$

augmenter $\bar{x}(I)$ de $\bar{x}_{min}$

diminuer $\bar{x}_i(I_i)$ de $\bar{x}_{min}$ $\{$*Assure l'égalité des traces*$\}$

Le fait que le coût des colorations issues d'une fission sont au plus le coût de l'instance de départ donne l'intuition d'une propriété de stabilité des ensembles des colorations fractionnaires de coût inférieur à une constante. De fait, la procédure de fission est inversée par celle de fusion, l'algorithme 3. Cette procédure prend deux instances $(\mathcal{T}_1, \mathcal{P}_1)$ et $(\mathcal{T}_2, \mathcal{P}_2)$ qui pourraient être issues de la fission d'un arbre $\mathcal{T}$, et une coloration fractionnaire de chaque instance, $\bar{x}_1$ et $\bar{x}_2$. Si les deux colorations ont la même trace sur e, alors la procédure de fusion construit une coloration fractionnaire de l'arbre global qui a la même trace sur e que les deux colorations partielles et qui a pour coût le maximum des coûts de ces colorations.

Il est à noter que la seule contrainte nécessaire pour que deux instances puissent être fusionnées est qu'elles "*pourraient être issues d'une fission*". En pratique cela signifie seulement qu'elles ont la même trace sur l'arc sur lequel s'opère la fusion.

L'idée de la construction récursive de la coloration fractionnaire est maintenant explicite : il s'agit d'opérer des fissions sur chaque arc pour obtenir des instances irréductibles par cette opération, de calculer des colorations fractionnaires sur ces cas de base puis de remonter à l'instance de départ par des fusions successives. La proposition suivante prouve que cette reconstruction progressive permet de décrire l'ensemble des colorations

fractionnaires de coût inférieur à une constante. Ce résultat valide la remarque faite plus haut : la fonction de trace est une quantité d'informations suffisante pour obtenir une coloration fractionnaire optimale.

Proposition 15 *L'ensemble de toutes les fusions possibles d'éléments de* $Tr_c(\mathcal{T}_1, \mathcal{P}_1)$ *et de* $Tr_c(\mathcal{T}_2, \mathcal{P}_2)$ *est égal à* $Tr_c(\mathcal{T}, \mathcal{P})$.

Preuve : Étant donné deux colorations fractionnaires $\bar{x}_1$ et $\bar{x}_2$ appartenant respectivement à $Tr_c(\mathcal{T}_1, \mathcal{P}_1)$ et $Tr_c(\mathcal{T}_2, \mathcal{P}_2)$ et ayant la même trace $X_1^e = X_2^e$ sur e, leur fusion par l'algorithme 3 produit une coloration $\bar{x}$ qui appartient à $Tr_c(\mathcal{T}, \mathcal{P})$. Pour vérifier cette affirmation, il suffit de noter que l'algorithme 3 maintient les invariants suivants à chaque étape :

1. Les chemins sont couverts par $\bar{x}_1$, $\bar{x}_2$ ou $\bar{x}$;
2. $\bar{x}(\mathcal{P}) + \bar{x}_1(\mathcal{P}_1) = \bar{x}(T, \mathcal{P}) + \bar{x}_2(\mathcal{P}_2) = c$.

Il s'ensuit qu'à la fin de la boucle principale $\bar{x}$ est une coloration fractionnaire de $(\mathcal{T}, \mathcal{P})$ de coût au plus c. De plus, la trace de $\bar{x}$ sur tout arc de $\mathcal{T}$ est égale à la trace de la coloration fractionnaire $\bar{x}_i$ concernée. Par conséquent l'ensemble des toutes les fusions possibles d'éléments de $Tr_c(\mathcal{T}_1, \mathcal{P}_1)$ et $Tr_c(\mathcal{T}_2, \mathcal{P}_2)$ est inclus dans $Tr_c(\mathcal{T}, \mathcal{P})$.

Enfin, comme il a été dit plus haut, la fission de tout élément $Tr_c(\mathcal{T}, \mathcal{P})$ induit deux éléments de $Tr_c(\mathcal{T}_1, \mathcal{P}_1)$ et $Tr_c(\mathcal{T}_2, \mathcal{P}_2)$ qui peuvent être fusionnés en un élément de $Tr_c(\mathcal{T}, \mathcal{P})$ (celui de départ).

Ces deux inclusions réciproques donnent le résultat. □

Corollaire 16 *Il est possible de calculer un programme linéaire de taille polynômiale dont les solutions sont les traces des éléments de* $Tr_c(\mathcal{T}, \mathcal{P})$.

Preuve : Supposons tout d'abord qu'un tel programme linéaire existe pour les étoiles de degré borné d. Nous utilisons ensuite un algorithme récursif pour générer un programme linéaire dont les solutions sont les traces des éléments de $Tr_c(\mathcal{T}, \mathcal{P})$.

Soient $(\mathcal{T}_1, \mathcal{P}_1)$ et $(\mathcal{T}_2, \mathcal{P}_2)$ le résultat de la fission de $(\mathcal{T}, \mathcal{P})$ sur un arc e. Supposons par récurrence que S_1 et S_2 sont des programmes linéaires décrivant $Tr_c(\mathcal{T}_1, \mathcal{P}_1)$ et $Tr_c(\mathcal{T}_2, \mathcal{P}_2)$. On note $X_1^e(p, q)$ et $X_2^e(p, q)$ les variables de traces des deux sous-instances sur e. Comme

le montre la proposition 15, la seule contrainte pour qu'une coloration de $(\mathcal{T}_1, \mathcal{P}_1)$ puisse être fusionnée à une de $(\mathcal{T}_2, \mathcal{P}_2)$ est que les fonctions de traces X_1^e et X_2^e soient égales.

Le programme linéaire suivant décrit donc les solutions de $Tr_c(\mathcal{T}, \mathcal{P})$:

$$S = S_1 \cup S_2 \cup \{X_1^e(p,q) = X_2^e(p,q) \mid \forall p, q\}$$

La récurrence et le fait qu'une fusion par arc de $\mathcal{T}$ suffise montrent que dans S il y a $O(l^2)$ contraintes d'égalité de trace par arc de $\mathcal{T}$ lorsque la charge est l. Il y a ensuite un programme par cas de base, c'est à dire par étoile centrée en un sommet non-terminal de $\mathcal{T}$. Si s est une borne supérieure sur la taille d'un tel programme, la taille du programme linéaire final est donc $O\left(n(l^2+s)\right)$ où n est le nombre de sommets de $\mathcal{T}$. Le programme linéaire 11 est une écriture synthétique du résultat de cet algorithme.

Programme linéaire 11 (traces des colorations fractionnaires)

$$\begin{array}{lrcl} \textit{Programmes pour les étoiles} & S(v) & & \forall\ v \textit{ un sommet} \\ \textit{Égalité des traces} & X_u^e(p,q) & = & X_v^e(p,q) \quad \forall\ e=(u,v) \textit{ un arc non-terminal} \\ & & & \forall\ p, q \textit{ chemins passant par } e \textit{ en sens opposé} \end{array}$$

X_u^e désigne ici la fonction de trace sur l'arc e d'une coloration fractionnaire de l'étoile centrée en u. Une analyse globale du programme linéaire 11 montre que les programmes pour chaque étoile définissent les variables de trace conformément aux comportements locaux des chemins tandis que les contraintes d'égalité de trace contraignent ces variables aux dépendances globales. Il ne reste plus qu'à exprimer les programmes linéaires pour les cas de base de la récursion, les étoiles de degré d borné, notées d-étoiles.

Une approche naïve pour obtenir un programme linéaire de taille polynômiale décrivant les traces de colorations fractionnaires des d-étoiles consiste à prendre une variable par stable de chemins. Il y en a un nombre polynômial comme nous le montrons ci-dessous et ils permettent trivialement d'exprimer les variables de trace au vu de leur définition.

Étudions tout d'abord le cas où chaque chemin est présent une et une seule fois. Nous appellerons une telle instance une *d-étoile canonique*. Un ensemble de chemins chargeant chaque arc exactement une fois est un stable de chemins maximal pour l'inclusion que nous appelons *stable canonique*. L'ensemble de ces ensembles canoniques est isomorphe à

l'ensemble des couplages parfaits dans $K_{d,d}$, le graphe biparti complet dont chaque partie comporte d éléments. En effet, un arc dans le couplage reliant les i^{eme} sommet de gauche au j^{eme} sommet de droite est isomorphe au choix du chemin passant sur le i^{eme} arc, par le centre de l'étoile, puis sur le j^{eme} arc si $i \neq j$. Si $i = j$, le couplage (i, i) est isomorphe au choix des deux chemins de longueur 1 présents sur le i^{eme} arc. Un exemple d'une telle correspondance est illustré en figure 4.2 pour $d = 3$. Notons $M(d)$ le nombre de ces stables canoniques. $M(d)$ ne dépend que de d et est donc borné dès que d l'est.

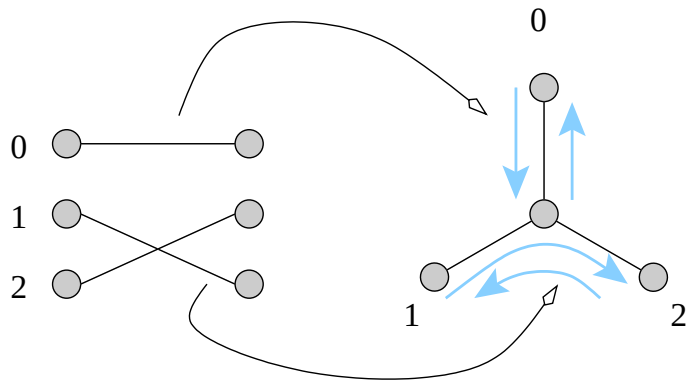


FIG. 4.2: *Correspondance entre un couplage et un stable canonique.*

De retour à une d-étoile générale, si la charge de l'ensemble des chemins est l, il y a au plus $(l+1)^{2d}$ stables qui soient isomorphes au même stable canonique : il y a au plus un chemin à choisir sur chaque arc et dans chaque direction et si la charge est l, le choix se fait, pour chaque arc et direction, entre un chemin parmi au plus l et laisser l'arc non chargé, ce qui revient à choisir le chemin vide $p_\emptyset$ défini plus haut. Comme tout stable est isomorphe à un ensemble canonique, quitte à prendre plusieurs chemins vides, le programme linéaire a au plus $O(M(d)l^{2d})$ variables.

Globalement, le programme linéaire 11 décrivant $Tr_c(\mathcal{T}, \mathcal{P})$ a $O(nM(d)l^{2d})$ variables et $O(nl^2)$ contraintes, ce qui est assurément polynômial dès que d est borné. □

Comme il est dit dans la preuve, le système linéaire proposé pour les d-étoiles est naïf et sans intelligence. La section suivante est consacrée à l'amélioration de ce système en vue de le rendre plus compact. Pour l'instant nous montrons que le programme linéaire décrit précédemment permet de résoudre la coloration fractionnaire.

Proposition 17 *La coloration fractionnaire de chemin dans les arbres de degré borné d se réduit à résoudre un programme linéaire de taille polynômiale.*

Preuve : Le corollaire 16 montre que calculer la trace d'un élément de $Tr_c(\mathcal{T}, \mathcal{P})$ se fait en résolvant un programme linéaire de taille polynômiale. En jouant sur c par dichotomie, il est facile de trouver le coût optimal d'une coloration et de calculer la trace d'une coloration optimale.

Il ne reste plus qu'à voir comment construire une coloration fractionnaire de $Tr_c(\mathcal{T}, \mathcal{P})$ à partir de sa trace $\mathbb{X}$, calculée grâce au corollaire 16. Une fois encore le processus est inductif. Pour chaque d-étoile, on calcule une coloration fractionnaire de trace celle induite par $\mathbb{X}$. Cette phase est immédiate puisque les relations entre les variables de trace et la coloration fractionnaire sont triviales pour une étoile. Ensuite, la proposition 15 montre comment fusionner deux colorations sur deux sous-arbres coïncidant sur un arc sous contrainte qu'elles aient la même trace. Cette dernière propriété est acquise puisque les colorations partielles que l'on manipule ont toutes des traces induites par $\mathbb{X}$. Après une fusion par arc, la coloration fractionnaire de $(\mathcal{T}, \mathcal{P})$ est construite. □

4.2.4 Réduction de la taille du problème

Le système linéaire pour les d-étoiles proposé dans la preuve du corollaire 16 est naïf et a une taille qui explose rapidement avec d : le nombre de variables est $O(nl^6)$ pour les arbres binaires et $O(nl^8)$ pour les arbres ternaires. De telles tailles rendent impossible le traitement d'instances de forte charge. Dans la suite nous montrons comment traiter le cas des d-étoiles plus intelligemment afin de réduire la taille finale du programme linéaire à $O(dM(d)l^2n)$, c'est à dire $O(nl^2)$ pour un degré borné. Les constantes multiplicatives restent grandes mais n'empêchent pas de considérer de fortes charges.

Proposition 18 *La coloration fractionnaire de chemin dans un arbre de degré borné d se réduit à résoudre un programme linéaire de taille $O(dM(d)l^2n)$.*

Preuve : Afin d'obtenir ce résultat et au vu de la proposition 17 et de la preuve du corollaire 16, il suffit de montrer qu'il est possible de décrire l'ensemble des traces des colorations fractionnaires d'une d-étoile avec un système linéaire de taille $O(dM(d)l^2)$.

Considérons donc une d-étoile $\mathcal{S}$ et supposons, sans perte de généralité, que l'ensemble de chemins la charge uniformément. Si ce n'est pas le cas, il suffit d'ajouter des chemins de longueur un sur tous les arcs sous chargés, cela ne diminuera pas la taille du système linéaire.

Comme il est montré dans la preuve du corollaire 16, les stables canoniques de $\mathcal{S}$ sont isomorphes aux couplages parfaits de $K_{d,d}$. L'idée est donc d'écrire le polytope des couplages de ce graphe biparti par des équations de flot à la manière de König (1931).

On note v_c le centre de $\mathcal{S}$, $v_0, \ldots, v_{d-1}$ ses fils et $\mathcal{M}$ l'ensemble des stables canoniques de chemins sur $\mathcal{S}$. Rappelons qu'un stable canonique est un ensemble de chemins chargeant chaque arc de $\mathcal{S}$ exactement une fois. Pour chaque ensemble $M \in \mathcal{M}$, nous construisons un réseau auxiliaire de flot F_M comme suit.

- Pour tout chemin $p \in \mathcal{P}$ isomorphe a un chemin de M, on ajoute un sommet $V(p)$ à F_M.
- Pour toute paire de sommets $V(p), V(q)$ telle que $\exists\ i \in [1, \ldots, d-1] \mid (v_c, v_i) \subseteq p$ et $(v_i, v_c) \subseteq q$, on ajoute l'arc $(V(p), V(q))$ à F_M.
- On définit la fonction de flot f_M (c'est à dire qui respecte des contraintes de conservation aux sommets) sur les arcs de F_M. Celle-ci induit un flot f'_M sur les sommets. Notons que puisqu'aucune source et aucun puits n'est présent, la fonction de flot f_M est une circulation.
- Pour chaque feuille v_i de $\mathcal{S}$, on ajoute à F_M la contrainte suivante sur f'_M.

$$\sum_{p \mid (v_c, v_i) \subseteq p} f'_M(V(p)) = c_M.$$

  c_M représente la somme des poids des stables isomorphes à M dans la coloration fractionnaire. C'est une première agrégation de l'information – au lieu d'un poids par stable on a un poids par classe – mais il est toujours possible de retrouver la coloration fractionnaire en décomposant le flot.

Toute solution au flot précédent induit une couverture des chemins $f'_M(p) = f'_M(V(p))$ et des sommes partielles des variables de trace $f_M(p, q) = f_M((V(p), V(q)))$.

Le système linéaire complet d'une d-étoile est donc le suivant.

- Les flots $F_M, M \in \mathcal{M}$: pris tous ensemble ;
- $X(p,q) = \sum_{M \in \mathcal{M}} f_M(p,q)$: les variables de trace ;
- $\sum_{M \in \mathcal{M}} f'_M(p) \geq 1$: les contraintes de couverture des chemins ;
- $\sum_{M \in \mathcal{M}} c_M \leq c$: la contrainte de coût.

Les propriétés du polytope des couplages de $K_{d,d}$ dues à König (1931), et l'isomorphisme entre ces couplages et les stables détaillé dans la preuve du corollaire 16 prouvent que ce système linéaire décrit la coloration fractionnaire de $\mathcal{S}$ par un codage des variables de trace.

La taille du système pour chaque stable canonique est $O(l^2 d)$. Il y en a $|\mathcal{M}| = M(d)$, d'où le résultat. □

Ce résultat permet de ne plus vraiment contraindre le degré à être borné. En effet, puisque $M(d) = O(d!)$, il suffit que $d = O(\max\{\log l / \log\log l, \log n / \log\log n\})$. Dans la suite nous montrons que le même raisonnement peut s'appliquer lorsque le graphe est de degré et largeur arborescente bornés.

4.2.5 Extension à la largeur arborescente bornée

De nombreux de résultats sur les arbres peuvent s'étendre aux graphes de largeur arborescente bornée. Dans le cas qui nous intéresse, l'algorithme suit la même trame, seuls des détails changent dans les opérations de fusion et de fission et dans la définition de la trace. La proposition suivante montre donc qu'il existe un algorithme polynomial fondé sur la résolution d'un programme linéaire. Ce résultat est toutefois plus existentiel que pratique car la taille d'un tel programme est grande et sa construction nécessite des décompositions du graphe très coûteuses (Bouchitté et Todinca 2001).

Proposition 19 *La coloration fractionnaire de chemins se résout en temps polynômial dans les graphes de degré et largeur arborescente bornés.*

Preuve : La preuve suit la même structure que dans le cas des arbres de degré borné. Les différences principales se situent au niveau de la définition de la trace et des opération de fission et de fusion. En effet, là où la séparation entre deux sous-instances se fait sur un arc de l'arbre, elle se fait sur une coupe du graphe de largeur arborescente bornée.

La largeur arborescente étant bornée, on peut ne considérer que des coupes dues aux séparateurs minimaux du graphe qui sont sujettes à la même borne k. La fonction de trace devant prendre en compte les k arcs de la coupe, il y aura donc $O(l^{2k})$ variables de trace par coupe si la charge est l.

De même les opérations de fission et de fusion s'appliquent à ces coupes, mais cela ne change pas grand chose aux algorithmes qui restent de toutes façons polynômiaux. □

On peut prouver de la même manière que la coloration fractionnaire de chemins est polynomiale sur les arbres d'anneaux.

L'existence de ces algorithmes permet de calculer des colorations fractionnaires optimales. L'intérêt, par exemple, de connaître le nombre chromatique fractionnaire d'une instance est qu'il donne une borne inférieure au nombre chromatique entier beaucoup plus fine que la charge de l'ensemble des chemins. La coloration fractionnaire permet aussi de guider la construction d'une bonne coloration entière, comme nous le montrons plus loin. Il est cependant intéressant d'estimer *a priori* le coût d'une coloration fractionnaire en fonction de la charge de l'ensemble des chemins. Cette estimation est difficile à opérer en l'état, particulièrement à cause de l'utilisation de la programmation linéaire dans le cœur de l'algorithme. La section suivante présente un travail visant à donner une borne supérieure sur le coût d'une coloration fractionnaire dans le cas des arbres binaires en étudiant une coloration particulière, dite *équilibrée*.

4.3 Étude de cas : les arbres binaires

Estimer *a priori* le coût d'une coloration fractionnaire optimale sur un arbre de degré borné quelconque est difficile pour plusieurs raisons. D'une part, il faudrait être en capacité d'isoler un nombre restreint de paramètres combinatoires, à la fois significatifs pour le coût d'une coloration fractionnaire, et simples à calculer. D'autre part, il faudrait avoir l'intuition du comportement d'un algorithme de résolution du programme linéaire 11.

L'approche choisie ici est autre. Afin de pouvoir faire une analyse complète, nous considérons le cas des arbres binaires. Nous définissons une coloration fractionnaire particulière qui a une structure combinatoire suffisamment explicite pour être, elle aussi,

analysable. Il est à noter que cette coloration n'est pas réservée aux arbres binaires, seule l'étude se restreint à ce cas particulier. Cette structure, que nous imposons à la coloration, est assez naturelle, puisqu'elle consiste à répartir équitablement le poids de la couverture d'un chemin entre toutes les possibilités. Cette approche, dite *équilibrée*, est à rapprocher d'une construction entière probabiliste : plutôt que de prendre le risque de faire des mauvais choix, nous n'en faisons aucun. D'ailleurs nous verrons, dans l'application au cas particulier des arbres binaires, que les résultats obtenus sont similaires à ceux de Auletta et coll. (2000). Leur algorithme construit une coloration entière en suivant une stratégie aléatoire comparable. Enfin, comme la coloration étudiée est un cas particulier, nous sommes assurés qu'une borne supérieure sur le coût d'une coloration équilibrée en est une sur le nombre chromatique fractionnaire.

4.3.1 Coloration fractionnaire équilibrée

La coloration fractionnaire *équilibrée* consiste donc à répartir uniformément, localement, la couverture d'un chemin entre tous les choix qui s'offrent à lui. Une conséquence globale de cette approche est que tous les chemins sont traités de la même manière. Étant donné que la construction d'une coloration fractionnaire est centrée sur la notion de trace, il n'est pas surprenant que la notion de *répartition équilibrée* le soit elle aussi. Une coloration fractionnaire sera donc dite *équilibrée* si la trace sur un arc ne dépend que du nombre de chemins traversant cet arc.

Définition 8 *Étant donnée une constante $w \in [0,1]$, une coloration* $w-$équilibrée *d'un ensemble de chemin est une coloration dont la fonction de trace sur tout arc e vaut $\frac{w}{l(e)}$ pour toute paire de chemins passant par e en sens opposé. C'est à dire,*

$$X^e(p,q) = \frac{w}{l(e)},\ X^e(p,q_\emptyset) = 1-w,\ \forall e,\ \forall p,q \text{ chemins passant par } e \text{ en sens opposé.}$$

La définition de la coloration fractionnaire équilibrée à partir des traces permet d'affirmer que toute instance admet une telle coloration et simplifie extrêmement son analyse.

Corollaire 20 *Étant donné une constante $w \in [0,1]$, tout ensemble de chemins $\mathcal{P}$ sur un arbre $\mathcal{T}$ admet une coloration $w-$équilibrée qui s'obtient en fusionnant les colorations $w-$équilibrées de toutes les instances induites sur les étoiles calculées* indépendamment.

Preuve : Supposons que l'on ait les colorations $w-$équilibrées de toutes instances induites par $(\mathcal{T}, \mathcal{P})$ sur les étoiles composant $\mathcal{T}$, calculées indépendamment au préalable. La définition d'une coloration $w-$équilibrée impose que l'égalité des traces est respectée puisque les fonctions de trace sur chaque arc ne dépendent que de la propriété locale qu'est la charge de cet arc. Ainsi les conditions de la proposition 15 sont réunies, et l'algorithme donné par la proposition 17 permet d'obtenir une coloration fractionnaire globale qui soit consistante avec les colorations locales en opérant une fusion sur chaque arc de $\mathcal{T}$. Par définition cette coloration est équilibrée.

Par ailleurs, la proposition 15 montre que le coût d'une telle coloration $w-$équilibrée est le maximum des coûts des colorations locales □

Ce dernier résultat nous permet d'affirmer que pour borner le coût de la coloration $w-$équilibrée d'un arbre, il suffit de borner celui de la coloration $w-$équilibrée d'une étoile. La section suivante est consacrée à l'analyse, simple mais exhaustive, de la coloration $w-$équilibrée des $3-$étoiles. Cette analyse montre qu'il existe une coloration équilibrée des $3-$étoiles de coût au plus $\frac{7}{5}l$, ce qui amène à la proposition suivante.

Proposition 21 *Pour tout ensemble de chemins $\mathcal{P}$ de charge l sur un arbre binaire $\mathcal{T}$, il existe une coloration fractionnaire de coût au plus $\frac{7l}{5}$. De plus, cette coloration peut être calculée en temps polynomial.*

Preuve : La proposition 23, démontrée en section 4.3.2, montre qu'une coloration $\frac{4}{5}-$équilibrée de tout ensemble de chemins de charge l sur une $3-$étoile coûte toujours moins de $\frac{7l}{5}$. Cette coloration particulière des $3-$étoiles est décrite dans la section suivante. Le corollaire 20 termine la preuve. □

Un corollaire direct de ce résultat est que pour tout ensemble de chemins $\mathcal{P}$ de charge l sur un arbre binaire $\mathcal{T}$, le stable maximum de chemins de $\mathcal{P}$ est de taille au moins $\frac{5|\mathcal{P}|}{7l(\mathcal{T},\mathcal{P})}$.

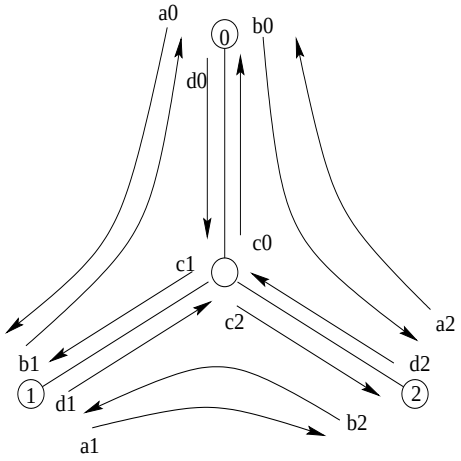
4.3.2 Coloration fractionnaire équilibrée des $3-$étoiles

L'objet de cette section est d'étudier la coloration équilibrée des $3-$étoiles. Afin de surestimer le plus finement possible le coût d'une telle coloration, nous opérons une

analyse de cas exhaustive des colorations $w-$équilibrées des $3-$*étoiles réduites* que nous définissons dans la suite. Cette étude de cas nous permettra ensuite de dériver une formule analytique du coût de la coloration en fonction de quelques paramètres décrivant une instance.

$3-$étoile réduite

Un ensemble de chemins générique sur une $3-$étoile nécessite 12 paramètres pour être complètement décrit comme le montre la figure 4.3. C'est un nombre bien trop important pour permettre une étude de cas exhaustive. Pour rendre une telle étude réalisable, nous utilisons une réduction classique qu'utilisent aussi Auletta et coll. (2000) et qui transforme toute instance sur une $3-$étoile en une $3-$*étoile réduite* qui se décrit avec seulement 3 paramètres.



$$\forall i,\ a_i + b_i + d_i = l_{\text{out},i}$$

$$a_2 + b_1 + c0 = l_{\text{in},0}$$

$$a_0 + b_2 + c_1 = l_{\text{in},1}$$

$$a_1 + b_0 + c_2 = l_{\text{in},2}$$

$$\forall i,\ l_{\text{in},i} \leq l,\ l_{\text{out},i} \leq l$$

FIG. 4.3: *3-étoile générique et ses paramètres.*

Cette réduction est opérée par l'algorithme 4. La méthode employée transforme une instance générique en une instance réduite tout en ne faisant qu'augmenter le coût de la coloration fractionnaire. Cette propriété nous autorise à n'étudier que les étoiles réduites pour obtenir une borne supérieure sur le coût de la coloration d'une instance quelconque.

L'algorithme 4 uniformise la charge sur l'étoile puis concatène des paires de chemins de longueur 1 en chemins de longueur 2. Une étoile réduite typique est présentée en figure 4.4. L'intérêt principal de cette méthode est donné par le lemme 22 et tient en deux points. D'une part le nombre de paramètres nécessaires pour décrire une instance

Algorithme 4 Réduction d'un ensemble de chemins $\mathcal{P}$ en une 3−étoile réduite

Entrée: sommet central $= v_c$, feuilles $= v_i,\ i = 0, 1, 2$

Entrée: l charge de $\mathcal{P}$

1: **pour tout** arc e sous-chargé **faire**

2: ajouter des chemins de longueur 1 sur e pour atteindre l

3: **répéter**

4: choisir p, chemin de v_i à v_c

5: choisir q, chemin de v_c à $v_j,\ j \neq i$

6: concaténer p et q en un chemin $(v_i \rightarrow v_c \rightarrow v_j)$

7: **jusqu'à** ce qu'il n'y ait plus de chemins à concaténer

réduite n'est plus que de 3, notamment parce que l'on montre que les chemins de longueur 1 restant sont concentrés sur un seul arc. D'autre part cette réduction ne fait qu'ajouter des chemins et sur-contraindre la coloration en concaténant des chemins entre eux, ce qui ne peut qu'augmenter le coût d'une coloration fractionnaire.

Lemme 22 *Une 3−étoile réduite peut être décrite avec 3 paramètres α, β et γ, $\alpha+\beta+\gamma = l$, la charge de l'ensemble de chemins. De plus, la réduction opérée par l'algorithme 4 ne peut qu'augmenter le coût d'une coloration fractionnaire.*

Preuve : L'algorithme 4 commence par ajouter des chemins de longueur 1 sur les arcs de l'étoile de façon à égaliser la charge de tous les arcs (lignes 1 et 2). Cette étape ne fait qu'augmenter le coût d'une coloration fractionnaire. Lorsque la charge est uniforme, il y a une loi de conservation de flot au centre de l'étoile puisque chaque arc porte le même nombre de chemins dans chaque direction.

La boucle `répéter-jusqu'à` des lignes 3 à 7 termine en $3l$ étapes puisqu'il y a au plus $3l$ chemins arrivant au centre de l'étoile et qu'à chaque étape un tel chemin est traité. La boucle termine lorsqu'il n'y a plus de paire de chemins candidate à la concaténation, c'est à dire lorsque les chemins de longueur 1 sont concentrés sur un même arc. Supposons, sans perte de généralité, qu'il s'agit de l'arc entre le centre de l'étoile v_c et v_0. L'algorithme maintient la loi de conservation de flot au centre puisqu'à chaque étape un chemin arrivant en v_c est concaténé à un chemin en partant. En conséquence, à la fin de l'algorithme le

nombre de chemins de longueur 1 partant de v_c est égal au nombre de chemins y arrivant et ils sont tous sur le même arc. On note γ cette quantité.

L'ensemble de chemins résultant est alors une combinaison convexe de 3 sortes de stables canoniques : il y a α *cycles positifs* ($v_0 \to v_1 \to v_2$), β *cycles négatifs* ($v_0 \to v_2 \to v_1$), et γ ensembles de 2 chemins de longueur 1 entre v_0 et v_c et 2 chemins entre v_1 et v_2. Ce paramétrage est décrit en figure 4.4. Ainsi, $\alpha + \beta + \gamma = l$ et ces 3 paramètres suffisent à décrire un tel ensemble de chemins.

Enfin, la concaténation de chemins ne peut qu'augmenter le coût d'une coloration fractionnaire puisqu'une coloration fractionnaire valide pour l'ensemble réduit est valide pour l'ensemble initial. □

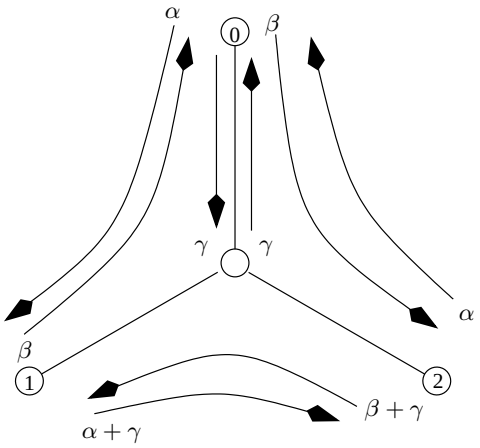


FIG. 4.4: *Une 3-étoile réduite et ses paramètres*

Les paramètres α, β et γ d'une étoile réduite peuvent s'exprimer en fonction du paramétrage d'une étoile générique comme suit.

$$a_0 = a_2 = \alpha, b_0 = b_1 = \beta, d_0 = c_0 = \gamma, a_1 = \alpha + \gamma, b_1 = \beta + \gamma.$$

Dans la suite nous définissons la notion de *doublon* qui est la base de notre étude de cas. En effet l'équilibrage de la coloration se traduit par le fait que chaque doublon est couvert identiquement et notre analyse consiste à identifier et compter les doublons de chaque type pour estimer le coût de leur couverture.

Les doublons

Nous appelons *doublon* une paire de chemins passant par un arc en sens opposé. Nous rappelons que l'on dit qu'un chemin *touche* un sommet v s'il en part, y arrive, ou passe par lui.

Définition 9 *À un sommet v_i d'une étoile, un* doublon *est un couple de chemins touchant v_i en sens opposé. Un* demi–doublon *est un chemin touchant v_i associé à aucun autre chemin, autrement dit, c'est un doublon où le second chemin est le chemin vide $p_\emptyset$.*

La définition d'une coloration fractionnaire $w-$équilibrée requiert que sur chaque arc la variable de trace de toute paire de chemins passant par cet arc en sens opposé vaille $\frac{w}{l}$. Les contraintes de couverture de chaque sommet impose qu'un demi–doublon soit couvert avec un poids $1-w$. Nous commençons donc par compter tous les doublons et demi–doublons qui doivent être couverts par notre coloration. Nous en déduisons ensuite la nature et le nombre de stables qui doivent être utilisés.

Définition 10 *Le* type d'un (demi–)doublon *dépend des chemins qui le composent. Nous nommons ces types en fonction des stables canoniques cités dans la preuve du lemme 22 et donnons le nombre de ces doublons présents à chaque sommet.*

- *Un doublon de type :*

  1. *A est composé de 2 chemins appartenant à un cycle positif ;*

     $v_0 : \alpha^2,\ v_{1,2} : \alpha(\alpha+\gamma)$

  2. *B est composé de 2 chemins appartenant à un cycle négatif ;*

     $v_0 : \beta^2,\ v_{1,2} : \beta(\beta+\gamma)$

  3. *C est composé de 2 chemins de longueur* 1 *;*

     $v_0 : \gamma^2$

  4. *D_i est composé de 2 chemins parallèles de longueur* 2 *ne touchant pas v_i ;*

D_0	$v_{1,2} : (\alpha+\gamma)(\beta+\gamma)$
D_1	$v_{0,2} : \alpha\beta$
D_2	$v_{0,1} : \alpha\beta$

5. $E_i,\ i = 1, 2$, *est composé d'un chemin de longueur* 1 *et d'un chemin touchant le sommet* v_i *dans un cycle positif ;*

   $v_0 : \alpha\gamma$ *chacun*

6. $F_i,\ i = 1, 2$, *est composé d'un chemin de longueur* 1 *et d'un chemin touchant le sommet* v_i *dans un circuit négatif ;*

   $v_0 : \beta\gamma$ *chacun.*

– *Un demi–doublon de type :*

1. $A_i^{\emptyset}$ *est un chemin ne touchant pas le sommet* v_i *dans un circuit positif ;*

$A_0^{\emptyset}$	$v_{1,2} : \alpha + \gamma$
$A_1^{\emptyset}$	$v_{0,2} : \alpha$
$A_2^{\emptyset}$	$v_{0,1} : \alpha$

2. $B_i^{\emptyset}$ *est un chemin ne touchant pas le sommet* v_i *dans un circuit négatif ;*

$B_0^{\emptyset}$	$v_{1,2} : \beta + \gamma$
$B_1^{\emptyset}$	$v_{0,2} : \beta$
$B_2^{\emptyset}$	$v_{0,1} : \beta$

3. $C_+^{\emptyset}$ *est un chemin de longueur* 1 *de* v_0 *à* v_c *;*

   $v_0 : \gamma$

4. $C_-^{\emptyset}$ *est un chemin de longueur* 1 *de* v_c *à* v_0 *;*

   $v_0 : \gamma$.

La coloration

La coloration équilibrée que nous construisons utilise des stables fortement reliés à la caractérisation des doublons qu'ils doivent couvrir. D'autre part, si notre but était de construire une coloration sans contrainte, il aurait été suffisant d'utiliser des stables maximaux pour l'inclusion. Cependant dans notre cas la contrainte de $w-$équilibrage nous amène à utiliser des stables non maximaux. Le tableau 4.5 récapitule les types de stables qui sont utilisés par notre coloration. La table 4.6 recense les (demi–)doublons couverts par chaque type de stables donné dans la table 4.5. Le nombre de stables de chaque type devant être utilisés est déduit du décompte des doublons à couvrir détaillé ci-dessous.

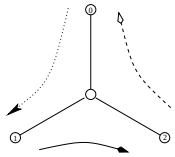
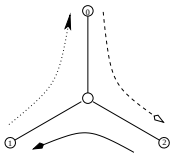
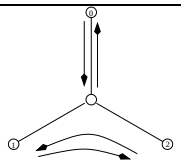
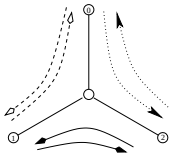
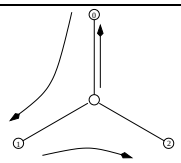
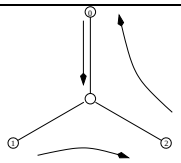
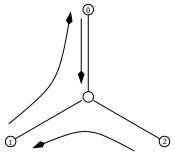
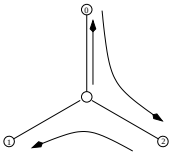
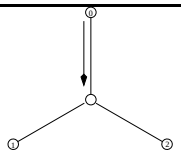
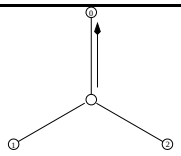
			
$A,\ A_i^{\emptyset}, i=0,1,2$	$B,\ B_i^{\emptyset}, i=0,1,2$	C	$D_i, i=0,1,2$
			
E_1	E_2	F_1	F_2
			
	$C_+^{\emptyset}$	$C_-^{\emptyset}$	

FIG. 4.5: *Les stables utilisés.*

Les stables de type A (respectivement B) couvrent un doublon de type A (B) à chaque sommet. Il y a α^2 (β^2) doublons de ce type en v_0 et $\alpha(\alpha+\gamma)$ en $v_{1,2}$. Il faut donc utiliser α^2 (β^2) ensembles de type A (B) et il reste $\alpha\gamma$ ($\beta\gamma$) doublons à couvrir en v_1 et v_2.

Les ensembles de type C couvrent un doublon de type C en v_0 et un de type D_0 en v_1 et v_2. Ceux de type D_0 couvrent un doublon de type D_0 en v_1 et v_2. Il y a γ^2 doublons de type C et $(\alpha+\gamma)(\beta+\gamma)$ doublons de type D_0. Il faut donc utiliser γ^2 ensembles de type C et $(\alpha+\gamma)(\beta+\gamma)-\gamma^2$ ensembles de type D_0 pour tout couvrir.

Les ensembles de type D_i, $i=1,2$ couvrent les doublons du même type, il en faut $\alpha\beta$.

Les ensembles de type E_i (respectivement, F_i) couvrent un doublon du type correspondant en v_0. Il y en a $\alpha\gamma$ ($\beta\gamma$), ce qui donne le nombre d'ensembles à utiliser. Ces ensembles couvrent aussi les doublons de type A (B) restant et des demi–doublons de type $A_0^{\emptyset}$ ($B_0^{\emptyset}$) en v_1 et v_2.

De par la définition d'une coloration fractionnaire $w-$équilibrée, tous ces stables sont pris avec poids $\frac{w}{l}$ et couvrent tous les doublons de l'étoile et une partie des demi–doublons. Pour terminer la couverture, on prend α ensembles de type $A_{1,2}^{\emptyset}$ et β de type $B_{1,2}^{\emptyset}$, chacun avec poids $1-w$. Ils couvrent les demi–doublons correspondants. Il reste enfin à couvrir 2α demi–doublons de type $C_{+/-}^{\emptyset}$ avec les stables du même type. Ces ensembles ont la

Type d'EI	Type de doublon			Nombre	Type d'EI	Type de doublon			Nombre
	v_0	v_1	v_2			v_0	v_1	v_2	
A	A			α^2	B	B			β^2
C	C	D_0		γ^2	D_0	D_0			$\alpha\gamma + \alpha\beta + \beta\gamma$
D_1	D_1		D_1	$\alpha\beta$	D_2	D_2			$\alpha\beta$
E_1	E_1	A	$A_0^\emptyset$	$\alpha\gamma$	E_2	E_2	$A_0^\emptyset$	A	$\alpha\gamma$
F_1	F_1	B	$B_0^\emptyset$	$\beta\gamma$	F_2	F_2	$B_0^\emptyset$	B	$\beta\gamma$

FIG. 4.6: *Couverture des doublons.*

propriété de pouvoir être combinés avec les ensembles de type D_0 pris précédemment, au moins pour une part d'entre eux. Ainsi, le coût total des ensembles de type $C^\emptyset_{+/-}$ et D_0 est coût $\left(C^\emptyset_{+/-}, D_0\right) = \max\left\{\frac{w}{l}(\alpha\gamma + \alpha\beta + \beta\gamma),\ 2\gamma(1-w)\right\}$.

Le coût total Γ de la coloration obtenue est la somme des poids de tous les ensembles choisis : $\Gamma = \frac{w}{l}(\alpha^2 + \beta^2 + \gamma^2 + 2\alpha\beta + 2\alpha\gamma + 2\beta\gamma) + \text{coût}\left(C^\emptyset_{+/-}, D_0\right) + 2(1-w)(\alpha+\beta)$.

En utilisant la réduction classique, $(\alpha^2+\beta^2+\gamma^2+2\alpha\beta+2\alpha\gamma+2\beta\gamma) = (\alpha+\beta+\gamma)^2 = l^2$, on obtient :

$$\Gamma = wl + \text{coût}\left(C^\emptyset_{+/-}, D_0\right) + 2(1-w)(\alpha+\beta).$$

Analyse numérique

L'équilibrage de la coloration et la couverture des demi–doublons $A_0^\emptyset$ et $B_0^\emptyset$ imposent les contraintes suivantes à w.

$$\alpha\gamma\frac{w}{l} \leq (\alpha+\gamma)(1-w) \text{ et } \beta\gamma\frac{w}{l} \leq (\beta+\gamma)(1-w).$$

Ces dernières combinées amènent à

$$1-w \geq \frac{\alpha\gamma}{\alpha+\gamma}\frac{w}{l} \text{ et } 1-w \geq \frac{\beta\gamma}{\beta+\gamma}\frac{w}{l}.$$

Il est facile de voir que $w = \frac{4}{5}$ est toujours un choix valide. Cette valeur de w donne une borne supérieure pour Γ. De plus, $\frac{4}{5}$ est la meilleure solution pour w quand $\alpha = \gamma = \frac{l}{2}$. La borne suivante est donc un maximum.

$$\Gamma \leq \frac{4l}{5} + \frac{2}{5}\max\left\{\frac{2(\alpha\beta+\alpha\gamma+\beta\gamma)}{l}, \gamma\right\} + \frac{2(\alpha+\beta)}{5}.$$

Cette valeur est maximale et vaut $\frac{7l}{5}$ lorsque $\alpha = \beta = \frac{l}{2}, \gamma = 0$, comme illustré par la figure 4.7. La proposition 23 conclut l'analyse sur ce résultat.

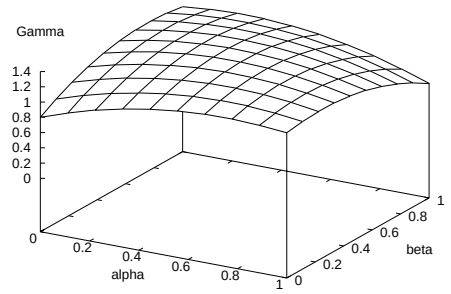


FIG. 4.7: $\Gamma(\alpha, \beta)$ *avec* $\gamma \leq \frac{2}{l}(\alpha\beta + \alpha\gamma + \beta\gamma)$.

Proposition 23 *Tout ensemble de chemins de charge l sur un arbre binaire admet une coloration fractionnaire $\frac{4}{5}-$équilibrée de coût au plus $\frac{7l}{5}$.*

Preuve : Nous prouvons la proposition pour les 3−étoiles réduites, le corollaire 20 et le lemme 22 donnent le résultat général.

- Si $\gamma \leq \frac{2}{l}(\alpha\beta + \alpha\gamma + \beta\gamma)$, alors $\Gamma = \frac{4}{5}(l + \frac{\alpha\beta+\gamma(l-\gamma)}{l}) + \frac{2}{5}(l-\gamma)$, qui est maximum pour $\alpha = \beta = \frac{l}{2}, \gamma = 0$. $\Gamma \leq \frac{7l}{5}$ (cf figure 4.7).
- Sinon $\Gamma = \frac{4l}{5} + \frac{2\gamma}{5} + \frac{2}{5}(l-\gamma) = \frac{6l}{5}$.

En conclusion, $\forall\ \alpha+\beta+\gamma = l,\ \Gamma \leq \frac{7l}{5}$. □

Du fractionnaire à l'aléatoire

Ce dernier résultat attire l'attention sur la proximité entre notre travail et l'algorithme aléatoire de coloration de chemins dans un arbre binaire, proposé par Auletta et coll. (2000). En effet, ces deux approches, *a priori* très différentes, se rapprochent sur plusieurs points, dont le résultat puisque l'algorithme de Auletta et coll. (2000) calcule une coloration des chemins de coût $\frac{7l}{5} + o(l)$.

Cet algorithme suit un parcours en largeur d'abord des sommets de l'arbre binaire, et colore les chemins selon un processus glouton. Quand un sommet v de l'arbre est traité, l'hypothèse est faite que les $2l$ chemins qui touchent v et des sommets précédemment traités (l dans chaque direction) sont coloriés avec ρl couleurs. La proportion $(\rho - 1)$ de couleurs "en trop" donne la marge de manoeuvre qui permet de colorier, sans trop rajouter de couleurs, les chemins qui touchent v et qui ne le sont pas encore, c'est à dire ceux qui restent dans le sous-arbre enraciné en v. Afin d'étendre la coloration partielle à ces chemins, l'algorithme se base sur une distribution dite *fortement uniforme* des chemins qui se définit de la même manière qu'une trace équilibrée et traite les chemins selon une classification de leur trajet similaire à la taxinomie des doublons décrite dans la définition 9 et la table 4.6.

En ce sens, la distribution fortement uniforme des chemins peut s'interpréter comme le résultat de la coloration fractionnaire $\rho-$équilibrée d'une étoile. De manière équivalente, la coloration fractionnaire $\rho-$équilibrée d'une étoile peut être vue comme la moyenne de cette distribution. Un autre argument vient consolider cette intuition : Auletta et coll. (2000) prouvent que la valeur du paramètre ρ permettant à leur algorithme de garantir, avec grande probabilité, une coloration valide la moins chère possible est $\frac{4}{5}$. Leur raisonnement est plus complexe que celui qui nous amène à choisir cette même valeur pour la coloration fractionnaire équilibrée, principalement parce que nous cherchons un paramètre qui assure l'existence d'une coloration fractionnaire pour une étoile alors qu'ils cherchent à minimiser le coût global d'une coloration entière qu'ils sont sûrs d'obtenir, mais certains arguments sont similaires. De plus, le cas qui pose le plus de problèmes à leur algorithme et les forcent à choisir $\rho = \frac{4}{5}$ est le cas où les chemins arrivant en v se séparent équitablement entre le fils droit et le fils gauche. Ce cas correspond dans notre étude à $\alpha = \beta = \frac{l}{2}$, c'est à dire la situation où $\frac{4}{5}$ est le meilleur choix pour notre paramètre et où la coloration fractionnaire $\frac{4}{5}-$équilibrée est la plus coûteuse.

Ces similitudes donnent l'intuition plus générale que la coloration fractionnaire peut être un bon guide pour une coloration aléatoire des chemins. Cette intuition est exploitée dans la section suivante.

4.4 Du fractionnaire optimal à l'entier approché

La coloration fractionnaire n'apporte, en soi, qu'une borne inférieure sur le coût d'une coloration entière plus serrée que la charge de l'ensemble des chemins. À l'aune des remarques de la section précédente, il semble toutefois raisonnable de penser exploiter une telle coloration fractionnaire pour construire une bonne coloration entière, au moins dans les arbres.

Le meilleur algorithme polynomial de coloration de chemins dans les arbres généraux a été proposé par Erlebach, Jansen, Kaklamanis, Mihail, et Persiano (1999). Cet algorithme déterministe colore un ensemble de chemins de charge l dans un arbre quelconque avec au plus $\frac{5l}{3}$ couleurs, ce qui donne une $\frac{5}{3}-$approximation du nombre chromatique. Les auteurs ont aussi prouvé que c'était le meilleur algorithme polynomial, déterministe et glouton. Dans la suite nous améliorons ce résultat dans les arbres de degré borné en exploitant une coloration fractionnaire optimale dans un algorithme aléatoire.

Étant donné une coloration fractionnaire optimale $\bar{x}$ d'un ensemble $\mathcal{P}$ de chemins dans un arbre de degré borné $\mathcal{T}$, l'idée consiste à opérer un arrondi aléatoire de $\bar{x}$, produisant ainsi une coloration entière partielle de $\mathcal{P}$, que nous notons x. La coloration est alors complétée en utilisant l'algorithme de Erlebach, Jansen, Kaklamanis, Mihail, et Persiano (1999).

La phase d'arrondi aléatoire consiste à prendre indépendamment chaque variable $x(I)$ pour chaque stable $I \in \mathcal{I}$ et à l'arrondir aléatoirement à $\bar{x}(I) = 1$ avec probabilité $x(I)$, $\bar{x}(I) = 0$ avec probabilité $1 - x(I)$. Ainsi, la valeur moyenne de $\bar{x}(I)$ vaut exactement $x(I)$. Après cet arrondi aléatoire il reste un ensemble $\mathcal{P}'$ de chemins non coloriés, c'est à dire que pour chacun des chemins de $\mathcal{P}'$, aucun stable le contenant n'a vu son poids arrondi à 1. Le coût de la coloration partielle est la somme des poids arrondis des stables. C'est donc une somme de Bernouilli de valeur moyenne la somme des poids fractionnaire χ_f. D'après les bornes de Chernoff, le coût de la coloration partielle est donc $\chi_f + o(\chi_f)$ avec grande probabilité. Enfin, l'analyse menée par Kumar (1998) montre qu'avec grande probabilité la charge de $\mathcal{P}'$ est $l(\mathcal{P}') \leq \frac{l}{e} + o(l)$. La coloration de $\mathcal{P}'$ demande donc au plus $\frac{5l}{3e} + o(l)$ couleurs supplémentaires pour terminer la coloration. Le coût total de cette

coloration aléatoire est donc, avec grande probabilité,

$$\chi_f(\mathcal{T}, \mathcal{P}) + \frac{5l}{3e} + o(l).$$

De plus, l'encadrement trivial $l \leq \chi_f \leq \chi$ nous donne les résultats d'approximation suivant.

Proposition 24 *Il existe un algorithme aléatoire polynomial d'approximation de facteur* $1.61 + o(1)$ *pour le problème de coloration de chemins dans un arbre de degré borné.*

Corollaire 25 *Pour tout ensemble de chemin* $\mathcal{P}$ *dans un arbre de degré borné* $\mathcal{T}$*, on peut borner le nombre chromatique des chemins ainsi :*

$$\chi(T, \mathcal{P}) \leq \chi_f(T, \mathcal{P}) + \frac{5l(T, \mathcal{P})}{3e} + o(l(T, \mathcal{P})).$$

Il est possible d'utiliser des arguments similaires aux arbres d'anneaux de degré borné et obtenir une approximation de facteur 2.22. Nous conjecturons aussi qu'il est possible d'écrire une approximation aléatoire de facteur $1 + o(1)$ se basant sur une coloration fractionnaire optimale qui se comporte comme l'algorithme de Auletta et coll. (2000) en considérant que se dernier se base sur une coloration $\frac{4}{5}$−équilibrée. Ce travail n'a toutefois pas encore abouti, notamment parce qu'il est difficile d'exprimer les propriétés de la distribution des traces obtenues par l'algorithme présenté en section 4.2.

Chapitre 5

Approximation aléatoire du multiflot entier

Nous avons vu au chapitre 3 que le routage optique dans les réseaux multifibres s'écrit comme un multiflot entier dans un graphe auxiliaire. Cette modélisation nous amène à chercher des algorithmes d'approximation du multiflot performants. Nous proposons donc en section 5.1 une amélioration de l'algorithme d'arrondi aléatoire de Raghavan (1994), introduit en section 1.5.3. Nous cherchons ensuite à produire ces solutions approchées le plus rapidement possible et pour cela étudions en section 5.2 les algorithmes combinatoires d'approximation du multiflot fractionnaire qui ont vocation à être intégrés au cœur des processus d'arrondi aléatoire. Nous améliorons un des algorithmes les plus efficaces de la littérature en utilisant des techniques de mise à jour dynamique des arborescences de plus courts chemins que nous spécialisons ensuite à notre modèle de routage optique.

5.1 Approximation du multiflot entier par arrondi aléatoire

Les problèmes de multiflots s'appliquent bien sûr à l'optimisation des réseaux de télécommunication mais aussi à bien d'autres situations telles que l'optimisation des réseaux de transport ou encore la conception de circuits VLSI. Ces nombreuses applications ont motivé et motivent encore un grand nombre d'études de recherche opérationnelle ou algo-

rithmiques. En effet, ces problèmes font partie des problèmes $\mathcal{NP}$-durs les plus difficiles à résoudre. Raghavan (1994) a proposé un algorithme d'arrondi aléatoire pour approximer le multiflot que nous présentons ici. Après avoir constaté que cet algorithme peut fournir des solutions qui violent les contraintes de capacité du réseau, nous en proposons une amélioration qui assure le respect de toutes les contraintes. L'intuition et la pratique montrent que notre algorithme se comporte mieux que celui de Raghavan (1994) bien que l'analyse théorique ne permette que de conclure à un comportement au moins aussi bon[1].

5.1.1 Arrondi aléatoire par chemins

L'algorithme 5, proposé par Raghavan (1994), est un arrondi aléatoire *par chemin*. Le problème général des techniques d'arrondi aléatoire est qu'elles ne permettent pas d'assurer que les contraintes sont toujours respectées, tout particulièrement quand il s'agit de contraintes d'égalité comme les lois de Kirchoff de conservation de flot. Pour contourner cette difficulté, l'algorithme de Raghavan (1994) ne procède pas à un véritable arrondi aléatoire des variables du multiflot, mais considère des chemins. Il s'appuie pour cela sur une version fractionnaire des idées de la section 3.3.1 qui assure qu'un flot fractionnaire f de valeur $|f|$ d'une source s à un puits t se décompose de manière *unique* en un ensemble de chemins de s à t pondérés tel que

- la somme des poids des chemins vaut $|f|$,
- pour tout arc e, la somme des poids des chemins passant par e vaut $f(e)$.

L'arrondi aléatoire par chemin consiste alors à choisir aléatoirement un des chemins de la décomposition avec probabilité le poids du chemin divisé par $|f|$: $Pr[p \text{ est choisi}] = \frac{f(p)}{|f|}$. Le fait d'arrondir le flot par chemin permet d'assurer le respect des contraintes de conservation de flot puisqu'elles sont intrinsèques aux chemins.

Le problème majeur de cette technique est que la décomposition d'un flot fractionnaire peut donner un ensemble de chemins de taille exponentielle. Il est donc impossible d'énumérer tous les chemins avant d'en choisir un. L'idée de Raghavan (1994) est de construire un des chemins de la décomposition aléatoirement à partir de la fonction de

[1] Ce travail a fait l'objet de la publication (Coudert et Rivano 2002)

Algorithme 5 Arrondi aléatoire du multiflot

Entrée: $G = (V, E),\ c : E \to \mathbb{N},\ C = \{(s_i, t_i),\ i = 1 \dots k\} \subseteq V^2,\ \forall i = 1 \dots k,\ d_i \in \mathbb{N}$

Entrée: `marche_aléatoire` : fonction de marche aléatoire sur un flot renvoyant un chemin

Entrée: `multiflot_fractionnaire` : calcul d'un multiflot fractionnaire

Sortie: $\forall i\ \mathcal{P}_i$ ensemble de d_i chemins de s_i à t_i.

Sortie: $\cup_i \mathcal{P}_i$ réalise le multiflot C, d sur G en respectant les capacités $c + o(c)$.

1: $f = \{f_i,\ i = 1 \dots k\}$ $\leftarrow$ `multiflot_fractionnaire`(G, c, C, d).
2: **pour tout** $i = 1 \dots k$ **faire**
3: **pour** d_i fois **faire**
4: $p \leftarrow$ `marche_aléatoire`(f_i)
5: $\mathcal{P}_i \leftarrow \mathcal{P}_i \cup \{p\}$

flot sur les arcs de sorte que la probabilité de construire un chemin soit exactement son poids dans la décomposition. Pour cela, il suffit de remarquer que les arcs sur lesquels la fonction de flot est non nulle forment un *graphe orienté acyclique* (DAG pour *Directed Acyclic Graph*). Dans ce DAG, la décomposition du flot en chemins consiste à prendre un chemin pour chacun des trajets possibles de la source au puits et de le pondérer par la proportion de flot qui emprunte ce trajet en supposant qu'à chaque bifurcation le flot se répartit uniformément entre tous les trajets possibles. Pour construire un tel chemin avec probabilité son poids, il suffit de faire une marche aléatoire dans le DAG guidée par le flot. C'est-à-dire qu'on part de la source et, à chaque bifurcation, on choisit une direction aléatoirement avec probabilité la proportion de flot qui va dans cette direction.

L'algorithme de Raghavan (1994) consiste finalement à faire une marche aléatoire pour chaque unité de flot devant être routé d'une source à une destination, chaque marche aléatoire étant indépendante des autres au sens des probabilités.

L'indépendance des marches aléatoires entre elles permet de faire une analyse assez simple des performances de l'algorithme 5.

Espérance de l'occupation d'un arc par une unité de flot

Il est difficile d'analyser l'algorithme en considérant les chemins eux-mêmes. En effet ceux-ci amènent des corrélations entre l'occupation des arcs puisqu'un chemin passe par plusieurs d'entre eux. Il faut donc arriver à ne considérer qu'un arc en particulier. Soit e

un tel arc. Le résultat de l'arrondi d'une unité du flot f, allant d'une source s à un puits t, utilise, sur cet arc, une unité de flot si un chemin p utilisant e est choisi, 0 sinon. Or, un chemin p est choisi avec probabilité son poids. De plus, on ne choisit qu'un seul chemin à la fois, les choix de deux chemins sont donc exclusifs, et la probabilité de la somme de ces événements vaut donc la somme des probabilités $\sum_{p \ni e} Pr[p \text{ est choisi}] = \sum_{p \ni e} \frac{f(p)}{|f|} = \frac{f(e)}{|f|}$. La quantité de flot arrondi sur l'arc e, pour une unité flot allant de s à t, vaut donc 1 avec probabilité $\frac{f(e)}{|f|}$.

Écart entre les capacités fractionnaires et arrondies

Comme chaque unité de flot est arrondie indépendamment des autres, la quantité de flot totale passant par e est la somme de variables mutuellement indépendantes qui valent 0 ou 1 avec la probabilité donnée ci-dessus. Il s'agit donc d'une *somme de Bernouilli*. L'espérance de la capacité arrondie $\hat{c}$ utilisée est donc la somme des espérances des variables de flot arrondies :

$$[\hat{c}(e)] = \sum_{f} \sum_{1}^{|f|} \frac{f(e)}{|f|} = \sum_{f} f(e) = c(e)$$

De plus, le théorème de Chernoff pour les sommes de Bernouilli donne

$$\forall\ e,\ \forall\ \delta,\ Pr[\hat{c}(e) - c(e) \geq \delta] \ \leq\ 2e^{0.38\delta^2 c(e)}.$$

Le résultat qui nous intéresse est que, pour tout arc, l'écart entre la capacité arrondie et la capacité fractionnaire ne soit pas trop grand. Si l'on fait abstraction des dépendances entre les variables aléatoires de deux arcs, et que l'on choisit pour chaque arc $\delta = \sqrt{c(e) \log |E|}$, on obtient la majoration suivante

$$Pr[\forall\ e,\ \hat{c}(e) - c(e) \geq \sqrt{c(e) \log |E|}] \ \leq\ O\left(\frac{1}{m}\right).$$

Autrement dit, avec haute probabilité l'algorithme 5 donne une $1+o(1)$-approximation des capacités. Deux remarques s'imposent directement. D'une part, avoir négligé les dépendances entre les sommes de Bernouilli sur chaque arc amène à une très forte majoration de la probabilité d'échec de l'algorithme, et par conséquent la borne $\sqrt{c(e) \log |E|}$ est lâche. D'autre part, il est évident que l'algorithme d'arrondi aléatoire est d'autant

plus performant que les charges sont importantes, puisque la déviation est un écart additif. La borne $\sqrt{c(e)\log|E|}$ laisse alors à penser que des charges supérieures à $\sqrt{\log|E|}$ amèneront de très bon résultats d'approximation.

En tout état de cause, l'indépendance des choix des chemins, qui permet une analyse théorique simple, est un facteur de sous-optimalité en pratique. En effet, il arrive que l'on choisisse un chemin qui viole les capacités sur un lien, alors qu'un autre choix était possible. Pour améliorer les résultats, nous proposons d'introduire des dépendances entre les choix, en recalculant des multiflots fractionnaires qui s'adapteront aux choix faits précédemment.

5.1.2 Introduction de dépendances par arrondis itératifs

L'argument principal de l'algorithme 5 est que la relaxation linéaire est un bon guide pour l'arrondi aléatoire, et que l'arrondi par chemin permet d'assurer que la solution construite est un flot. Cependant à chaque arrondi d'un chemin, la qualité de guide du flot fractionnaire initial se dégrade. Notre stratégie consiste à profiter du fait que chaque arrondi de chemin construit une portion de flot pour l'intégrer à la relaxation linéaire, en modifiant les capacités du réseau de flot. Ainsi en recalculant un flot fractionnaire, nous retrouvons un bon guide qui, de plus, tient compte des choix faits en amont. L'algorithme 6 fonctionne sur ce principe et, à chaque étape, arrondi un chemin, met à jour les capacités le long de ce chemin, et résout le nouveau multiflot fractionnaire.

L'intuition dicte que l'algorithme 6 va calculer des solutions plus proche de l'optimal que l'algorithme 5. Nous verrons dans la section suivante que cette intuition est validée par la pratique, mais l'analyse théorique ne donne pas de conclusion aussi tranchée.

Conjecture sur l'analyse de l'algorithme 6

Le processus d'arrondi de chaque unité de flot ne change pas par rapport à l'algorithme 5, l'analyse de Raghavan (1994) est donc toujours valide, et une unité de flot de f arrondie utilise un arc e avec probabilité $\frac{f(e)}{|f|}$. Le changement est qu'à chaque étape un nouveau flot est calculé, et la quantité de flot fractionnaire passant sur e peut être modifiée. Cet algorithme est donc plus général que l'algorithme de Raghavan (1994). En particulier

Algorithme 6 Calcul approché de multiflot entier

Entrée: $G = (V, E),\ c : E \to \mathbb{N},\ C = \{(s_i, t_i),\ i = 1 \dots k\} \subseteq V^2,\ \forall i = 1 \dots k,\ d_i \in \mathbb{N}$

Entrée: `marche_aléatoire` : fonction de marche aléatoire sur un flot renvoyant un chemin

Entrée: `multiflot_fractionnaire` : calcul d'un multiflot fractionnaire

Sortie: $\forall i\ \mathcal{P}_i$ ensemble de d_i chemins de s_i à t_i.

Sortie: $\cup_i \mathcal{P}_i$ réalise le multiflot C, d sur G en respectant les capacités $c + o(c)$.

1: **répéter**
2: $f = \{f_i,\ i = 1 \dots k\}\ \leftarrow$ `multiflot_fractionnaire`(G, c, C, d).
3: **pour** i tel que $d_i > 0$, choisi aléatoirement, **faire**
4: $p \leftarrow$ `marche_aléatoire`(f_i)
5: $\mathcal{P}_i \leftarrow \mathcal{P}_i \cup \{p\}$
6: $d_i \leftarrow d_i - 1$
7: **pour tout** $e \in\ p$ **faire**
8: $c(e) \leftarrow c(e) - 1$
9: **jusqu'à** $\sum_i d_i = 0$

ce dernier peut être "émulé" par l'algorithme 6 en utilisant la procédure de recalcul du multiflot fractionnaire suivante :

> Après avoir arrondi une unité de flot pour la commodité i, le nouveau multiflot est égal au précédent pour toutes les autres commodité tandis que le flot de la commodité i est simplement multipliée par $\frac{d_i - 1}{d_i}$ puis d_i est décrémenté. De la sorte le choix aléatoire d'un chemin se fait avec la même probabilité que lors de l'étape suivante, à moins que d_i ne vaille 1 auquel cas la commodité est éliminée.

Par ailleurs, cet algorithme de calcul de multiflot fractionnaire est clairement sous-optimal par rapport à la capacité utilisée sur les arcs. Il semble donc que l'utilisation d'un calcul de multiflot optimisant ce critère donnera des résultats au moins aussi bon.

Pour ces raisons, et parce que la pratique confirme cette intuition, nous conjecturons que l'algorithme 6 approxime les capacités d'un multiflot entier au moins aussi bien que l'algorithme de Raghavan (1994).

Nous avons également implanté une heuristique intermédiaire qui arrondit aléatoirement une unité de flot de chaque commodité avant de résoudre un nouveau programme linéaire. Cette heuristique est plus rapide que l'algorithme 6, mais sûrement d'une précision intermédiaire entre celles des précédents algorithmes, selon la même intuition que celle guidant l'algorithme précédent. Encore une fois, la pratique valide l'intuition.

Dans la suite, nous étudions le comportement de ces algorithmes sur deux exemples théoriques et un plus important, issus de l'optimisation du routage optique d'un réseau réel.

5.1.3 Résultats de simulations et discussions

Nos simulations ont été effectuées sur les réseaux de flot du routage optique (*cf* chapitre 3) de deux réseaux : un anneau à 10 sommets et un réseau pan-américain reliant 65 villes par 75 liens bidirectionnels. Nous étudions deux instances de communication aléatoires sur l'anneau. La première instance (I_1) représente 376 chemins optiques, la seconde (I_2) 316. L'instance de communication sur le réseau pan-américain est une instance *réaliste* de 1305 chemins optiques. Les calculs ont été effectués sur un PIII 933MHz avec 512Mo de RAM.

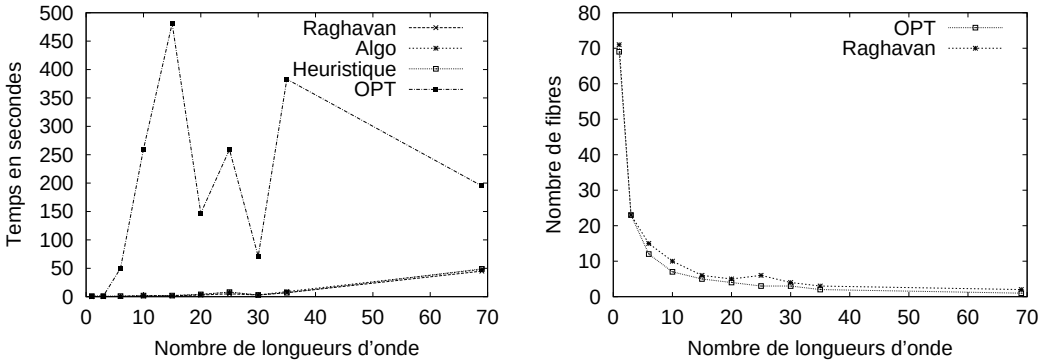


FIG. 5.1: *Anneau à 10 sommets avec* I_1.

Nous reportons sur la figure 5.1 les résultats de nos simulations avec I_1, sans conversion. La courbe de droite représente le nombre minimum de fibres nécessaires à la résolution du RWA en fonction du nombre de longueurs d'onde. Celle de gauche, les temps mis par CPLEX et nos heuristiques pour résoudre le RWA dans chacun de ces cas. Sans aucune

surprise, nos heuristiques ont été bien plus rapides que CPLEX.

Par contre, la dépendance quasi-chaotique du temps de résolution de l'ILP au nombre de longueurs d'onde que l'on peut l'observer dans la figure 5.1 était inattendue. La taille de l'ILP augmente linéairement avec le nombre de longueurs d'onde. Nous nous attendions donc à ce que le temps de résolution de l'ILP croisse également avec le nombre de longueurs d'onde, comme nous l'observons sur la figure 5.3 pour le réseau pan-américain. Nous observons également ce comportement curieux avec I_2. Le ratio entre les temps de résolution optimale par CPLEX et de calcul de l'algorithme 6 est reporté sur la figure 5.2. Le nombre de longueurs d'onde et le nombre de fibres varient dans $(4, 14)$.

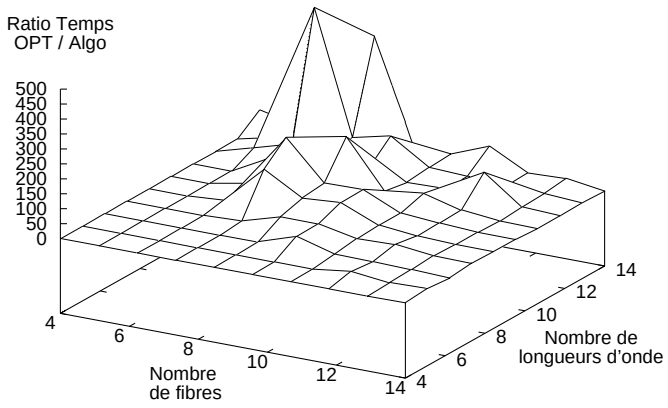


FIG. 5.2: *Anneau à 10 sommets avec* I_2

Nous pensons que ce phénomène est dû à la structure particulière de l'anneau, notamment sa symétrie. Notre formulation est aussi peut être trop générale pour ce type de réseau où il n'y a pas vraiment de routage mais un choix binaire du sens de rotation.

Néanmoins, nos algorithmes fournissent généralement une solution optimale et l'arrondi aléatoire admet un écart d'au plus 3 fibres par lien. Dans le cas de l'anneau monofibre sans conversion, nos résultats correspondent bien à l'analyse théorique du problème similaire de la *coloration des graphes arcs-circulaires* faite par Kumar (1998).

Les résultats obtenus sur le réseau pan-américain, reportés sur la figure 5.3, sont plus attendus. Nos algorithmes sont exponentiellement plus efficaces que CPLEX sur l'ILP et nous autorisent à manipuler des réseaux plus gros avec de plus grandes instances. Par

exemple, CPLEX n'a pas pu résoudre l'ILP avec plus de 22 longueurs d'onde, alors que nos heuristiques ont pu atteindre 66. Remarquons que ces limites sont autant dues à l'énorme temps de calcul qu'à de trop importants besoins en mémoire de CPLEX, que ce soit pour l'ILP ou pour les relaxations linéaires.

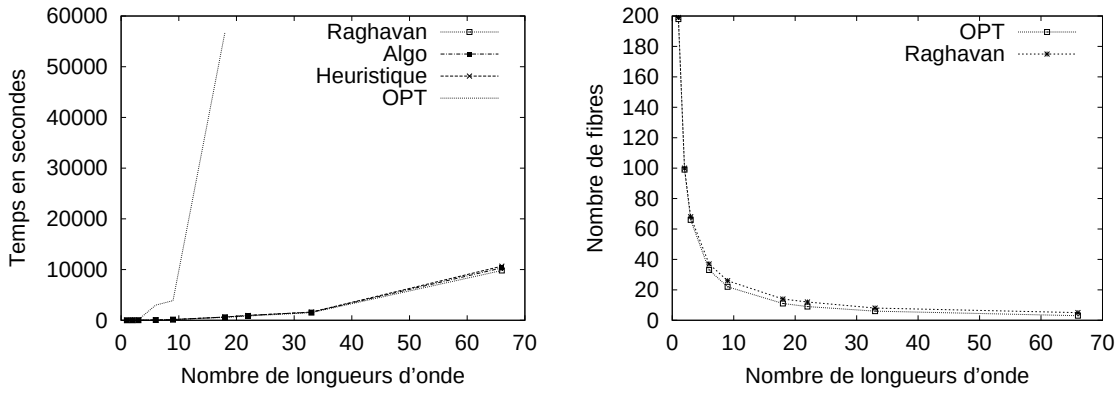


FIG. 5.3: *Résultats pour le réseau pan-américain*

Nous pensons pouvoir aller bien plus loin avec la formulation *arc-chemin* de la relaxation fractionnaire du multiflot, ainsi qu'avec des techniques efficaces de recherche opérationnelle telle que la génération de colonnes.

5.2 Approximations combinatoires du multiflot fractionnaire

Le travail qui suit s'attache à présenter plusieurs améliorations à l'algorithme combinatoire d'approximation du multiflot fractionnaire proposé par Fleischer (2000). Dans un contexte général ou bien spécifique au dimensionnement de réseaux développé au chapitre 3, il s'agit avant tout d'accélérer le calcul d'un multiflot fractionnaire qui a vocation à être utilisé comme la routine principale de l'algorithme 6[2].

Approximer le multiflot fractionnaire Les algorithmes 5 et 6 présentés en section 5.1 calculent un multiflot entier approché en utilisant un ou plusieurs calculs de multiflots

[2]Ce travail a fait l'objet de la publication (Bouklit, Coudert, Lalande, Paul, et Rivano 2003)

fractionnaires. De plus, l'algorithme 6 nécessite un calcul de multiflot fractionnaire pour chaque commodité dans un processus itératif. La question de la rapidité de résolution du multiflot fractionnaire n'en devient que plus cruciale.

Se pose alors la question de l'utilité de solutions fractionnaires optimales. En effet, ces algorithmes induisent des approximations qui sont additives sur les capacités. En fait, si l'on regarde plus précisément, cet écart est calculé par rapport aux capacités fractionnaires utilisées. Par conséquent, utiliser une $(1+\epsilon)$-approximation du multiflot fractionnaire amènerait à une solution dont les capacités sont, au pire, $(1+\epsilon)OPT + \sqrt{(1+\epsilon)OPT}$. L'ordre de grandeur de l'écart d'approximation ne changera donc pas de manière importante et il est naturel de chercher à exploiter le degré de liberté ainsi dégagé pour accélérer le calcul du multiflot entier.

Nous présentons ici en détail les algorithmes de Garg et Konemann (1998) et de Fleischer (2000) qui partagent une stratégie commune fondée sur les travaux de Young (1995). L'idée principale est de générer une solution du multiflot en se guidant sur la construction d'une solution au problème dual, qui a ici une interprétation combinatoire en termes de plus courts chemins, par des modifications de faible amplitude.

5.2.1 Pousser le long du chemin le moins chargé

L'algorithme de Garg et Konemann (1998) – algorithme 7 – se fonde sur le travail de Young (1995), et exploite l'interprétation combinatoire du dual du multiflot maximum exposée au chapitre 1. L'idée principale est de construire un multiflot en poussant de petites quantités de flot le long des chemins violant le plus les contraintes de longueur du dual, tout en maintenant une dépendance exponentielle entre la quantité de flot traversant un arc et sa longueur. Cela revient à choisir à chaque étape le chemin passant par les arcs les moins chargés possibles, ce qui va dans le sens de l'intuition. La forme de la dépendance, entre la longueur d'un arc et la quantité de flot qu'il supporte, conditionne les performances de l'algorithme. Dans la suite, nous notons SSSP (s,t) le plus court chemin de s à t.

La métrique Dans le cas où toutes les capacités valent c, la longueur d'un arc est de la forme $\delta(1+\epsilon)^i$ quand la quantité de flot traversant cet arc est $i\frac{c}{\alpha}$ où ϵ est le paramètre

d'approximation choisi et α et δ deux constantes fonctions de ϵ que nous déterminerons plus tard. La correspondance entre la longueur d'un arc et la quantité de flot est bien exponentielle. Quand les capacités varient, la correspondance n'est pas aussi directe et il n'y a pas de formule close pour la longueur d'un arc, mais l'idée est la même.

Algorithme 7 $(1+\epsilon)$-approximation du multiflot maximum

Entrée: $G=(V,E),\ |V|=n,\ |E|=m,\ C=\{(s_i,t_i),\ i=1\dots k\}\subseteq V^2, \epsilon>0$

Sortie: l métrique sur E t.q. tout chemin $s_i \to t_i$ soit de longueur ≥ 1.

Sortie: f $(1+\epsilon)$-approximation du multiflot maximum de C sur G

1: {Initialisation} $\forall p\in\mathcal{P},\ f(p)=0,\ \forall e\in E,\ l(e)=\delta$
2: **répéter**
3: $\quad\{l(\emptyset)\equiv\infty\}\ \min_p=\emptyset$
4: $\quad$**pour tout** $i=1\dots k$ **faire**
5: $\quad\quad p \leftarrow$SSSP (s_i,t_i)
6: $\quad\quad$**si** $l(p)<l(\min_p)$ **alors**
7: $\quad\quad\quad \min_p \leftarrow p$
8: $\quad c_m \leftarrow \min_{e\in\min_p} c(e)$
9: $\quad f(\min_p)\leftarrow f(\min_p)+\frac{c_m}{\alpha}$
10: $\quad\forall e\in\min_p,\ l(e)\leftarrow l(e)(1+\epsilon\frac{c_m}{c(e)})$
11: **jusqu'à** $l(\min_p)\geq 1$

L'algorithme A l'initialisation de l'algorithme, le flot est nul et la métrique vaut uniformément δ (ligne 1). L'algorithme suit alors un processus glouton itératif : à chaque itération, il détermine, parmi toutes les commodités, le plus court chemin d'une source à sa destination (lignes 3 à 7), pousse $\frac{c}{\alpha}$ unités de flot le long de ce chemin (ligne 9) et multiplie la longueur de chaque arc du chemin par $(1+\epsilon\frac{c}{c(e)})$ (ligne 10), où c est la capacité minimale d'un arc du chemin (ligne 8). L'algorithme s'arrête quand toutes les contraintes du dual sont satisfaites, c'est à dire quand la longueur de tous les chemins de $\mathcal{P}$ dépasse 1 (ligne 11).

Correction de l'algorithme

La preuve de correction de l'algorithme 7 est en deux temps. Il faut tout d'abord prouver que le flot calculé respecte les capacités. Pour cela il faut estimer le nombre de fois que l'on pousse du flot sur un arc et choisir α de sorte que les capacités soient respectées. Dans un deuxième temps, il faut prouver l'$(1+\epsilon)$-approximation du flot optimal. Cette preuve utilise le résultat de dualité qui dit que les fonctions objectives du primal et du dual ont le même optimum, le ratio d'approximation est donc borné supérieurement par le ratio entre la fonction objective du primal et celle du dual obtenues par l'algorithme.

Proposition 26 *Le flot calculé par l'algorithme 7 est admissible pour* $\alpha = \log_{1+\epsilon} \frac{1+\epsilon}{\delta}$.

Preuve : Soit un arc e. Chaque fois que $\frac{c(e)}{\alpha}$ unités de flot sont poussées sur e, la longueur de e a été multipliée par au moins $(1+\epsilon)$ avec égalité si la poussée s'est faite en une seule fois : si la poussée a été faite en plusieurs étapes de l'algorithme, il y a eu poussée de $\lambda_1 \frac{c(e)}{\alpha}, \ldots, \lambda_j \frac{c(e)}{\alpha}$ avec $\sum_j \lambda_j = 1$. Dans ce cas, $l(e)$ a été multipliée par $\Pi_j(1+\epsilon\lambda_j) > 1 + \epsilon \sum_j \lambda_j$.

Lorsque l'algorithme 7 termine, $l(e) \leq 1+\epsilon$ puisque du flot n'est poussé sur un arc que s'il apparaît dans un plus court chemin de longueur inférieure à $1+\epsilon$. Comme la longueur initiale de e est δ, il y a eu au plus $\log_{1+\epsilon} \frac{1+\epsilon}{\delta}$ multiplication par $1+\epsilon$ de $l(e)$.

Par conséquent, le flot passant sur e est $f(e) \leq c(e) \frac{\alpha}{\log_{1+\epsilon} \frac{1+\epsilon}{\delta}}$. □

Proposition 27 *L'algorithme 7 calcule une* $(1+\epsilon)$*-approximation du flot maximal.*

Idée de preuve : Pour prouver cette proposition, on utilise le théorème 3 de dualité qui dit que la fonction objective du dual et celle du primal prennent la même valeur à l'optimal. Comme le primal est un problème de maximisation, le dual est un problème de minimisation. Donc le flot maximal est toujours plus petit que le volume du réseau donné par une métrique admissible par le dual et plus grand qu'un flot respectant les capacités. Par conséquent, si l'on borne supérieurement le ratio entre le volume du réseau donné par la métrique que calcule l'algorithme 7 et la valeur du flot qu'il construit, alors on obtient aussi une borne supérieure au ratio entre le flot construit et le flot max. La preuve est technique et apporte peu pour la suite, on peut la consulter dans (Garg et Konemann 1998). □

Complexité de l'algorithme

La preuve de complexité s'appuie sur le même type d'argument que la proposition 26. En effet, là encore il s'agit de borner supérieurement le nombre d'itérations en estimant combien de fois un lien a été allongé. Il est important de noter que la complexité prouvée dans (Garg et Konemann 1998) est fausse et corrigée dans (Fleischer 2000) à l'occasion de communications personnelles entre les auteurs.

Proposition 28 *L'algorithme 7 termine en temps* $O\left(kT_{sssp}\frac{m\log n}{\epsilon^2}\right)$*, où* k *est le nombre de sources de commodités,* $n = |V|$*,* $m = |E|$*, et* T_{sssp} *est le temps de calcul d'un arbre des plus courts chemins depuis un sommet.*

Preuve : Chaque itération coûte le calcul du plus court chemin pour chaque commodité. Cela se fait par k calculs d'arbre des plus courts chemins, un par source distincte. Reste à borner le nombre d'itérations.

L'argument donné pour la preuve de la proposition 26 montre qu'un arc e a été allongé au plus $\log_{1+\epsilon}\frac{1+\epsilon}{\delta}$ fois. De plus, dans le pire des cas, chaque chemin rallongé n'est constitué que d'un seul arc. Il y a donc eu, au plus, $m\log_{1+\epsilon}\frac{1+\epsilon}{\delta}$ itérations.

Pour $\delta = (1+\epsilon)((1+\epsilon)n)^{-\frac{1}{\epsilon}}$, $\log_{1+\epsilon}\frac{1+\epsilon}{\delta} = O(\frac{\log n}{\epsilon^2})$, ce qui donne le résultat. □

Le choix de δ fait dans la preuve est en fait imposé par la preuve d'approximation. Il est aussi à noter que n peut être remplacé par la longueur maximum d'un chemin impliqué dans l'algorithme, mais sans connaissance supplémentaire sur la topologie de G, il est impossible de caractériser cette grandeur autrement qu'en la surestimant par n.

Par ailleurs, il est évident en lisant cette preuve que l'estimation du nombre d'itérations est une borne supérieure très lâche. En effet, il est faux que tous les plus courts chemins ne comportent qu'un seul arc à moins de conditions très particulières. En effet dès qu'il existe deux chemins entre une source et une destination, au moins l'un des deux comporte plusieurs arcs. Étant donné que toutes les longueurs sont initialement fixées à δ et à moins que le second chemin comporte plus de $\frac{1}{\delta}$ arcs, il arrivera forcément un moment où il sera utilisé. Nous verrons d'ailleurs qu'en pratique, si le nombre d'itérations reste grand, il est très largement inférieur à la borne théorique de l'analyse.

5.2.2 Pousser le long du chemin le moins chargé à ϵ près

La complexité de l'algorithme 7 dépend du nombre de commodités différentes. En pratique cette quantité peut croître jusqu'au nombre de sommets dans le graphe. L'algorithme de Fleischer (2000) (algorithme 8) élimine cette dépendance.

Algorithme 8 $(1+\epsilon)$-approximation rapide du multiflot maximum

Entrée: $G=(V,E),\ |V|=n,\ |E|=m,\ C=\{(s_i,t_i),\ i=1\dots k\}\subseteq V^2, \epsilon>0$

Sortie: l métrique sur E t.q. tout chemin $s_i \to t_i$ soit de longueur ≥ 1.

Sortie: f $(1+\epsilon)$-approximation du multiflot maximum de C sur G

1: {Initialisation} $\forall p \in \mathcal{P},\ f(p)=0,\ \forall e\in E,\ l(e)=\delta$

2: {Borne inférieure sur la longueur du SSSP } $\lambda=\delta$

3: **tant que** $\lambda \leq 1$ **faire**

4: **pour tout** $i=1\dots k$ **faire**

5: $P \leftarrow$SSSP $(s_i \to t_i)$

6: **tant que** $l(P)\leq(1+\epsilon)\lambda$ **faire**

7: $c_m \leftarrow \min_{e\in P} c(e)$

8: $f(P)\leftarrow f(P)+c_m/(\log_{1+\epsilon}\frac{1+\epsilon}{\delta})$

9: $\forall e\in P,\ l(e)\leftarrow l(e)(1+\epsilon\frac{c_m}{c(e)})$

10: $P\leftarrow$SSSP $(s_i\to t_i)$

11: $\lambda\leftarrow\lambda(1+\epsilon)$

L'algorithme 8 fonctionne sur le même principe que l'algorithme 7. La principale différence se trouve dans le choix des chemins sur lesquels du flot est poussé, donc dans la boucle entre les lignes 3 et 11. L'idée principale mise en oeuvre ici est qu'il n'est pas utile de pousser du flot le long du plus court chemin, il suffit d'en prendre une $(1+\epsilon)$-approximation.

En conséquence l'algorithme maintient une borne inférieure λ sur la longueur du plus court chemin, initialisée à δ, la longueur initiale d'un arc. Ensuite, l'algorithme prend une commodité, pousse du flot le long des SSSP correspondants tant que ceux-ci sont au plus $1+\epsilon$ fois plus long que λ puis passe à la suivante (ligne 4 à 10). Quand toutes les commodités ont leur SSSP plus long que $(1+\epsilon)\lambda$, cette valeur est une borne inférieure pour la longueur du plus court chemin, λ est mis à jour (ligne 11), et la boucle recommence.

L'intérêt principal de cette technique est que chaque poussée de flot ne nécessite plus

qu'un seul calcul de SSSP et non plus un par commodité, ce qui élimine donc le facteur k dans la complexité.

Correction et complexité

La preuve de la proposition 26 de faisabilité du flot calculé ne prend en compte que l'amplitude d'une augmentation de longueur. Puisque cette amplitude ne change pas, la proposition 26 est valide pour l'algorithme 8. La preuve du facteur d'approximation est elle aussi très similaire.

La complexité de l'algorithme 8 se démontre encore une fois de la même manière que la proposition 28.

Proposition 29 *L'algorithme 8 termine en* $O\left(T_{sssp}\frac{m\log n}{\epsilon^2}\right)$*, où* $n = |V|$*,* $m = |E|$*, et* T_{sssp} *est le temps de calcul d'un arbre des plus courts chemins depuis un sommet.*

Preuve : Chaque poussée de flot nécessite le calcul d'un plus court chemin de la source à la destination de la commodité concernée. Un arc e étant allongé au plus $\log_{1+\epsilon}\frac{1+\epsilon}{\delta}$ fois, dans le pire des cas, il y a au plus, $m\log_{1+\epsilon}\frac{1+\epsilon}{\delta}$ itérations. □

En fait, il y a une dépendance au nombre de commodités qui est cachée. En effet, pour chaque commodité, le calcul du SSSP qui permet de décider qu'il faut passer à la commodité suivante n'amène pas de poussée de flot. Cependant, si le nombre de commodités est d'un ordre inférieur ou égal à celui du nombre d'arcs m, cette dépendance est négligeable. Cette propriété est assurée si l'on suit la remarque faite à la section 3.3 et que l'on regroupe les commodités par sources.

Enfin, en s'inspirant de cette remarque, il est possible de gagner un facteur additif dans le temps de calcul en prenant en compte la topologie du réseau. En effet, si le plus court chemin en nombre d'arcs entre une source et une destination comporte au moins ρ arcs, les $k\log_{1+\epsilon}\rho$ premiers calculs de plus courts chemins sont inutiles. Ces premières itérations peuvent s'éviter de deux manières, soit en connaissant ρ a priori et en initialisant λ à $\rho\delta$, soit en conservant le minimum des longueurs faisant échouer le test ligne 6 puis en mettant λ à jour avec cette valeur. De cette manière, λ n'est plus une borne inférieure sur la longueur d'un plus court chemin mais bel et bien la longueur du plus court chemin

à chaque itération de la boucle des lignes 3 à 11 et, tout en initialisant λ à δ, après k calculs de SSSP, λ vaudra $\rho\delta$. Ce gain, s'il est significatif en pratique, n'est toutefois pas assez important pour modifier la complexité asymptotique de la proposition 8.

Dans la suite, nous nous proposons d'améliorer encore cet algorithme. En effet, la modification apportée à la version de Garg et Konemann (1998) ouvre la porte à l'utilisation de techniques plus efficaces de calcul des plus courts chemins par le biais des algorithmes dynamiques de maintien de la structure de l'arbre des plus courts chemins, largement étudiés dans la littérature. Ensuite, nous proposons des spécialisations des calculs de plus courts chemins aux topologies rencontrées lors de l'optimisation du routage optique présentée plus haut.

5.2.3 Les plus courts chemins dynamiques

L'algorithme de Fleischer (2000) prend comme parti pour améliorer la version de Garg et Konemann (1998) de calculer une séquence de plus courts chemins pour une même commodité ou pour une même source si on regroupe les commodités par source commune. Entre chaque calcul, la longueur des arcs du graphe a été modifiée. Ce type de séquence de calculs a été étudié dans la littérature sous le nom de *calculs de plus courts chemins dynamiques*. Nous nous proposons d'importer ces techniques, notamment celles présentées dans (Frigioni et coll. 2000) afin d'accélérer sensiblement l'algorithme 8.

La motivation principale de l'étude des plus courts chemins dynamiques dans la littérature concerne les graphes dont la longueur d'un arc est augmentée ou diminuée, ou bien quand un arc apparaît ou disparaît. La dynamique ainsi prise en compte peut, par exemple, modéliser celle de réseaux radio mobiles de type *ad-hoc* où la longueur d'un arc modélise l'énergie à dépenser pour envoyer un paquet d'un nœud à l'autre : elle croît à mesure que deux nœuds s'éloignent jusqu'à ce que le lien soit coupé. Dans ce contexte, connaître les trajets de moindre énergie d'un point vers tous les autres nœuds correspond à calculer l'arbre des plus courts chemins issus de ce point et il est important de pouvoir mettre à jour cette structure rapidement. Bien entendu, ce genre d'algorithme s'utilise dans bien d'autres contextes dont les applications ne sont pas aussi directement concrètes. Il est à noter que des études se sont faites et se poursuivent sur la mise à jour

dynamique de la structure des plus courts chemins entre toute paire de nœuds (*All Pair Shortest Paths*), les derniers travaux en date étant dus à Baswana, Hariharan, et Sen (2003). Toutefois ces algorithmes, qui pourraient s'utiliser dans l'algorithme de Garg et Konemann (1998), sont encore préliminaires et trop coûteux par rapport à ce qu'apporte Fleischer (2000). Nous nous concentrons donc sur les algorithmes de mise à jour de la structure d'arbre des plus courts chemins issus d'un nœud.

Les études se fondent, selon les cas, sur la mise à jour de la structure après la modification ponctuelle d'un arc ou bien sur l'analyse amortie d'une séquence de mise à jour consécutive à une séquence de modification d'un arc à la fois. L'idée principale, pour être plus efficace que relancer un calcul complet, consiste à remarquer que seule une partie de l'arbre existant sera impactée par la mise à jour. Par exemple, dans le cas simple de l'augmentation de la longueur d'un arc, la distance d'un sommet à la source ne peut qu'augmenter. Ainsi, les sommets qui ne sont pas dans un sous-arbre issu d'une des extrémités de l'arc modifié n'ont aucune raison de voir leur distance ou leur père dans l'arbre modifiés. Le principe de la plupart de ces algorithmes est donc de repérer et isoler la portion impactée de l'arbre puis d'opérer un calcul complet dans celle-ci. Un bon historique de ces recherches et les meilleurs résultats connus sont rassemblés dans (Frigioni et coll. 2000) sur lequel nous nous fondons pour la suite.

Dans le cas qui nous préoccupe, la situation est sensiblement différente. En effet, chaque mise à jour doit avoir lieu non pas après la modification d'un seul arc, mais après que les longueurs de l'ensemble des arcs d'un plus court chemin ont été modifiées. En appliquant directement les algorithmes de la littérature, il faudrait calculer une mise à jour pour chaque arc modifié, ce qui conduit à des résultats catastrophiques. Dans la suite nous montrons comment tirer parti du fait que le chemin le long duquel les arcs sont allongés est un plus court chemin, qui était donc complètement inclus dans l'arbre des plus courts chemins précédent. Nous proposons une modification de l'algorithme de Frigioni, Marchetti-Spaccamela, et Nanni (2000) pour l'allongement d'un seul arc et nous prouvons que notre version a la même complexité au pire.

L'algorithme de mise à jour

L'algorithme de Frigioni et coll. (2000) pour la mise à jour de l'arbre des plus courts chemins issus d'une source s après l'allongement d'un arc se décompose en deux phases. Dans un premier temps, les nœuds sujets à modification sont marqués *rouges*, *roses* ou *blancs* selon l'impact de l'allongement sur leur place dans l'arbre des plus courts chemins. Un nœud est

- *rouge* si sa distance à s augmente ;
- *rose* si sa distance ne change pas mais s'il change de père dans l'arbre ;
- *blanc* s'il n'est pas impacté par la modification.

Dans un deuxième temps, une procédure prend en entrée une bonne coloration rouge-rose-blanc des nœuds avec les nœuds rouges stockés dans un tas et met à jour l'arbre en traitant ces nœuds par ordre croissant de leur distance à s. Le traitement de ces nœuds s'apparente à l'algorithme de calcul des plus courts chemins de Dijkstra (1959). Il est aussi intéressant de noter que la coloration rouge-rose-blanc n'est qu'un artifice pour la preuve de correction. En effet, puisque la deuxième étape ne prend en compte que les sommets rouges, seul ce marquage est explicite, le marquage blanc consistant à ne rien faire et le rose à simplement changer le père du nœud lors de la phase de coloration. La proposition suivante prouve la validité de ces deux étapes combinées.

Proposition 30 ((Frigioni, Marchetti-Spaccamela, et Nanni 2000)) *Lors de l'augmentation de la longueur d'un arc, si l'on fournit à la deuxième étape un coloriage valide des sommets, on obtient une arborescence des plus courts chemins correcte.*

La proposition 30 signifie que la phase de mise à jour des sommets rouges ne fait pas d'hypothèse sur la cause de la modification, il suffit que le marquage des nœuds soit valide. Afin de prendre en compte l'allongement de tous les arcs d'un chemin, nous pouvons donc nous contenter de donner une procédure de coloration des nœuds, l'algorithme 9, qui prend en entrée l'arbre des plus courts chemins initial et le chemin modifié et produit une bonne coloration rouge-rose-blanc des nœuds. La deuxième phase de la mise à jour reste inchangée et nous invitons le lecteur intéressé à consulter (Frigioni et coll. 2000) pour sa description et les preuves de correction et performance.

Dans la suite, nous utilisons les même notations que Frigioni, Marchetti-Spaccamela, et Nanni (2000) : la distance de s à un nœud u est notée $D(u)$ dans l'algorithme et l'arbre $T(s)$ des plus courts chemins issus de s est implicitement décrit par l'ensemble des pères des nœuds dans $T(s)$, notés $P(u)$ pour tout u. Enfin, la longueur d'un arc est notée l.

Algorithme 9 Marquage des nœuds pour mise à jour de l'arbre des plus courts chemins

Entrée: $T(s)$ un arbre des plus courts chemins issus de s, $s \to t$ un plus court chemin.

Sortie: Une bonne coloration des nœuds rouge.

1: {Initialisation} $\forall y \neq s$ un nœud de $s \to t$, Enfiler(M,$\langle y, D(y)\rangle$).
2: **tant que** Non-Vide(M) **faire**
3: $\quad\langle z, D(z)\rangle \leftarrow$ Extraire-Min(M).
4: $\quad$**si** $\exists\, q \notin M$ voisin non rouge de z t.q. $D(q) + l(q,z) = D(z)$ **alors**
5: $\quad\quad P(z) \leftarrow q \quad \{z$ est rose$\}$
6: $\quad$**sinon**
7: $\quad\quad couleur(z) \leftarrow$ rouge
8: $\quad\quad$**pour tout** v fils de $z \notin M$ **faire**
9: $\quad\quad\quad$Enfiler(M,$\langle v, D(v)\rangle$)

Le principe de l'algorithme 9 est similaire à la procédure de marquage de Frigioni et coll. (2000), seule l'initialisation diffère véritablement. L'algorithme commence par insérer les nœuds du chemin $s \to t$ dans une queue M (ligne 1). Cette queue ne contiendra que les sommets impactés par la modification (les rouges et les roses). Ensuite, tant que M n'est pas vide (ligne 2), il traite le nœud le plus proche de s (ligne 3) et vérifie si sa distance est conservée (ligne 4). Dans ce cas, son père est mis à jour, le sommet est implicitement marqué rose (ligne 5) et tout le sous-arbre issus de ce nœud est implicitement marqué blanc vu qu'il ne sera jamais inséré dans M. Dans le cas contraire, le nœud est marqué rouge (ligne 7) et ses fils qui ne le sont pas déjà sont insérés dans M puisqu'ils doivent au moins être roses (lignes $8, 9$). On peut remarquer que la seule possibilité pour qu'un tel nœud soit déjà dans M est qu'il ait été inséré à l'initialisation et qu'il appartiennent donc à $s \to t$.

Correction

La correction de l'algorithme 9 signifie que la coloration rouge-rose-blanc implicite est valide, c'est à dire que seuls et tous les nœuds rouges voient leur distance à s augmentée par l'allongement des arcs du chemin $s \rightarrow t$. La preuve de correction est une conséquence directe des 4 propriétés suivantes.

($\mathcal{P}_1$) Tout les nœuds du chemin $s \rightarrow t$ sont insérés dans M

($\mathcal{P}_2$) Tout les fils d'un nœud rouge sont insérés dans M, aucun fils d'un nœud rose n'est inséré dans M à moins qu'il ne soit dans $s \rightarrow t$.

($\mathcal{P}_3$) Un nœud est blanc si et seulement s'il n'est jamais inséré dans M.

($\mathcal{P}_4$) Soient u et v 2 nœuds insérés dans M, mais pas nécessairement présents simultanément dans M. Si u est extrait avant v, alors $D(u) < D(v)$.

La propriété ($\mathcal{P}_1$) correspond à l'initialisation de M, ligne 1. La propriété ($\mathcal{P}_2$) correspond aux lignes 4 à 9 et traduit le fait que si un sommet est marqué rouge, alors la distance à s ou le père de ses fils doit changer, ils doivent donc être marqués rose ou rouge et donc être insérés dans M. La propriété ($\mathcal{P}_3$) signifie que si un nœud est rose, alors la distance et le père des nœuds du sous-arbre de ses fils ne changera pas puisque conserver la même structure assure que les distances n'augmentent pas, ce qui est la seule alternative.

Le reste de la correction de la correction vient de la propriété ($\mathcal{P}_4$), la seule qui nécessite une preuve.

Preuve : [de la propriété ($\mathcal{P}_4$)]

On note $t(x)$ la "*date*" à laquelle le nœud x est extrait de M.

($\mathcal{P}_4$) est trivialement vraie lorsque le premier nœud est extrait.

Soit v un nœud. Supposons, par récurrence, que ($\mathcal{P}_4$) est vraie pour tout $u, u' \in S$, l'ensemble des nœuds extraits avant v.

Au temps $t(v)$, $v = \min\{x \in M\}$. Soit $P(v) \in S$ le père de v.

- $\forall u \in S$ t.q. $t(u) < t(P(v))$, ($\mathcal{P}_4$) et le fait que le père de v ait une distance à s inférieure à celle de v impliquent que $D(u) \leq D(P(v)) \leq D(v)$.
- $\forall u \in S$ t.q. $t(P(v)) < t(u)$, par définition de S $t(u) < t(v)$. De plus, puisque v est inséré dans M au plus tard quand $P(v)$ en est extrait (lignes 8 et 9), il s'ensuit que

v est déjà dans M à $t(u)$. Ainsi, u et v sont dans M simultanément à $t(u)$. M étant une file, à $t(u)$ $u = \min\{x \in M\}$, et donc $D(u) < D(v)$.

Conséquemment, $(\mathcal{P}_4)$ est vraie pour $S \cup \{v\}$. □

Dans la suite, nous noterons SSSP le calcul d'un plus court chemin avec l'algorithme de Dijkstra (1959) et USSSP la mise à jour de l'arbre des plus courts chemins avec notre version modifiée de l'algorithme de Frigioni et coll. (2000).

Complexité

Tout d'abord, remarquons que notre version modifiée de l'algorithme de Frigioni et coll. (2000) a la même complexité que l'original. En effet, dans le cas où seul le premier arc du chemin $s \to t$ est modifié, l'ensemble des sommets rouges calculé par le marquage de Frigioni et coll. (2000) contient, dans le pire des cas, l'ensemble calculé par notre algorithme 9. Il s'agit maintenant d'évaluer la complexité de l'algorithme 8 utilisant la procédure d'USSSP.

Malheureusement il n'existe pas de formule close pour la complexité de la procédure d'USSSP en fonction des paramètres "classiques" du réseau. Dans (Frigioni et coll. 2000), la complexité est exprimée en fonction du nombre α de sommets impactés. Cela permet aux auteurs de comparer différents algorithmes de mise à jour entre eux alors qu'il est difficile de dire autre chose que "cette procédure est plus efficace que l'algorithme de Dijkstra (1959)".

Pour évaluer leur algorithme, les auteurs introduisent la notion d'*accounting function*, une sorte de degré maximum réparti. Cette notion permet d'affecter chaque arc à une de ses extrémités en répartissant les arcs pour que chaque nœud n'en "possède" pas trop. En utilisant cette affectation pour définir un parcours des arcs, les auteurs peuvent borner supérieurement la complexité de la procédure de mise à jour par $O(\alpha\beta \log n)$, où β est le nombre maximum d'arcs possédés par un sommet, ce qui dépend beaucoup de la structure du réseau : $\beta \leq 3$ pour les réseaux planaires, $\beta \leq d$ pour les réseaux de degré maximum d, ..., $\beta = O(\sqrt{m})$ pour les réseaux généraux.

Dans certains cas, cette borne supérieure peut excéder la complexité de l'algorithme de Dijkstra (1959), mais, par construction, la complexité de l'algorithme d'USSSP est

toujours plus petite que $O(m + n \log n)$.

L'utilisation de la mise à jour de l'arbre des plus courts chemins dans l'algorithme 8 ne permet d'accélérer que la ligne 10, le calcul de la ligne 5 doit toujours être fait par l'algorithme de Dijkstra (1959). Si l'on regarde l'analyse de performance faite à la proposition 29, on voit que sur les au plus $\frac{\log n}{\epsilon^2} m$ calculs de plus courts chemins, $\frac{\log n}{\epsilon^2} k$ restent des SSSP, les autres pouvant être remplacés par des USSSP. On peut donc exprimer la complexité de l'algorithme 8 en termes des complexités de SSSP et USSSP de la sorte :

$$\frac{\log n}{\epsilon^2} \left(k.\text{SSSP} + (m-k).\text{USSSP}\right).$$

Dans la suite, nous étudions la complexité de l'$(1+\epsilon)$-approximation du multiflot maximum dans le contexte du routage optique dans les réseaux multifibres sans conversion de longueur d'onde. La modélisation faite précédemment permet, lorsqu'il n'y a pas d'équipement de conversion, de spécialiser les algorithmes de calcul et de mise à jour de l'arbre des plus courts chemins et ainsi accélérer l'algorithme 8 de manière importante.

5.2.4 Spécialisation au multiflot du routage optique

Le but de ce travail est d'utiliser l'algorithme 8 pour le dimensionnement des réseaux optiques et notamment comme la procédure centrale de l'algorithme 6 d'approximation du routage optique. En effet, la complexité importante en temps et en espace de la résolution exacte du programme linéaire rend difficile l'optimisation de réseaux de grande taille ou lorsque le nombre de longueurs d'onde disponible devient grand comme, par exemple, pour les réseaux DWDM (de l'ordre de la centaine de longueurs d'onde) et UDWDM (millier). Déjà, intégrer l'algorithme 8 utilisant la mise à jour des plus courts chemins dès que possible permet de sensiblement accélérer les algorithmes 5 et 6 sur de gros réseaux.

Afin d'aller plus loin, nous spécialisons les algorithmes de calcul et de mise à jour des plus courts chemins sur les topologies générées par notre modèle décrit en section 3.3.2 dans le cas où il n'y a pas d'équipement de conversion dans les nœuds. Un exemple de ces topologies est donnée en figure 3.8, page 85.

Calculs de plus courts chemins

Considérons le réseau de flot associé à un réseau optique $G = (V, E)$ avec n nœuds, m liens et supportant w longueurs d'onde devant transporter un ensemble de requêtes de communications C. Le nombre d'arcs (t, D_s_t) est égal au nombre de destinations du nœud s. Dans le cas où il y a au moins une communication entre chaque paire de nœuds, le réseau de flot a donc $O(n^2)$ nœuds et $O(wn^2)$ arcs. Si l'on intègre directement ces quantités dans l'algorithme 8, on obtient que le temps de calcul d'une $(1+\epsilon)$-approximation du multiflot maximum qui nous intéresse est

$$O\left(\frac{\log n}{\epsilon^2} wn^4(w + 2\log n)\right).$$

Cette complexité reflète l'efficacité de l'algorithme de Fleischer (2000) sur notre réseau de flot, mais en supposant que ce réseau a une topologie inconnue. Dans notre cas nous pouvons spécialiser les algorithmes de plus courts chemins en prenant en compte les remarques topologiques suivantes :

- Lorsque l'on calcule un plus court chemin entre S et S', seul les nœuds concernant le flot issu de S sont impliqués. Les autres nœuds "spéciaux", qui concernent le flot issu d'une autre source U, peuvent être tout simplement oubliés dans le calcul. Ainsi, le graphe considéré lors du calcul n'a que $O(nw)$ nœuds et $O(mw)$ arcs, ce qui améliore les calculs des lignes 5 et 10.
- Le nombre d'itérations dépend, lui, toujours du nombre total d'arcs du réseau de flot, $O(wn^2)$.

En intégrant ces deux remarques et l'amélioration du calcul de plus court chemin, la complexité est réduite à

$$O\left(\frac{\log n}{\epsilon^2} n^2w^2(m + n\log nw)\right).$$

Mises à jour

La structure en couches du réseau de flot (voir la figure 3.7 pour un exemple) peut être exploitée afin d'améliorer encore plus nos algorithmes en spécialisant cette fois la mise à

jour de plus courts chemins. En effet, il est facile de se convaincre que lorsque la longueur des arcs est augmentée le long d'un plus court chemin (lignes 8 et 9 de l'algorithme 8), une seule des w couches est modifiée. La mise à jour peut donc être accélérée en n'utilisant l'algorithme d'USSSP que sur une couche puis en mettant à jour les arcs (t_i, D_s_t) et (D_s_t, S') d'une autre manière.

L'algorithme décrit en section 5.2.3, lancé sur la seule couche concernée, prend un temps $O(\alpha\beta \log n) < O(m + n \log n)$. La mise à jour des arcs entre les sommets D_s_t et S' peut s'effectuer grâce à des structures de tas ordonnant les pères potentiels dans l'ordre de leur distance. Cela permet de terminer la mise à jour de l'arbre des plus courts chemins en temps $O(n \log w)$. Au total, la complexité de l'USSSP avec cette stratégie est, au plus, $O(m + n \log nw)$. Il s'en suit que l'algorithme de Fleischer (2000) utilisant nos améliorations permet de calculer les multiflots fractionnaires du routage optique d'un réseau WDM en temps, au plus,

$$O\left(\frac{\log n}{\epsilon^2} n^2 w (m + n \log nw)\right).$$

soit une amélioration d'un facteur $O(w)$ par rapport à la précédente version, et d'un facteur $O\left(\frac{n^4 w^2 \epsilon^2}{\log n}\right)$ par rapport à la résolution du programme linéaire, en $O(n^8 w^3)$.

Expérimentations

Nous avons testé ces algorithmes sur des réseaux réels. Nos expérimentations ont été réalisées sur un PIV, cadencé à 2.2Ghz avec 512MB de mémoire.

Une remarque pratique s'impose. Toutes les complexités données précédemment supposent l'utilisation de l'algorithme de Dijkstra (1959), le plus efficace asymptotiquement. Malgré une moins bonne complexité asymptotique, $O(mn)$, l'algorithme de Bellman-Ford est plus rapide sur des graphes de moins de quelques milliers de sommets. En effet, les tas de Fibonacci utilisés par Dijkstra (1959) impliquent d'importantes constantes cachées. Notre implémentation utilise donc l'algorithme de Bellman-Ford, y compris pour les mises à jour.

Une première série de tests concerne un réseau dorsal américain appelé réseau *pan-américain* constitué de 65 noeuds représentant les grandes agglomérations américaines interconnectées par 78 liens. La matrice de trafic utilisée est constituée de 1305 requêtes

de communication connectant 192 paires de noeuds. Les temps de calculs représentés en figure 5.4, montrent que notre implémentation de l'algorithme 8 permet de traiter des réseaux de grande taille, à l'inverse du solveur de programme linéaire CPLEX. En particulier, il devient possible de traiter des réseaux WDM *denses*, utilisant de quelques centaines à plusieurs milliers de longueurs d'onde par fibre, alors que CPLEX échoue sur le réseau *pan-américain* à partir de ce seuil critique : 13180 secondes sont nécessaires pour la dernière instance admissible pour CPLEX comprenant 99 longueurs d'onde. Au delà, la taille mémoire du programme linéaire dépasse les capacités de la machine.

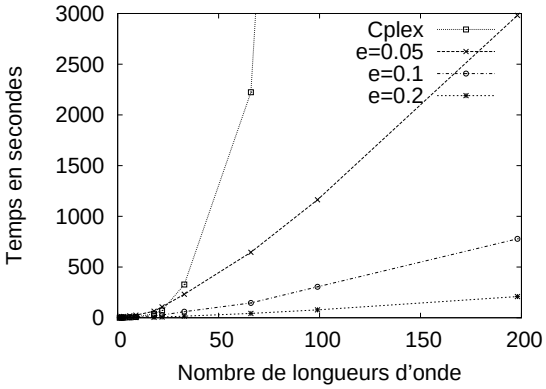


FIG. 5.4: *Temps de calcul. Algorithme 8 et* CPLEX

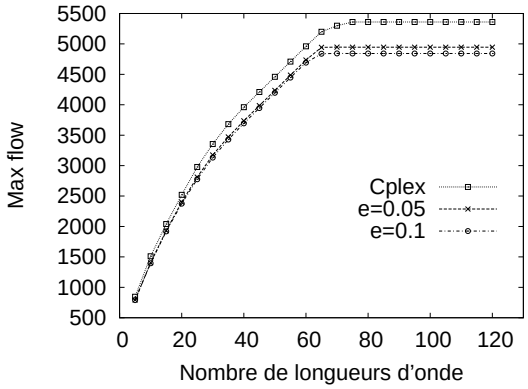


FIG. 5.5: *Flot maximum exact et approché*

La deuxième série d'expérimentations concerne le réseau NSFNET, décrit dans (Swaminathan et Sivarajan 2002), constitué de 14 noeuds et 21 liens symétriques avec une instance de communication de 268 requêtes. Pour que la taille du problème soit conséquente, les requêtes ont été multipliées par 20 et le nombre de fibres par lien à été arbitrairement fixé à 5. La figure 5.5 présente la valeur du flot maximal obtenu par l'algorithme 8 pour $\epsilon = 0.05$ et $\epsilon = 0.1$. Lorsque $\epsilon = 0.1$, la valeur du flot est bien à 10% du flot maximal calculé par CPLEX (différence de 518 unités de flot). Par contre, pour $\epsilon = 0.05$, la valeur du flot devrait se situer à moins de 5% de la valeur optimale, ce qui n'est pas le cas. En effet, lorsque ϵ est trop proche de zéro, des problèmes de précision numérique apparaissent. Ceux-ci, déjà évoqués dans (Fleischer 2000), sont dus à la manipulation simultanée de valeurs d'ordres de grandeur très différents.

La figure 5.6 présente le temps d'exécution pour le calcul du flot maximal pour l'instance I du réseau NSFNET, celle-ci étant multipliée par un facteur k, tandis que la figure 5.7 donne le ratio entre ce temps de calcul et le temps pour l'instance I seule. La figure 5.6, comme la figure 5.4, illustre bien la dépendance linéaire entre le temps de calcul et le nombre de longueurs d'onde disponibles. Par ailleurs, la figure 5.7, montre une dépendance logarithmique à la taille de l'instance : pour un facteur multiplicatif de 20, le temps est multiplié par 4. Notons aussi que lorsque la valeur du flot maximum est atteinte, aux alentours de 60 longueurs d'onde d'après la figure 5.5, le temps n'augmente plus et le ratio Temps(k.I)/Temps(I) reste constant.

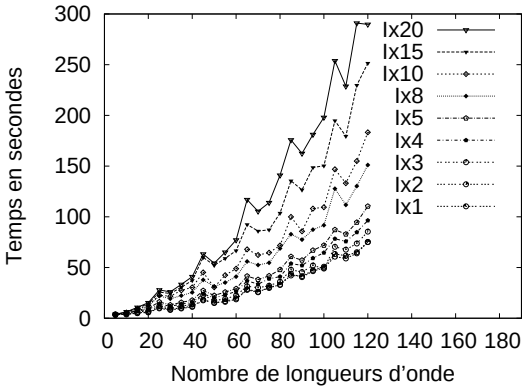


FIG. 5.6: *Temps de calcul pour* $\epsilon = 0.05$ *sur* NSFNET *et son instance I, multipliée par* $k = 1 \ldots 20$

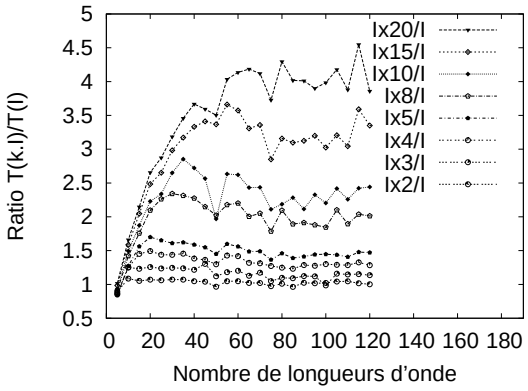


FIG. 5.7: *Ratio Temps(k.I)/Temps(I) pour* $\epsilon = 0.05$

Conclusion et perspectives

Les technologies utilisées dans les réseaux de communication sont nombreuses et en constante évolution. La plupart du temps, étudier des solutions spécifiques à chaque innovation force au recyclage systématique des mêmes idées algorithmiques de base, au détriment d'avancées tant pour la recherche que pour l'ingénierie. Une approche pour éviter cet écueil consiste en une unification transversale des problématiques sous-jacentes, grâce à des modélisations structurelles et algorithmiques pertinentes. Par exemple, des structures combinatoires aussi générales que les graphes ont permis de prendre en compte des caractéristiques des réseaux filaires statiques classiques, et d'en extraire des problématiques algorithmiques fondamentales, telles que le calcul des plus courts chemins ou les réseaux de flots.

Les travaux qui ont été présentés dans cette thèse suivent cette approche en s'intéressant plus spécifiquement au dimensionnement des réseaux optique d'infrastructure. L'optimisation de ces réseaux est essentielle aux opérateurs de télécommunication, qui demandent la garantie d'une exploitation efficace des ressources qu'ils déploient. Nos travaux s'insèrent donc dans l'effort de recherche d'algorithmes efficaces d'approximation, notamment pour les problématiques d'optimisation combinatoire fondamentales que font apparaître les modélisations – structurelles et algorithmiques – des réseaux d'infrastructure.

Nous suivons dans nos travaux une méthodologie générale fondée sur des agrégations d'information. Ainsi, nous traitons nos problèmes par des processus algorithmiques complexes dont le cœur d'optimisation est le plus compact possible, ce qui nous permet d'obtenir des approximations performantes. Nous répondons de la sorte à la demande de garanties sur la qualité de l'optimisation des systèmes de communication installés sur le terrain, de grande taille et ayant tendance à croître rapidement.

Résumé des contributions de la thèse

Nous avons présenté, au chapitre 3, une nouvelle modélisation des réseaux optiques WDM multifibres. Cette modélisation répond à un manque crucial de formalisme, sur lequel nous avions conclu, au chapitre 2, l'état de l'art des modèles et algorithmes pour les réseaux optiques. En ne considérant pas un routage détaillé au niveau des fibres, mais agrégé au niveau des câbles, nous optons pour une nouvelle lecture des contraintes d'affectation de longueurs d'onde fondée sur des conflits de groupe. Nous avons montré que le problème d'affectation de longueurs d'onde s'écrivait alors comme un problème de coloration d'hypergraphes, et développé une écriture du problème de routage optique comme un multiflot dans un réseau de flot auxiliaire.

Nous avons ensuite étudié le problème de coloration de chemins. Ce problème fondamental d'optimisation combinatoire s'applique à un grand nombre de situations, notamment à l'affectation de longueurs d'onde dans les réseaux optiques monofibres. Nous avons montré, au chapitre 4, comment étendre la formulation en couverture du problème de coloration de sommets. Nous nous sommes ensuite concentré sur la relaxation linéaire de ce problème, la coloration fractionnaire, et développé un algorithme polynomial efficace dans les arbres de degré borné, puis, par extension, dans les graphes de largeur arborescente bornée. Nous avons majoré le coût d'une coloration fractionnaire dans les arbres binaires par $\frac{7}{5}$ fois la charge. Nous avons enfin donné un algorithme d'arrondi aléatoire obtenant une $(1 + \frac{5}{3e} + o(1))$–approximation dans les arbres de degré borné, ce qui améliore le meilleur algorithme connu pour ce cas.

Nous avons enfin présenté des avancées algorithmiques pour les problèmes de multiflot entier et fractionnaire dans le chapitre 5. Nous avons donné un algorithme d'arrondi aléatoire incrémental pour l'approximation du multiflot entier. Motivés par le besoin d'un calcul rapide de multiflot fractionnaire pour l'algorithme précédent, nous nous sommes intéressés aux approximations combinatoires de ce problème. En employant des techniques de calcul dynamique des plus courts chemins, nous avons amélioré l'un des meilleurs algorithme de la littérature. Dans le cas des réseaux de flot impliqués dans la modélisation du routage optique du chapitre 3, cet algorithme nous permet d'obtenir une accélération d'un facteur $O\left(\frac{n^4 w^2 \epsilon^2}{\log n}\right)$ par rapport à la complexité moyenne du simplexe sur le programme linéaire correspondant.

Quelques questions ouvertes

À plusieurs reprises dans nos travaux, nous avons constaté une distance considérable entre l'analyse théorique et les résultats expérimentaux des processus d'arrondi aléatoires. En effet, sur des situation pratiques issues de l'optimisation des réseaux optique d'infrastructure, les algorithmes donnent de bien meilleurs résultats que ce que garantissent les analyses probabilistes. Il faut déterminer la part dans cette distance des particularités structurelles des réseaux, et celle de la trop grande généralité des résultats probabilistes que nous utilisons. D'autre part, il convient toujours de se demander si un algorithme a atteint le meilleur ratio d'approximation possible, ou de chercher à l'améliorer. En particulier, les recherches suivantes nous semblent intéressantes.

- Une étude théorique d'approximabilité de la (k, c)–coloration des hypergraphes reste à faire. En effet, l'algorithme de Leighton et coll. (2001) produit des solutions d'une qualité proche de la meilleure dont l'existence est garantie (Srinivasan 1996), mais aucun résultat négatif n'existe sur l'existence d'une amélioration.
- Concernant la coloration de chemins, nous pensons pouvoir améliorer l'algorithme de Auletta et coll. (2000). Celui-ci peut s'interpréter comme un arrondi aléatoire incrémental guidé par une coloration fractionnaire équilibrée, et nous cherchons un algorithme suivant la même méthode, mais en étant guidé par une coloration fractionnaire optimale. Nous butons toutefois sur la difficulté de caractériser les distributions des traces des colorations fractionnaires optimales.
- Le cas du calcul de multiflot fractionnaire est différent. En dépit d'un bon comportement théorique, notre algorithme, comme celui de Fleischer (2000), met en jeu des valeurs extrêmement petites, difficilement exploitables en pratique : sur un réseau à 100 sommets et avec $\epsilon = \frac{1}{100}$, la longueur initiale des arcs est de l'ordre de 10^{-200}. Dans une implémentation usuelle, ces valeurs sont considérées nulles par la machine. Une stratégie augmentant progressivement la précision de calcul nous semble appropriée pour palier à ce problème (Coudert, Rivano, et Roche 2003).

Perspectives de recherche

Nos recherches nous ont amenés à envisager de nombreuses perspectives de recherche, notamment en direction de structures combinatoires et d'algorithmes à même de prendre en compte d'autres types de problématiques d'optimisation et de réseaux.

Une des principales voies qu'il nous semble important d'explorer concerne les réseaux dynamiques. La réalité des réseaux est évidemment dynamique, qu'il s'agisse des réseaux les plus proches des utilisateurs – où la dynamique reflète directement l'activité des terminaux, aléatoire et très sporadique – jusqu'aux réseaux d'infrastructure intercontinentaux – où, du fait des agrégations successives des flux de données, la dynamique est beaucoup plus lissée par des effets statistiques, et peut être quasi déterministe – en passant par les réseaux où le média est lui même dynamique, comme les réseaux radio par exemple. Les problématiques issues de ces considérations dépendent fortement, à première vue, du type de dynamique et de technologie concernées. De nombreuses études sont faites, mais il manque les outils permettant de confronter les approches entre elles et de progresser collectivement, au lieu de recycler des idées similaires dans des contextes légèrement différents.

Dans ce domaine, la réalisation d'une unification des problématiques d'optimisation est donc pour le moins nécessaire. Pour autant, elle n'est pas immédiate. En effet, un prérequis majeur est le développement de notions combinatoires prenant structurellement en compte des aspects dynamiques. Les prémices d'un tel développement apparaissent, notamment, dans la notion de *graphes évolutifs* (Bui Xuan et coll. 2003), ou dans les calculs de *flots dynamiques* (Köhler et Skutella 2002). Ces notions combinatoires et algorithmiques, fondés sur les travaux de Ford et Fulkerson (1958), prennent structurellement en compte la dynamique déterministe d'une topologie. Elles s'appliquent notamment à la modélisation de réseaux dont la dynamique est effectivement déterministe, tels que les réseaux de transports ferroviaires par exemple, ou bien à l'analyse compétitive de protocoles pour des réseaux dont la dynamique est aléatoire (Faragó et Syrotiuk 2001).

Cet effort de recherche doit être poursuivi, au moins dans deux directions spécifiques. D'une part, il faut enrichir l'ensemble d'outils disponibles, pour en rendre l'exploitation opérationnelle possible. D'autre part, il faut prendre en compte structurellement les dynamiques aléatoires. L'optimisation efficace des réseaux de télécommunication le nécessite.

Références

- Agrawal, M., N. Kayal, et N. Saxena (2002). PRIMES is in P. `http://www.cse.iitk.ac.in/news/primality.html`, Indian Institute of Technology Kanpur - India.
- Armand, P., J. C. Gilbert, et S. Jan-Jégou (2001). A feasible BFGS interior point algorithm for solving convex minimization problems. *SIAM Journal on Optimization 11*(1), 199–222.
- Auletta, V., I. Caragiannis, L. Gargano, C. Kaklamanis, et P. Persiano (2001). Sparse and limited wavelength conversion in all-optical tree networks. *Theoretical Computer Science 266*(1-2), 887–934.
- Auletta, V., I. Caragiannis, C. Kaklamanis, et P. Persiano (2000). Randomized path coloring on binary trees. In L. N. in Computer Science (Ed.), *APPROX'00*, Volume 1913, pp. 60–71. Springer-Verlag.
- Ausiello, G., P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, et M. Protasi (1999). *Complexity and Approximation*. Springer.
- Baroni, S., P. Bayvel, R. J. Gibbens, et S. K. Korothy (1999). Analysis and design of resilient multifiber wavelength-routed optical transport networks. *Journal of Lightwave Technology 17*(5), 743–754.
- Baswana, S., R. Hariharan, et S. Sen (2003). Maintaining all-pairs approximate shortest paths under deletion of edges. In *ACM/SIAM SODA'03*.
- Beauquier, B. (2000). *Communications dans les réseaux optiques par multiplexage en longueur d'onde*. Ph. D. thesis, Université de Nice-Sophia Antipolis.
- Beauquier, B., P. Hell, et S. Perennes (1998). Optimal wavelength-routed multicasting. *DAMATH : Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science 84*.

- Bellare, Goldreich, et Sudan (1998). Free bits, PCPs and non-approximability - towards tight results. *SIAM Journal of Computing 27*, 804–915.
- Berge, C. (1983). *Graphes*. Paris : Gauthier-Villars.
- Bermond, J.-C., L. Gargano, S. Pérennes, A. A. Rescigno, et U. Vaccaro (1996). Efficient collective communication in optical networks. In *ICALP'96*, Volume 1099 de *Lecture Notes in Computer Science*, pp. 574–585. Springer-Verlag.
- Bertimas, D. et J. N. Tsitsiklis (1997). *Introduction to Linear Optimization*. Athena Scientific.
- Bouchitté, V. et I. Todinca (2001). Treewidth and minimum fill-in : grouping the minimal separators. *SIAM J. on Computing 31*(1), 212–232.
- Bouklit, M., D. Coudert, J.-F. Lalande, C. Paul, et H. Rivano (2003). Approximate multicommodity flow for WDM networks design. In J. Sibeyn (Ed.), *SIROCCO 10*, Numéro 17 in Proceedings in Informatics, Umea, Sweden, pp. 43–56. Carleton Scientific.
- Bui Xuan, B., A. Ferreira, et A. Jarry (2003). Computing shortest, fastest, and foremost journeys in dynamic networks. *International Journal of Foundations of Computer Science 14*(2), 267–285.
- Caragiannis, I., A. Ferreira, C. Kaklamanis, S. Pérennes, P. Persiano, et H. Rivano (2001). Approximate constrained bipartite edge coloring. In V. L. A. Branstädt (Ed.), *WG'01*, Volume 2204 de *Lecture Notes in Computer Science*, Boltenhagen, Germany, pp. 21–31. Springer-Verlag.
- Caragiannis, I., A. Ferreira, C. Kaklamanis, S. Pérennes, et H. Rivano (2001). Fractional Path Coloring with Applications to WDM Networks. In *ICALP'01*, Lecture Notes in Computer Science, Crete, Greece, pp. 732–743. Springer-Verlag.
- Chlamtac, I., A. Ganz, et G. Karmi (1992). Lightpath communications : An approach to high bandwidth optical WAN's. *IEEE Transactions on Communications 40*(7), 1171–1182.
- Choi, H., S. Subramaniam, et H.-A. Choi (2002). Optimal wavelength assignment algorithms for permutation traffic in multi-fiber WDM ring networks. *Photonic Network Communications 4(1)*, 37–46.

- Choplin, S. (2002). *Dimensionnement de réseaux virtuels de télécommunication.* Ph. D. thesis, Université de Nice - Sophia Antipolis.
- Chvátal, V. (1983). *Linear Programing.* W. H. Freeman and Company.
- Chélius, G., D. Coudert, A. Ferreira, I. Guérin-Lassous, et H. Rivano (2003). Optimisation du contrôle d'accés dans les réseaux radio *ad-hoc* sous un modèle en hypergraphe. Travail en cours.
- Cormen, T. H., C. E. Leiserson, et R. L. Rivest (1990). *Introduction to algorithms.* The MIT Press.
- Coudert, D. et H. Rivano (2002). Lightpath assignment for multifibers WDM optical networks with wavelength translators. In *IEEE Globecom'02*, Taiwan. OPNT-01-5.
- Coudert, D., H. Rivano, et X. Roche (2003). Approximation combinatoire incrémentale du multiflot fractionnaire. Travail effectué au cours du stage de première année d'ÉNS Lyon de Xavier Roche, encadré par D. Coudert et H. Rivano.
- Dantzig, G. (1963). *Linear Programming and Extensions.* Princeton University Press.
- Dijkstra, E. W. (1959). A note on two problems in connection with graphs. *Numer. Math 1*, 269–271.
- Duhamel, C., B. Vatinlen, P. Mahey, et F. Chauvet (2003). Minimizing congestion with a bounded number of paths. In *Algotel'03*, Collioure, France, pp. 155–160.
- Duval, G. (2003). Je t'aime, moi non plus. *Alternative économiques* (215), 39–41.
- Eramo, V. et M. Listanti (2003). Input wavelength conversion in optical packet switches. *IEEE Communications letters 7*(6), 281–283.
- Erlebach, T. et K. Jansen (1997). Call scheduling in trees, rings and meshes. In *HICCS'97*, Volume 1, pp. 221–222. IEEE Computer Society Press.
- Erlebach, T., K. Jansen, C. Kaklamanis, M. Mihail, et P. Persiano (1999). Optimal wavelength routing on directed fiber trees. *Theoretical Computer Science 221*(1–2), 119–137.
- Faragó, A. et V. Syrotiuk (2001). MERIT : A unified framework for routing protocol assessment in mobile ad hoc networks. In *ACM Mobicom 01*, pp. 53–60. ACM.
- Farkas, G. (1896). On the algebraic foundation of the applications of the mechanical principle of fourier. *Mathematikai és Physikai Lapok 5*, 49–54. Hongrois, titre

original : "A fourier-féle mechanikai ekv alkalmazásának algebrai alapja".

- Ferreira, A., J. Galtier, J. Petit, et H. Rivano (2001). Re-routing algorithms in a meshed satellite constellation. *Annales of Telecommunications 56*(3-4), 169–174.
- Ferreira, A., S. Pérennes, A. W. Richa, H. Rivano, et N. Stier (2003a). Models, Complexity and Algorithms for the Design of Multifiber WDM Networks. In *ICT'03*, Papeete, French Polynesia.
- Ferreira, A., S. Pérennes, A. W. Richa, H. Rivano, et N. Stier (2003b). Models, Complexity and Algorithms for the Design of Multifiber WDM Networks. *Telecommunication Systems*. To be published.
- Fleischer, L. (2000). Approximating fractional multicommodity flows independent of the number of commodities. *SIAM J. Discrete Math. 13*(4), 505–520.
- Floriani, L. (2001). *Techniques for the exploratory study of heuristics in wireless network design*. Ph. D. thesis, Université de Nice-Sophia Antipolis – France Telecom R&D.
- Ford, L. et D. Fulkerson (1962). *Flows in Networks*. Princeton University Press.
- Ford, L. R. et D. R. Fulkerson (1958). Constructing maximal dynamic flows from static flows. *Operations Research 23*, 419–433.
- Foschini, G. J. (1996). Layered space-time architecture for wireless communication in a fading environment when using multiple antennas. *Bell Labs Technical Journal 1*(2), 41–59.
- Frigioni, D., A. Marchetti-Spaccamela, et U. Nanni (2000). Fully dynamic algorithms for maintaining shortest paths trees. *Journal of Algorithms 34*(2), 251 – 281.
- Garg, N. (1994). *Multicommodity Flows and Approximation Algorithms*. Ph. D. thesis, Indian Institute of Technology, Delhi.
- Garg, N. et J. Konemann (1998). Faster and simpler algorithms for multicommodity flow and other fractional packing problems. In *IEEE Symposium on Foundations of Computer Science*, pp. 300–309.
- Gargano, L., P. Hell, et S. Pérennes (1997). Colouring paths in directed symmetric trees with applications to WDM routing. In *ICALP'97*, Volume 1256 de *Lecture Notes in Computer Science*, pp. 505–515. Bologna, Italy : Springer-Verlag.

- Golumbic, M. et R. Jamison (1985). The edge intersection graphs of paths in a tree. *Journal of Comb. Theory 38*, 8–22.
- Gonzaga, C. C. (1989). An algorithm for solving linear programming in $O(n^3L)$ operations. In *Progress in Mathematical Programming*, Berlin, pp. 1–28. Springer Verlag.
- Graham, R. L., D. E. Knuth, et O. Patashnik (1994). *Concrete Mathematics*. Addison-Wesley Publishing Company.
- Grötschel, M., L. Lovász, et A. Schrijver (1981). The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica 1*, 169–197.
- Grötschel, M., L. Lovász, et A. Schrijver (1993). *Geometric Algorithms and Combinatorial Optimization* (Second Corrected ed.), Volume 2. Springer-Verlag.
- Hochbaum, D. S. (Ed.) (1997). *Approximation Algorithms for NP-Hard Problems*. PWS Publishing Compagny.
- Hyytiä, E. et J. Virtamo (1999). Wavelength assignment in multifibre in WDM-networks. Rapport technique COST257TD(99)04, Helsinki University of Technology.
- Jansen, K. et H. Zhang (2002). Approximation algorithms for general packing problems with modified logarithmic potential function. In *TCS'02*, pp. 255–266.
- Karapetyan, I. A. (1980). On coloring of arc graphs. *Doklady Akad. Nauk Armianskoi CCP 70*(5), 306–311. In Russian.
- Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. *Combinatorica 4*, 373–395.
- Khachian, L. (1979). A polynomial algorithm in linear programming. *Soviet Mathematics Doklady 20*, 191–194.
- Klee, V. et G. Minty (1972). How good is the simplex algorithm ? *Inequalities-III*, 159–175.
- König, D. (1931). *Graphok és matrixok*, pp. 116–119. Mat. Fiz. Lapok.
- Krishnaswamy, R. et K. N. Sivarajan (2001). Algorithms for Routing and Wavelength Assignment Based on Solutions of LP-Relaxations. In *IEEE Communications Letters*, Volume 5, pp. 435–437. IEEE.
- Kumar, V. (1998). Approximating arc circular colouring and bandwidth allocation in

all-optical ring networks. In *APPROX'98*.

- Kuri, J., N. Puech, M. Gagnaire, et E. Dotaro (2002). Routing and Wavelength Assignment of Scheduled Lightpath Demands in a WDM optical transport Network. In *IEEE ICOCN 2002*, Singapore.
- Köhler, E. et M. Skutella (2002). Flows over time with load-dependent transit times. In *13th Annual ACM-SIAM Symposium on Discrete Algorithms*, San Francisco, USA, pp. 174–183.
- Leighton, T., C. Lu, S. Rao, et A. Srinivasan (2001). New algorithmic aspects of the Local Lemma with applications to routing and partitioning. *SIAM Journal on Computing 31*(2), 626–641.
- Leighton, T., F. Makedon, S. Plotkin, C. Stein, E. Tardos, et S. Tragoudas (1995). Fast approximation algorithms for multicommodity flow problems. *J. Comput. Syst. Sci. 50*(2), 228–243.
- Li, G. et R. Simha (2001). On the Wavelength Assignement Problem in Multifiber WDM Star and Ring Networks. *IEEE/ACM Transaction on Networking 9*(1), 60–68.
- Li, L. et A. Somani (2000). A New Analytical Model for Multifiber WDM Networks. *IEEE JSAC 18*(10), 2138–2145.
- Lovász, L. (1975). On the ratio of optimal integral and fractional covers. *Discrete Mathematics 13*, 383–390.
- Lu, C.-J. (1998). Deterministic hypergraph coloring and its applications. In *RANDOM'98*, Volume 1518 de *Lecture Notes in Computer Science*, pp. 35–46. Springer-Verlag.
- Margara, L. et J. Simon (2000). Wavelength assignment problem on all-optical networks with k fibres per link. In *ICALP'00*, pp. 768–779.
- Margara, L. et J. Simon (2001). Decidable properties of graphs of all-optical networks. In *ICALP'01*, pp. 768–779.
- Margot, F. (2001). Pruning by isomorphism in Branch-and-Cut. In K. Aardal et B. Gerards (Eds.), *Lecture Notes in Computer Science 2081 : 8th IPCO Conference*, pp. 304–317. Springer.

- Mehrotra, A. et M. A. Trick (1996). A column generation approach for graph coloring. *INFORMS Journal on Computing 8*(4), 344–354.
- Motwani, R. et P. Raghavan (1995). *Randomized Algorithms*. Press Syndicate of the University of Cambridge.
- Mémento FT R&D 19 (2002). Les communications optiques du futur. Rapport technique, Conseil scientifique de France Télécom R& D, `http://www.rd.francetelecom.fr/fr/conseil/mento19/index.html`.
- Niessen, T. et J. Kind (1998). The round-up property of the fractional chromatic number for proper circular arc graphs. *Journal of Graph Theory*.
- Plotkin, S. A., D. B. Shmoys, et E. Tardos (1995). Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research 2*, 257–301.
- Porto (2000). Planification et Optimisation des Réseaux de Transport Optiques. Document soumis à autorisation légale de France Télécom R&D. Projet RNRT entre le projet CNRS/I3S/INRIA Mascotte, Alcatel Corporate Research Centre et France Télécom R&D.
- Qiao, C. et M. Yoo (1999). Optical burst switching (obs) - a new paradigm for an optical internet. *J. High Speed Networks 8*(1), 69–84.
- Raghavan, P. (1994). Probabilistic construction of deterministic algorithm : Approximating packing integer programs. *Journal of Computer and Systems Sciences 38*, 683–707.
- Ramamurthy, B. et B. Mukherjee (1998). Wavelength conversion in WDM networking. *IEEE Journal on Selected Areas of Communication 16*(9), 1061–1073.
- Reeves, C. (2003). Landscapes of modern heuristic search. In EURO/INFORMS *Istambul'03*, Istambul, Turquie, pp. 23. Tutorial.
- Ritzenthaler, S. (2003). What are the trends ? alcatel's vision. In *Algotel'03*, Banyuls-sur-mer. Exposé invité `http://dept-info.labri.u-bordeaux.fr/algotel03/CameraReady/Ritzenthaler-slides.pdf`.
- Rivano, H. (2001). Planification de réseaux optiques WDM k-fibres. In *AlgoTel'01*, Saint Jean de Luz, France, pp. 41–46. INRIA.

- Saad, M. et Z.-Q. Luo (2002). A Lagrangean Decomposition Approach for the Routing and Wavelength Assignment in Multifiber WDM Networks. In *IEEE Globecom'02*, Taiwan. OPNT-05-2.
- Schrijver, A. (2000). A course in combinatorial optimization. `http://www.cwi.nl/~lex/files/dict.ps`, CWI and Department of Mathematics, Amsterdam, The Netherlands.
- Shahrokhi, F. et D. Matula (1990). The maximum concurrent flow problem. *Journal of the ACM 37*, 318 – 334.
- Sharma, V. et E. Varvarigos (2000). An analysis of limited wavelength translation in regular all-optical WDM networks. *IEEE/OSA Journal of Lightwave Technology 18*(12), 1606–1619.
- Shor, N. Z. (1970). Utilization of the operation of space dilation in the minimization of convex functions. *Cybernetics 6*, 7–15.
- Srinivasan, A. (1996). An extension of the lovász local lemma, and its applications to integer programming. In *ACM-SIAM SODA'96*, pp. 6–15.
- Subramaniam, S., M. Azizoglu, et A. Somani (1999). On optimal converter placement in wavelength-routed networks. *ACM/IEEE Transactions on Networking 7*(5), 754–766.
- Swaminathan, M. et K. Sivarajan (2002). Practical routing and wavelength assignment algorithms for all optical networks with limited wavelength co nversion. In *IEEE ICC*, New-York City. IO4 - 4.
- Tarjan, R. E. (1985). Decomposition by clique separators. *Discrete Mathematics 55*(2), 221–232.
- Togni, O. (2000). Placement de convertisseurs de longueur d'onde dans les réseaux optiques. In *Algotel'2000*, pp. 35–40.
- Tucker, A. (1975). Coloring a family of circular arcs. *SIAM Journal of Applied Mathematics 29*(3), 493–502.
- Union Européenne (2002). 8th report on the implementation of the telecommunications regulatory package. Rapport technique, Information Society, `http://europa.eu.int/information_society/topics/telecoms/`

implementation%/annual_report/8threport/index_en.htm.

- Wang, W., L. Rau, et D. J. Blumenthal (2002). 40 gbps all-optical wavelength conversion using xpm in a distributed fiber raman amplifier. In *Conference for Optical Amplifiers and their Applications (OAA '02)*, Vancouver, Canada.
- Wilhelm, W. E. (2001). A technical review of column generation in integer programming. *Optimization and Engineering 2*, 159–200.
- Xu, L., H. G. Perros, et G. N. Rouskas (2001). A survey of optical packet switching and optical burst switching. *IEEE Communications 39*(1), 136–142.
- Young, N. (1995). Randomized rounding without solving the linear program. In *ACM/SIAM SODA'95*.
- Young, N. (2001). Sequential and parallel algorithms for mixed packing and covering. In *FOCS'01*, pp. 538–546.
- Yu, J., X. Zheng, C. Peucheret, A. T. Clausen, H. N. Poulsen, et P. Jeppesen (2000). 40-Gbit/s all-optical wavelength conversion based on a nonlinear optical loop mirror. *IEEE/OSA Journal of Lightwave Technology 18*(7), 1001–1006.
- Yudin, D. et A. Nemirovskii (1977). Informal complexity and efficient methods for the solution of convex extremal problems. *Matekon 13*, 25–45.
- Zhang, X. et C. Qiao (1998). Wavelength assignment for dynamic traffic in multi-fiber WDM networks. In *ICCCN'98*, pp. 479–585.