Université Joseph Fourier, Grenoble

École Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique

Thèse d'informatique
pour l'obtention du grade de docteur de l'Université Joseph Fourier

préparée au laboratoire G-SCOP
Institut National Polytechnique de Grenoble,
Université Joseph Fourier,
CNRS, UMR 5272

Routages optimaux : tours, flots et chemins

présentée par Guyslain Naves

soutenue le 11 janvier 2010

Jury :

Nadia Brauner Vettier (Université Joseph Fourier, Grenoble),	Examinateur
Christoph Dürr (École Polytechnique, CNRS, Palaiseau),	Examinateur
Jean Fonlupt (Université Paris 6),	Rapporteur
Thomas Liebling (École Polytechnique Fédérale de Lausanne),	Examinateur
Jens Vygen (Université de Bonn),	Rapporteur
András Sebő (Laboratoire G-SCOP, CNRS, Grenoble),	Directeur de thèse

Remerciements

Le temps est venu de clore ces vingt années d'études, en remerciant tous ceux dont j'ai croisé le chemin. Le mien a une origine, ma famille, dans laquelle j'ai toujours trouvé tout le soutien nécessaire pour pouvoir avancer sereinement. J'accorde une pensée spéciale à Alban.

Mes études en informatique ont véritablement débuté à l'École Normale Supérieure de Cachan. J'y ai rencontré des chercheurs débordant d'enthousiasme, renforçant ma volonté de devenir l'un d'eux. En particulier, Hubert Comon-Lundh fut d'un dévouement extraordinaire envers ses étudiants, consacrant pour nous plus de temps qu'il n'est raisonnable. C'est aussi à Hubert que je dois d'avoir rencontré András. J'ai bénéficié à Cachan de la saine émulation et de la convivialité de notre petite promotion, avec Lisa, Mathilde, Arnaud, Basile, Bruno, Clément, Philippe et Thibaud. Mes échanges avec Arnaud n'ont fait qu'augmenter au fur et à mesure de l'éloignement de nos sujets de recherches. J'espère qu'il me pardonnera de l'avoir mis en retard dans sa thèse suite aux très nombreuses questions que je lui ai posées.

Pendant ma scolarité à Cachan, j'ai effectué plusieurs stages de recherche. Merci au professeur Liebling, à Patricia Bouyer et à Nicolas Markey pour votre accueil et pour m'avoir proposé des sujets correspondant à mes goûts. C'était pour moi l'occasion de rencontrer beaucoup de futurs collègues. Heureusement, les arguments de décroisement ne s'appliquent pas aux vies humaines, ce sera un plaisir de revoir chacun d'entre eux.

En arrivant à Grenoble pour commencer cette thèse, je fus chaleureusement accueilli, et plus particulièrement par Nadia Brauner qui s'est empressée de me faire résoudre ses problèmes. L'ambiance amicale et studieuse a été un facteur déterminant pour le déroulement de ma thèse. Yann Kieffer partage mes goûts pour l'aspect logique de l'informatique théorique. Nos discussions ont été très enrichissantes, et pas seulement celles concernant la dégustation des sushis.

Vincent Jost a quitté Grenoble alors que j'arrivais. Cela ne nous a pas empêché de regarder ses problèmes de *freakombinatorics*, comme il les appelle. Je ne le regrette pas puisque notre collaboration a été fructueuse, et n'en est probablement qu'à ses débuts. Vincent ne possède pas seulement les compétences qu'il me manque (et il y en a beaucoup), mais surtout de grandes qualités humaines.

András n'a pas le temps. Il a toujours cinq articles en retard au moins, doit monter des projets et discuter avec ses invités. Il a pourtant fait un grand sacrifice en m'acceptant comme étudiant. Et il a parfaitement assumé son rôle, malgré mon impatience, mes attentes contradictoires et mon constant

désaccord avec chacun de ses arguments. Je lui dois presque tout ce que je sais en optimisation combinatoire, et il m'a aussi beaucoup appris en dehors des maths. J'espère qu'il osera reprendre des étudiants après moi, car j'ai beaucoup apprécié sa façon de faire.

Le passage d'un chemin n'a pas que des conséquences locales ; nous ne sommes pas seulement influencés par les personnes que nous rencontrons. Je dois beaucoup en particulier à László Lovász et son livre d'exercices [35]. J'aimerais avoir plus de temps à y consacrer. J'ai aussi beaucoup appris de Lex Schrijver, András Frank et beaucoup d'autres.

Je remercie les membres du jury pour avoir répondu à notre invitation et pour leur cordialité. Leurs remarques ont contribuée à l'amélioration de ce document. Je ne sais pas quelle est la suite de ma trajectoire, mais mon prochain sommet est l'université McGill de Montréal. Merci à Bruce Shepherd pour l'invitation.

Table des matières

Avance sur ton chemin, car il n'existe que par ta marche.

Saint Augustin

LE MOINE : *Le Moine meurt mais ne s'enfuit pas !*
CORTO MALTESE : *D'accord, alors meurs !*
LE MOINE : *Le Moine fuit mais ne meurt pas !*
CORTO MALTESE : *Dame, il y aurait bien une solution... Meurs à moitié, et rends-toi à moitié...*

La Ballade de la mer salée, Hugo Pratt

Première partie

Théorie des flots et optimisation combinatoire

Avant-propos.

Cette thèse porte sur deux sujets, relativement distincts, bien que traitant tous deux de graphes et de connexité. Ils s'inscrivent dans le cadre de l'optimisation combinatoire, branche des mathématiques discrètes. Cela peut expliquer au lecteur informaticien que malgré l'intitulé "thèse d'informatique" de ce document, nous sommes plus intéressés par les aspects mathématiques des objets que nous étudions, que par les aspects d'algorithmique et de complexité. Pour autant, l'auteur de ce travail étant informaticien de formation, le lecteur mathématicien pourrait estimer que cette thèse est fortement imprégnée d'informatique théorique. Ce qui n'est pas contradictoire si nous considérons l'informatique théorique comme un vaste sous-ensemble des mathématiques.

Le premier sujet porte sur les multiflots entiers. Ce travail a été en partie réalisé avec la collaboration active d'András Sebő, directeur de cette thèse, en particulier sur le recensement et la classification des résultats connus avant cette thèse. Notre principal apport concerne l'étude de la complexité du problème du biflot entier dans les graphes planaires.

Le second sujet tourne autour d'un problème original, posé par Vincent Jost avec qui nous avons travaillé, sur les tours graphiques. Nos réponses sont encore modestes, et la richesse du problème nous laisse espérer des résultats très intéressants pour les prochaines années.

Chacun de ces sujets est traité dans une partie indépendante. Nous avons fait précéder le tout par un chapitre introductif apportant les outils communs nécessaires, dans une présentation inhabituelle. Plutôt que de lister les définitions et théorèmes utiles, nous avons préféré les motiver, et pour coller au cadre de cette thèse, ces motivations proviennent de la théorie des flots. Le lecteur averti pourra survoler ce chapitre pour vérifier les notations que nous avons choisies, au demeurant assez classiques, alors que le lecteur novice devrait, souhaitons-le, y trouver les intuitions et les fondamentaux requis.

0.1 Graphes et connexité

En introduction de notre article [46], nous écrivions à propos de la théorie des flots :

> The field has been developed in parallel with the tools of optimization, polyhedral combinatorics and graph theory [1].

Nous nous proposons donc de commencer cette thèse par un éclairage mutuel de la théorie des flots, voire plus généralement les problèmes de connexité, et des méthodes générales de l'optimisation combinatoire. Cela nous permet d'introduire les notions utilisées par la suite, tout en procurant au lecteur un point de vue que nous espérons original et intéressant.

Dans son acceptation la plus courante, l'*optimisation combinatoire* est la branche des mathématiques vouée essentiellement à la résolution des problèmes de la forme suivante :

Problème d'Optimisation Combinatoire
Entrée : $\mathcal{H}$ famille de sous-ensembles de E fini, $f : E \to \mathbb{R}$ calculable.
Sortie : $H \in \mathcal{H}$ tel que $f(H) = \sum_{e \in H} f(e)$ est maximal.

Précisons que $\mathcal{H}$ n'est pas donné explicitement par un liste d'ensembles, mais plus généralement par une description logique *concise*. Typiquement, $\mathcal{H}$ sera défini comme l'ensemble des sommets d'un graphe G vérifiant une propriété précise, par exemple être un stable, l'entrée serait donc juste une description du graphe dans ce cas. Remarquons dans l'exemple précédent cette notation usuelle en optimisation combinatoire : soit une fonction f à valeurs réelles, pour tout $U \subseteq \mathrm{Dom}(f)$, nous notons $f(U) = \sum_{u \in U} f(u)$.

Avant de donner la définition d'un graphe, précisons quelques notions de complexité. Un problème de décision est la donnée d'un langage $L \subset \Sigma^*$ sur un alphabet fini Σ. Résoudre le problème, c'est donner une méthode formelle pour ce langage (un *algorithme*), permettant de déterminer par un calcul fini si un mot quelconque donné $w \in \Sigma^*$ appartient à L. La *complexité* (en temps) d'un algorithme est la fonction associant à chaque mot le temps de calcul nécessaire pour déterminer si ce mot est dans le langage, et nous choisissons d'étudier les complexités de manière asymptotique, par rapport à la longueur des mots, généralement en donnant des bornes supérieurs.
Un algorithme est *polynomial* si sa complexité est ainsi bornée par un po-

1. Ce domaine s'est développé en parallèle avec les méthodes de l'optimisation, de la combinatoire polyédrale et de la théorie des graphes

lynôme. Les problèmes admettant un algorithme polynomial forment par définition la classe de complexité P. Autrement dit, un problème est dans P s'il est possible de le résoudre en temps polynomial.
La classe de complexité NP est la classe des problèmes certifiables en temps polynomial : il existe un algorithme tel que pour tout mot $w \in \Sigma^*$, $w \in L$ ssi il existe un mot y_w tel que l'algorithme accepte le mot (w, y_w) en temps polynomial par rapport à la taille de w. En langage naturel, y_w est un certificat, une preuve du fait que w appartient au langage, et cette preuve est raisonnablement courte. Une des grandes questions ouvertes en mathématiques est de déterminer si l'inclusion P $\subseteq$ NP est propre. Un langage est par définition dans coNP si son complémentaire est dans NP.
Notons $L(\Pi)$ le langage définissant le problème Π. Un problème $\Pi \in$ NP est NP-complet, si tout problème Π' de NP s'y réduit : pour tout $w \in \Sigma^*$, il existe $y \in \Sigma^*$ calculable polynomialement tel que $w \in L(\Pi')$ ssi $y \in L(\Pi)$. Pour montrer qu'un problème est NP-complet, il suffit de montrer qu'un autre problème NP-complet s'y réduit.
Retenons que P est habituellement admis comme étant la classe des problèmes qu'il est possible de résoudre, et que les problèmes NP-complets ne sont pas susceptibles d'être résolus efficacement. Cette vision (algorithmique) est criticable (et de plus en plus critiquée), mais est par certains côtés suffisamment raisonnable pour lui admettre une légitimité. Nous mettrons néanmoins un sens plus mathématique sur ces notions par la suite, justifiant leur rôle pour déterminer la complexité *mathématique* d'un problème.
Pour clore cette partie sur la complexité, précisons que tous nos problèmes peuvent être décrits naturellement par des langages, de sorte que les mots sont de taille linéaire en la taille (calculée usuellement) des objets mathématiques étudiés. Le modèle calculatoire choisi est le modèle à *mémoire à accès direct*. Nous faisons ainsi abstraction des formalismes nécessaires en théorie de la complexité.

Définissons maintenant les graphes, objet central dans cette thèse. $G = (V, E)$ est un graphe si V est un ensemble fini, et E est un multi-ensemble de paires d'éléments de V. Les éléments de V sont appelés *sommets*, et ceux de E sont appelés *arcs*. Les graphes sont aisément représentables par des dessins : chaque sommet est représenté par un point, et un arc (u, v) est représenté par une flèche du point représentant u vers celui représentant v. Pour un arc $e = (u, v)$ nous disons que e est un arc *sortant* de u, et un arc *entrant* de v. Les sommets u et v sont dits adjacents s'il existe un arc (u, v) ou (v, u). Nous noterons couramment uv pour l'arc (u, v). Les arcs sortants ou entrants de u sont dits *incidents* à u. Nous considérerons seulement des

graphes sans boucle : une *boucle* est un élément $(v, v) \in E$. On appelle *graphe simple* les graphes sans boucle tels que E est un ensemble : toute paire de sommets est représentée au plus une fois dans E. Il arrive souvent que l'ordre des éléments d'un arc n'ait pas d'importance : G est alors dit *non-orienté*, par opposition à *orienté* (nous appelons aussi *digraphe* les graphes orientés), et (u, v) et (v, u) sont alors identifiés (de manière équivalente, E a ses éléments dans $\binom{V}{2}$, l'ensemble des paires de V). Dans un graphe non-orienté, nous appelons *arêtes* les arcs (picturalement, nous dessinons des traits et non des flèches). Pour un graphe G orienté, nous appelons *graphe non-orienté sous-jacent* à G le graphe dont les arêtes sont les arcs de G. G est biparti s'il existe une partition A, B de V tel que $E \subseteq A \times B$.
Quelques graphes non-orientés particuliers interviennent régulièrement. Nous notons K_i la *clique* à i sommets, S_i le *stable* à i sommets et C_i le *cycle* à i sommets, définis par :

- $K_i = (\llbracket 1, n \rrbracket, \{uv \ : \ u, v \in V, u \neq v\})$,
- $S_i = (\llbracket 1, n \rrbracket, \emptyset)$,
- $C_i = (\llbracket 1, n \rrbracket, \{1n\} \cup \{(i, i+1) \ : \ i \in \llbracket 1, n-1 \rrbracket\})$, pour $i \geq 3$.

Le complémentaire d'un graphe simple $G = (V, E)$ est le graphe $\overline{G} = (V, \{uv \notin E\})$. Prendre le complémentaire est une involution : $G = \overline{\overline{G}}$. Le complémentaire de K_i est S_i.

L'une des premières questions que nous pouvons nous poser en présence d'un graphe (en toute généralité orienté), et l'un des premiers problèmes étudiés dès le XVIII$^{\text{ième}}$ siècle, est de déterminer s'il est possible d'aller d'un sommet à un autre par une suite d'arcs. C'est une généralisation du problème du labyrinthe : comment trouver un chemin vers la sortie, et si possible un plus court chemin ? Cela va constituer notre premier problème d'optimisation combinatoire, et probablement l'un des plus importants. Définissons donc une *marche* P d'un graphe comme une séquence d'arcs de la forme $u_0u_1, u_1u_2, \ldots, u_{n-1}u_n$. Si les arcs sont distincts, nous parlons de *chemin*. u_0 et u_n sont appelés *extrémités* du chemin, et $u_iu_{i+1}, u_{i+1}u_{i+2}, \ldots, u_{j-1}u_j$ est un *sous-chemin* de P. Dans le cas orienté, u_0 est appelé *origine* ou *source* du chemin, et u_n est appelé *destination* ou *puits*, on appelle (u, v)-chemin un chemin d'origine u et de destination v. La *longueur* d'un chemin est le nombre d'arcs le composant. Les adjectifs *court* et *long* font référence à la longueur des chemins. Une marche est *fermée* si ses deux extrémités sont identiques. Si en plus c'est un chemin, il s'agit d'un *cycle*. Nous noterons P^{-1} le chemin $u_nu_{n-1}, \ldots, u_2u_1, u_1u_0$ du graphe $\overleftarrow{G} = (V, \{uv \ : \ vu \in E(G)\})$, et si $Q = v_0v_1, \ldots v_{p-1}v_p$ est un chemin avec $v_0 = u_n$, PQ est le chemin

$u_0u_1, \ldots, u_{n-1}u_n, v_0v_1, \ldots, v_{p-1}v_p$. Un (A, B)-chemin, pour $A, B \subseteq V$ est un chemin dont l'origine appartient à A et la destination à B.
Pour tout objet combinatoire $\mathcal{H}$ dans un graphe (en particulier les chemins), nous noterons $V(\mathcal{H})$ et $E(\mathcal{H})$ les ensembles de sommets et d'arcs respectivement, couverts par cet objet.

Formellement, le problème du labyrinthe s'écrit :
PLUS COURT CHEMIN
Entrée : G un graphe, u et v deux sommets de G.
Sortie : P un plus court (u, v)-chemin de G

Parfois, l'existence d'une solution de coût suffisamment faible nous suffit. Nous distinguons problème d'optimisation (trouver la meilleure solution) et problème d'existence (trouver une solution dont le coût ne dépasse pas la borne, s'il en existe). Les deux formulations sont en fait équivalentes.

La théorie des flots traite d'un aspect de la connexité dans les graphes, et généralise le problème de trouver un chemin. Un graphe G est dit *connexes* si pour toute paire de sommets u et v de G, il existe un (u, v)-chemin dans le graphe non-orienté sous-jacent de G, et *fortement connexes* s'il existe un (u, v)-chemin dans G. La connexité n'est pas une notion typique de la théorie des graphes. En particulier, la connexité est un aspect fondamental de la topologie. Mais notre intérêt étant dans l'étude d'ensembles finis, la connexité de ces ensembles se ramène à celle de leur graphe d'adjacence : le graphe défini sur cet ensemble, les arcs représentant l'adjacence des éléments.

Informellement, un (s, t)-flot (entier) est un ensemble de (s, t)-chemins disjoints (disjoints dans le sens où les ensembles d'arcs utilisés par chaque chemin sont disjoints). s est appelé *source* ou *origine* du flot, et t est appelé *puits* ou *destination*. L'utilisation de la même terminologie que pour les chemins est justifiée par le fait qu'un chemin n'est qu'un cas particulier de flot.

Plus généralement, nous nous donnons une fonction entière sur les arcs, déterminant la *capacité* (intuitivement, il s'agit plutôt d'un débit maximal) de chaque arc. Dans ce cas, par chaque arc peuvent passer autant de chemins que la capacité de cet arc. Définissons alors précisément le problème du flot. Soit G un graphe, et prenons s et t deux de ses sommets. Notons $\mathcal{P}_{st}$ l'ensemble des (s, t)-chemins de G :

Flot Entier Maximum
Entrée : G un graphe, s et t deux sommets de G, $c : E(G) \to \mathbb{N}$.
Sortie : Maximiser $|P|$, pour P multi-ensemble de (s,t)-chemins, tels que tout arc $e \in E(G)$ est dans au plus $c(e)$ chemins de P.

En notant le multi-ensemble comme un vecteur caractéristique $\chi \in \mathbb{N}^{\mathcal{P}_{st}}$ sur les chemins de $\mathcal{P}_{st}$, la contrainte énoncée s'écrit :

$$\text{Pour tout } e \in E(G), \quad \sum_{e \in P \in \mathcal{P}_{st}} \chi_P \leq c(e)$$

Plus tard, nous verrons des flots multi-commodités : soient $(s_1,t_1),\ldots,(s_k,t_k)$ plusieurs paires de sommets d'un graphe G. Un multiflot entier est un multi-ensemble de chemins dans $\mathcal{P}_{s_1t_1} \cup \ldots \cup \mathcal{P}_{s_kt_k}$, tel que chaque arc e appartient à au plus $c(e)$ de ces chemins. Dans cette thèse, nous prenons le parti de traiter seulement des problèmes d'existence, qui contiennent suffisamment de questions ouvertes. Nous noterons H le graphe des arcs de demandes $(s_i,t_i)_i$:

Multiflot Entier
Entrée : G un graphe, H graphe avec $V(H) \subseteq V(G)$, $r : E(H) \to \mathbb{N}$, $w : E(G) \to \mathbb{N}$.
Sortie : Existe-t-il un multi-ensemble de chemins de G contenant $r(uv)$ (u,v)-chemins pour tout $uv \in E(H)$, tel que tout arc $e \in E(G)$ est utilisé par au plus $w(e)$ de ces chemins ?

0.2 Programmation linéaire et circulations

L'une des idées fondamentales du domaine est de formuler les problèmes d'optimisation combinatoire comme des programmes mathématiques, en particulier des programmes linéaires. Citons par exemple A. Schrijver dans son livre [57] : "linear programming forms the hinge in the history of combinatorial optimization" [2]. Cette idée a été utilisée très tôt par Hitchcock [24] pour résoudre le problème du transbordement, qui est un cas particulier de flot.

Nous supposerons que le lecteur possède les connaissances d'algèbre linéaire nécessaires à la compréhension de ce qui vient. Un *polyèdre* est un sous-ensemble de $\mathbb{R}^n$ de la forme $P = \{x \ : \ Ax \leq b\}$ avec A matrice $m \times n$

2. l'introduction de la programmation linéaire est la charnière de l'histoire de l'optimisation combinatoire

et b vecteur colonne de dimension m. Une inégalité $ax \leq \beta$ est *valide* pour P si elle l'est pour tout point de P. $\mathcal{H} = \{x \ : \ ax \leq \beta\}$ est un *hyperplan support* de P si $ax \leq \beta$ est valide pour P et $P \cap \mathcal{H} \neq \emptyset$. Une *face* de P est l'intersection de P avec un de ses hyperplans support. Si une face contient un seul élément, celui-ci est appelé *sommet* de P. Un *point extrême* v d'un convexe Q est un point n'appartenant pas à l'enveloppe convexe $\text{conv}(Q-v)$ des autres points de Q. Un polyèdre est dit *de plein rang* si son intérieur est non-vide. Un *polytope* est l'enveloppe convexe d'un ensemble fini de points de $\mathbb{R}^n$. Les polytopes sont les polyèdres bornés. Un polyèdre est *entier* si chacune des ces faces contient un point entier (à coordonnées entières). En particulier, un polytope entier est un polytope dont tous les sommets sont entiers. Pour un point x^0 d'une face de $\{x \ : \ Ax \leq b\}$, les inégalités *serrées* (ou *actives*) sont les inégalités $a_i x \leq b_i$ du système $Ax \leq b$ atteintes avec égalité par x^0. Lorsqu'il n'y aura pas d'ambiguïté, pour un vecteur colonne c, nous omettrons la notation de la transposition en vecteur ligne c^T, et noterons donc simplement c.

Un programme linéaire est un problème d'optimisation de la forme suivante :

$$\max\{cx \ : \ Ax \leq b\}$$

où $A \in \mathbb{R}^{m \times n}$ est une matrice, $b \in \mathbb{R}^m$ et $c \in \mathbb{R}^n$ sont des vecteurs. x est un vecteur d'inconnues. Il s'agit donc de trouver une solution d'un système d'inéquations linéaires, maximisant une forme linéaire donnée. Géométriquement, nous cherchons donc un point dans un polyèdre (c'est-à-dire une intersection d'hyperespaces), qui soit le plus possible dans la direction donnée par le vecteur c. S'il existe un tel point (ce n'est pas toujours le cas, le polyèdre pouvant être vide ou infini), il appartient à une de ses faces.

Dans la définition de FLOT MAXIMUM que nous avons donné, les contraintes sont naturellement des inégalités linéaires. Notons provisoirement $\mathcal{P}$ l'ensemble des chemins de G, et réécrivons précisément :

$$\begin{array}{lll} \max \quad 1x \quad \text{tel que} & & \qquad (1) \\ \quad \sum_{P \in \mathcal{P}, e \in P} x_P \leq w(e) & (\forall e \in E(G)) & \\ \quad x_P \geq 0 & (\forall P \in \mathcal{P}) & \end{array}$$

Dans cette thèse, nous noterons indifférement 1 pour le réel ou les vecteurs dont tous les coefficients sont 1. La forme linéaire objectif de (1) est donc la somme des coefficients de x, ce qui correspond bien au nombre de chemins

pris. Les contraintes avaient déjà été formulées ainsi, seul est ajouté la contrainte de positivité, puisque χ était le vecteur caractéristique d'un multi-ensemble.

Dans cette formulation (1), les solutions peuvent être rationnelles : il s'agit d'une version *relaxée* du problème de flot, *a priori* plus facile à résoudre. Un *programme linéaire en nombres entiers* est un programme linéaire auquel est ajouté la contrainte d'intégralité : le vecteur inconnu est choisi entier. Par exemple, pour les flots :

$$\begin{array}{ll} \max & 1x \quad \text{tel que} \\ & \sum_{P\in\mathcal{P}, e\in P} x_P \leq w(e) \qquad (\forall e \in E(G)) \\ & x \in \mathbb{N}^{\mathcal{P}} \end{array} \tag{2}$$

Cette formulation correspond bien exactement au problème initial.

Notons que dans ces deux formulations, entière ou relachée, le nombre de variables (d'inconnus) est égal au nombre de (s,t)-chemins, qui peut être très grand comme nous l'avons déjà signalé. Il n'est pas raisonnable d'espérer trouver une solution, si la taille de celle-ci est déjà trop grande par rapport aux données du problème (ici, le graphe et sa fonction de capacité essentiellement). Nous donnons donc une deuxième formulation. Plutôt que de chercher le nombre de chaque type de chemin, nous demandons combien de chemins faire passer par chaque arc. Les contraintes doivent nous permettre ensuite de montrer qu'il existe un flot tel qu'effectivement chaque arc est bien dans ce nombre exact de chemins du flot. Nous avons donc une variable par arc, et nous aurons besoin d'un théorème pour prouver qu'il s'agit bien d'une formulation du problème de flot maximum. Pour des raisons de commodité, nous ajoutons un arc supplémentaire ts à $E(G)$, que nous appelerons h, avec une capacité infinie, et nous l'appelons *arc de demande*. Plutôt que de considérer les (s,t)-chemins du graphe, nous allons considérer les cycles passant par l'arc h. Un flot devient un ensemble de cycles respectant les capacités (une *circulation*), et nous cherchons à maximiser le nombre de cycles passant par h

Nous notons $\delta^+(X)$ pour tout $X \subseteq V$, l'ensemble des arcs $uv \in E(G)$ avec $u \in X$, $v \notin X$ et $\delta^-(X)$ l'ensemble des arcs $uv \in E(G)$ avec $u \notin X$, $v \in X$. Par abus de notation, nous noterons $\delta^+(u) = \delta^+(\{u\})$, et $\delta^-(v) = \delta^-(\{v\})$. $\delta^+(X)$ est l'ensemble des arcs *sortants* de X, $\delta^-(X)$ est l'ensemble des arcs *entrants* de X. $\delta(X) = \delta^+(X) \cup \delta^-(X)$ est la *coupe* de G engendrée par le *bord* X. Si $X \subset V(G)$ contient u mais pas v, on dit que $\delta(X)$ est une

(u,v)-coupe. Si $\delta^-(X) = \emptyset$, $\delta^+(X)$ est une *coupe orientée*. Nous définissons $\delta(X,Y) = \{uv \in E(G) \ : \ u \in X, v \in Y\}$ et $d(X,Y) = |\delta(X,Y)|$. Le voisinage $N(U)$ d'un ensemble de sommets est $\{v \notin U \ : \ \exists u \in U / uv \in E(G)\}$, $N(u) = N(\{u\})$. Une deuxième formulation est alors :

$$\begin{array}{lll} \max & x_h \quad \text{tel que} & \\ & \sum_{e \in \delta^+(u)} x_e = \sum_{e \in \delta^-(u)} x_e & (\forall u \in V(G)) \\ & c_e \geq x_e \geq 0 & (\forall e \in E(G)) \end{array} \tag{3}$$

Le premier type de contraintes porte le nom de *loi de conservation* ou *loi de Kirchhoff*, par référence aux lois dans les réseaux électriques, qui sont une application des résultats de théorie des flots, voir par exemple [50]. Ces contraintes expriment que le nombre de cycles qui entrent dans chaque sommet est égal au nombre de cycles qui en sortent.

Nous avons besoin d'un résultat élémentaire, sur l'existence d'un (s,t)-chemin. Il s'agit aussi d'un modèle du genre de théorème apprécié en optimisation combinatoire : il existe un certain objet ssi un autre objet n'existe pas.

Théorème 0.2.1. *Soient G un graphe, s et t deux sommets distincts de G. Il existe un (s,t)-chemin si et seulement s'il n'existe pas de (t,s)-coupe orientée.* □

La remarque suivante est fondamentale dans l'étude des flots : soit C un cycle de G. Alors pour tout $U \subset V(G)$, $|\delta^+(U) \cap C| = |\delta^-(U) \cap C|$. En remplaçant U par les singletons, nous obtenons que les lois de Kirchhoff sont vérifiées par les cycles, et donc plus généralement par les flots. Réciproquement, les points entiers du polyèdre défini par les contraintes de (3) correspondent à des solutions du problème de flot maximum.

Proposition 0.2.2. *Soit x une solution entière au programme* (3)*, il existe un (s,t)-flot $\mathcal{P}$ dans G tel que $x(e) = |\{P \ : \ e \in \mathcal{P}\}|$.* □

La preuve de ce théorème repose sur le Théorème 0.2.1 : étant donné un flot entier $|\mathcal{P}|$ dans (G,x), nous appliquons ce théorème sur le graphe G_f, obtenu en supprimant de G les arêtes e telles que $f(e) = 0$, avec $f(e) = x(e) - |\{P \in \mathcal{P} \ : \ e \in P\}|$ si $e \in E(G)$. Nous trouvons alors soit un chemin simple à rajouter à $\mathcal{P}$, soit une (s,t)-coupe orientée $f(\delta^-(X)) = 0$, mais puisque f vérifie aussi les lois de conservation de flot, $f(h) = 0$ donc $|\mathcal{P}| = x(h)$.

Nous venons de mettre en évidence qu'il existe plusieurs formulations pour un même problème, avec des polyèdres complètement différents. Dans

la formulation (3), le nombre de contraintes dépend linéairement du nombre d'arêtes du graphe, tout comme le nombre de variables. Ceci permet de décider si un vecteur de $\mathbb{R}^E$ est réalisable, c'est-à-dire s'il vérifie chacune des contraintes du programme linéaire (relaxé) (3), ou bien de trouver une contrainte violée, ce qui s'appelle *séparer* le vecteur du polyèdre. La méthode des ellipsoïdes, mise au point par Grötschel, Lovász et Schrijver [22], permet de trouver une solution réelle optimale dès lors que nous possédons un algorithme de séparation polynomial. Dans le cas des flots, nous pouvons donc trouver ainsi des solutions optimales au programme relaxé, mais malheureusement ces solutions ne sont pas forcément entières : il faudrait idéalement pouvoir montrer que les sommets du polyèdre du programme (3) sont entiers pour en déduire un algorithme de résolution du problème de flot maximum.

0.3 Dualité et théorème de Menger

Dans le Théorème 0.2.1, nous avons montré une propriété intéressante : soit il existe un (s,t)-chemin, soit il existe une (s,t)-coupe U avec $\delta^+(U) = \emptyset$, et seule l'une des deux possibilités est vraie. Si nous cherchons un (s,t)-chemin et qu'il n'en existe pas, nous pouvons donner une garantie de non-existence, par une (s,t)-coupe Y avec $\delta^+(Y) = \emptyset$. Il est toujours plus facile de montrer l'existence d'un objet, que de montrer son inexistence. Pour prouver l'existence, il suffit de montrer cet objet. Ainsi, tout résultat permettant de prouver la non-existence d'objets revêt une importance particulière. Voyons un exemple un peu plus subtil, généralisant le précédent pour bien en expliquer l'intérêt, et qui constitue l'un des premiers résultats de l'optimisation combinatoire. Il donne une condition d'existence d'un flot de taille donnée.

Théorème 0.3.1 (Menger, 1927)**.** *Soit G un graphe, s et t deux sommets de G. Alors le nombre maximum de (s,t)-chemins arc-disjoints est égal au cardinal minimum d'une (s,t)-coupe sortante.*

Reformulons différemment : il existe k (s,t)-chemins arc-disjoints ssi il n'existe pas de (s,t)-coupe C avec $\delta^+(C) < k$. Un théorème de la forme : l'existence d'un objet est équivalente à la non-existence d'un autre objet, est appelé *bonne caractérisation* de l'objet en question. En théorie de la complexité, le problème de déterminer l'existence d'un tel objet est dans la classe $\mathrm{NP} \cap \mathrm{coNP}$, et les deux points de vue sont en fait équivalents. En plus d'être une condition nécessaire à l'existence d'un algorithme efficace pour résoudre

le problème, une bonne caractérisation permet souvent une analyse fine de la structure mathématique du problème étudié. Revenons à notre problème de flot, et plus généralement à la programmation linéaire en nombres entiers, puisque nous avons donné une formulation de notre problème.
Malheureusement, il semblerait (sauf si P = NP) qu'il n'existe pas de bonne caractérisation pour la programmation linéaire en nombres entiers, mais nous en avons une pour la programmation linéaire.

Lemme 0.3.2 (Farkas [12]). *Soit un polyèdre $P = \{x \in \mathbb{R}^n \ : \ Ax \leq b\}$, défini par des contraintes linéaires. Alors P est vide ssi il existe un vecteur $y \in \mathbb{R}^m_+$ tel que $yA = 0$ et $yb < 0$.*

Pour les problèmes d'optimisation, nous pouvons l'appliquer comme ceci : il existe une solution de valeur k ssi le polyèdre $P = \{x \in \mathbb{R}^n \ : \ Ax \leq b, cx \geq k\}$ est non-vide. Réécrivons les contraintes de (1), pour exprimer l'existence d'un flot de taille k :

$$\sum_{e \in P} x_P \leq c_e \quad (\forall e \in E(G)) \tag{4}$$

$$-x_e \leq 0 \quad (\forall e \in E(G)) \tag{5}$$

$$-1 \cdot x \leq -k \tag{6}$$

De cette écriture, par le Lemme 0.3.2, la non-existence de solution équivaut donc à l'existence de vecteurs $y \in \mathbb{R}^E_+$, $z \in \mathbb{R}^{\mathcal{P}_{st}}_+$ et $w \in \mathbb{R}$ tel que :

$$\begin{cases} \sum_{e \in P} y_e \geq w \quad (\forall P \in \mathcal{P}_{st}) \\ \sum_{e \in E} c_e y_e < kw \end{cases} \tag{7}$$

La première contrainte de (7) exprime que w est une borne inférieure sur la longueur du plus court (s,t)-chemin. La somme des longueurs d'une solution est donc supérieure ou égale à kw, mais la seconde contrainte donne justement une borne inférieure de kw sur la somme des longueurs de chaque arc, multipliées par les capacités des arcs. Nous avons :

$$kw \leq \sum_{e \in E} x_e c_e \leq \sum_{e \in E} y_e c_e < kw \tag{8}$$

pour tout flot x de k chemins, contradiction. Il est donc assez naturel que l'existence de y et w empêche la réalisation d'un flot de taille k, mais la réciproque n'est pas intuitive.
Nous n'avons pas encore donné de formulation pour le problème du multi-flot entier. Par analogie avec ce que nous avons fait pour le flot maximum,

nous pouvons par exemple prendre des variables sur les (s_i, t_i)-chemins, et exprimer une contrainte par arc du graphe. Notons pour toute fonction réelle positive z sur les arcs d'un graphe, $\lambda_z(u, v) = \min_{P \in \mathcal{P}_{uv}} z(P)$ la longueur d'un plus court (u, v)-chemin. En appliquant de la même manière le Lemme 0.3.2 nous obtiendrions le résultat suivant, popularisé sous le nom de Théorème Japonais :

Théorème 0.3.3 (Onaga, Kakusho [48]). *Soit $(G, c, (s_i, t_i)_{i \in I}, (r_i)_{i \in I})$ une instance du problème de multiflot entier n'admettant pas de solution. Alors, il existe $z : E(G) \to \mathbb{R}^+$ tel que :*

$$\sum_{e \in E} c_e z_e < \sum_{i \in I} r_i \lambda_z(s_i, t_i) \tag{9}$$

La réciproque est évidente : si nous avons un vecteur de longueur $z : E(G) \to \mathbb{R}_+$, toute solution est composée de (s_i, t_i)-chemins chacun plus long que la distance $\lambda_z(s_i, t_i)$, donc la somme des longueurs est bornée par le membre droit de (9). Le membre gauche donne une borne supérieure pour tout ensemble de chemins du graphe respectant les capacités.

Revenons au Lemme 0.3.2 : $\{x \ : \ Ax \leq b \wedge cx \geq k\}$ est vide ssi il existe un vecteur positif y tel que $yA = c$ et $yb < k$. Cette formulation nous donne envie d'essayer de trouver un vecteur y minimisant yb, car cela nous donnerait la meilleure borne supérieure sur notre maximum que permet d'obtenir cette méthode. Nous définissons donc le *programme dual* de $\max\{cx \ : \ Ax \leq b\}$ par :

$$\min\{yb \ : \ y \in \mathbb{R}^+ \wedge yA = c\} \tag{10}$$

En fait, cela mène à un résultat très fort, le théorème de dualité de la programmation linéaire :

Théorème 0.3.4 (von Neumann [62], Gale, Kuhn et Tucker [18]). *Pour tout A, c et b :*

$$\max\{cx \ : \ Ax \leq b\} = \min\{yb \ : \ y \in \mathbb{R}^+ \wedge yA = c\} \tag{11}$$

pourvu qu'au moins l'un des deux optimums soit fini.

De plus, soient x tel que $Ax \leq b$ et $y \geq 0$ tel que $yA = c$, alors x et y sont des optimaux ssi $y(Ax - b) = 0$, c'est le principe des *écarts complémentaires*.

Reprenons nos deux formulations et donnons leurs formulations duales. Pour la formulations avec variables sur les chemins, nous obtenons :

$$\begin{aligned} \min \quad & y \cdot c \quad \text{s.t.} \\ & \sum_{e \in P} y_e \geq 1 \quad (\forall P \in \mathcal{P}_{st}) \\ & y_e \geq 0 \quad (\forall e \in E(G)) \end{aligned} \tag{12}$$

C'est délicat de donner une interprétation d'une solution fractionnaire de ce programme, mais remarquons que pour tout point entier du polyèdre défini par les contraintes de ce programme, $\{e \in E(G) \ : \ y_e \geq 1\}$ intersecte tous les (s,t)-chemins, donc contient les arcs sortants d'une (s,t)-coupe. Donc, les points entiers minimaux correspondent à des (s,t)-coupes. Si nous arrivons à montrer que les optimaux de nos deux programmes (primal et dual) sont entiers, nous obtiendrons par le théorème de dualité une version pondérée du théorème de Menger.
La formulation avec variables sur les arcs donne :

$$\begin{aligned} \min \quad & y \cdot c \quad \text{s.t.} \\ & y_e \geq 0 \quad (\forall e \in E(G) \setminus \{h\}) \\ & y_{uv} = z_u - z_v \quad (\forall e = uv \in E(G) \setminus \{h\}) \\ & x_s = 1 \wedge x_t = 0 \end{aligned} \tag{13}$$

Bien sûr, ici aussi l'interprétation des points entiers ramène à des (s,t)-coupes.

Une direction possible pour chercher une solution a un problème de programmation linéaire est de tenter de trouver un vecteur d'amélioration : étant donné une solution réalisable x (mais pas nécessairement optimale), exprimer un autre programme linéaire décrivant l'existence d'un vecteur $\vec{u}$, tel qu'il existe un $\varepsilon > 0$, avec $x + \varepsilon\vec{u}$ réalisable, et cu positif. Cette technique est connue sous le nom de *méthode primale-duale*, et il est intéressant de noter qu'elle a historiquement été introduite comme une généralisation d'un algorithme de résolution du *problème de transbordement*, qui est un cas particulier du problème de flot. Elle s'applique d'ailleurs aux flots, donnant l'*algorithme de Ford-Fulkerson*.

0.4 Unimodularité et max-flot-min-coupe

Nous avons remarqué que si les solutions optimales d'un programme linéaire et de son dual sont des vecteurs entiers, nous avons une bonne caractéri-

sation des objets combinatoires correspondants par le théorème de dualité. De ce fait, nous aimerions avoir des conditions suffisantes utiles permettant de décider si un polyèdre est *entier*, c'est-à-dire par définition si toutes ses faces contiennent un point entier. Une première condition suffisante peut être dérivée de la formule de Cramer, et nous permet de montrer facilement que le polytope des flots est entier.
Une matrice entière $A \in \mathbb{N}^{m \times n}$ est dite *unimodulaire* si son déterminant est ± 1, et *totalement unimodulaire* si toute sous-matrice carrée de A a déterminant -1, 0 ou 1.

Théorème 0.4.1. *Soit $P = \{x \ : \ Ax \leq b\}$ un polyèdre, avec A totalement unimodulaire et b entier. Alors P est entier.*

Preuve. Nous esquissons une preuve pour le cas où le polyèdre est de plein rang. Les sommets d'un polyèdre de dimension n sont définis par l'intersection de n hyperplans indépendants supports du polyèdre, c'est-à-dire il existe une sous-matrice A' de A, de rang n, composée de n lignes de A telle que $A'x = b$ et $\det A' \neq 0$. Donc $x = A'^{-1}b$, or par la formule de Cramer, les coefficients de A'^{-1} sont entiers puisque $\det A' \in \{-1, 1\}$, donc x est entier. □

Soit M la matrice d'incidence sommets-arcs de G, *i.e.* définie par :

$$m_{w,uv} = \begin{cases} 1 & \text{si } v = w, \\ -1 & \text{si } u = w, \\ 0 & \text{sinon.} \end{cases}$$

Soit $P = \{x \ : \ Mx = 0, x \geq 0, x \leq c\}$ le *polytope des circulations* de G, où c est le vecteur de capacité. Nous reconnaissons dans $Mx = 0$ les lois de Kirchhoff : en effet, la contrainte associée au sommet v est $(\chi_{\delta^-(v)} + \chi_{\delta^+(v)})x = 0$. M est totalement unimodulaire : une sous-matrice correspond à un sous-graphe de G, il suffit donc de montrer que M a déterminant dans $\{-1, 0, 1\}$. Or, si G possède un cycle (pas forcément orienté), le déterminant est nul puisque les colonnes correspondantes ont une somme nulle avec les coeeficients suivants : 1 si l'arc apparait dans le sens du cycle, -1 sinon. Et si G est acyclique, il suffit de développer le déterminant selon les sommets de degré 1. Nous avons même que :

$$\begin{pmatrix} M \\ I \\ -I \end{pmatrix} \text{ est totalement unimodulaire.}$$

Donc si c est entier, toute face contient un point entier, ce qui nous confirme l'intégralité du flot maximum, et de son programme dual. Nous en déduisons le théorème suivant de bonne caractérisation :

Théorème 0.4.2 (Max-Flot-Min-Cut). *La valeur d'un (s,t)-flot maximum de G est égale au minimum de la somme de capacités d'une (s,t)-coupe sortante.*

En fait, puisque les sommets sont caractérisés par les contraintes *serrées* (atteintes avec égalité) du programme linéaire, il suffit de montrer qu'un ensemble générateur de ces contraintes serrées est unimodulaire pour prouver l'intégralité du sommet. Ceci nous donne envie de généraliser la notion d'unimodularité totale. Un système $Ax \leq b$ est *fortement localement unimodulaire* (LSU) en un sommet x_0 si la sous-matrice de A définie par les contraintes serrées en x_0 admet une sous-matrice $n \times n$ de déterminant ± 1. Nous avons donc que pour une fonction de poids entière, si l'optimum est atteint en un sommet et que le système $Ax \leq b$ est LSU en ce sommet, ce sommet est entier, par les précédents arguments.

Nous disons qu'un système $Ax \leq b$ est *total-dual entier* (TDI) si A et b sont rationnels, et que pour tout vecteur d'entiers c, le programme :

$$\min\{yb \ : \ y \geq 0, yA = c\}$$

est atteint par une solution y entière, si le minimum est fini. L'intérêt réside dans le théorème suivant :

Théorème 0.4.3 (Edmonds et Giles [10]). *Si $Ax \leq b$ est TDI et b entier, alors $\{x \ : \ Ax \leq b\}$ est entier.*

Soit $Ax \leq b$ un système d'inéquation, supposons que pour tout $c \in \mathbb{Z}^m$ tel que le programme dual $\min\{yb \ : \ yA = c, y \geq 0\}$ possède une solution, il possède une solution telle qu'une sous-matrice $n \times n$ de la sous-matrice des lignes de A correspondant aux coefficients non-nuls de y soit totalement unimodulaire. Alors ce système $Ax \leq b$ est TDI. Ceci fournit un schéma de preuve courant pour montrer l'intégralité d'un polyèdre : il suffit de montrer que pour toute forme linéaire objectif, il existe une solution du dual sous une forme bien choisie, telle que nous pouvons trouver une sous-matrice totalement unimodulaire. Ainsi, nous définissons les systèmes *fortement localement totalement unimodulaire* (LSTU) en un sommet si la sous-matrice des contraintes serrées est totalement unimodulaire. En résumé :

Théorème 0.4.4 ([10], [20],[25] entre autres, voir [53]). *Soit $\mathcal{S} := Ax \leq b$ un système d'inéquations linéaires, et soient les propriétés suivantes :*

(i) A est totalement unimodulaire,
(ii) $\mathcal{S}$ est LSTU en chaque sommet,
(iii) $\mathcal{S}$ est TDI,
(iv) $\mathcal{S}$ est LSU en chaque sommet,
(v) $P = \{x \ : \ Ax \leq b\}$ est entier.

Alors, pour un système de plein rang, (i) $\Longrightarrow$ (ii), (ii) $\Longrightarrow$ (iii), (iii) $\Longrightarrow$ (iv) et (iv) $\Longrightarrow$ (v). □

Notons qu'aucune des réciproques éventuelles n'est valide. Lorsque le système n'est pas de plein rang, des implications analogues existent.

Il existe de nombreuses applications de ces résultats polyédraux en théorie des flots. Nous mentionnons plus loin le théorème de Lucchesi et Younger [36] qui se prouve en montrant qu'un polyèdre bien choisi est LSTU :

Théorème 0.4.5 (Lucchesi et Younger). *La cardinalité d'un transversal minimum des coupes orientées d'un graphe est égal au nombre maximum de coupes orientées disjointes.*

$$\begin{aligned} &\min\{|E| \ : \ \forall U \subset V(G), \delta^-(U) = \emptyset \Rightarrow \delta^+(U) \cap E \neq \emptyset\} \\ =&\max\{|\mathcal{C}| \ : \ \mathcal{C} \text{ ensemble de coupes orientées disjointes}\} \end{aligned} \tag{14}$$

Preuve. Nous commençons par écrire un programme linéaire dont les solutions entières correspondent à des couvertures de coupes :

$$\begin{aligned} \min \quad & c \cdot x \quad \text{s.t.} \\ & x_a \geq 0 \\ & \sum_{a \in \delta^+(U)} x_a \geq 1 \quad (\forall U \subset V, \delta^-(U) = \emptyset) \end{aligned} \tag{15}$$

Les vecteurs entiers du système de ce programme sont des vecteurs caractéristiques d'ensembles d'arcs dès lors que le minimum est fini (puisque dans ce cas les coefficients des vecteurs sont clairement tous au plus 1), le second type de contraintes exprime la non-orthogonalité de ces vecteurs avec les vecteurs caractéristiques des coupes orientées, il s'agit donc bien de couver-

tures de coupes orientées. Écrivons le programme dual :

$$\begin{aligned} \max \quad & 1 \cdot y \quad \text{s.t.} \\ & y_U \geq 0 \quad (\forall U \subset V, \delta^+(U) = \emptyset) \\ \sum_{U, e \in \delta^+(U), \delta^-(U) = \emptyset} & y_U \leq c_e \quad (\forall e \in E) \end{aligned} \tag{16}$$

Ici, les vecteurs entiers correspondent à des vecteurs caractéristiques de multi-ensembles de coupes orientées, avec pour contrainte que chaque arc e peut être dans au plus c_e de ces coupes. Donc, en prenant $c = 1$, il s'agit de coupes disjointes, et donc un minimum entier correspond dans ce cas à une couverture de coupes orientées minimum, et un maximum correspond à un ensemble de coupes orientées disjointes. Il suffit donc de montrer que le système sous-jacent au programme (15) est LSTU.
Soit donc c un vecteur de naturels, et y un optimum pour le programme (16). Notons $\mathcal{F}$ l'ensemble des bords des coupes orientées $U \subseteq V$ avec $y_U \neq 0$ (cette famille définit les contraintes serrées par y). Nous montrons que $\mathcal{F}$ peut être choisie comme *famille laminaire* : deux ensembles U_1 et U_2 sont *croisés* si $U_1 \cap U_2, U_1 \setminus U_2$ et $U_2 \setminus U_1$ sont non-vides. Une *famille laminaire* est une famille d'ensembles deux-à-deux non-croisés. Pour montrer l'existence de $\mathcal{F}$, nous utilisons un argument de décroisement : soient U_1 et U_2 deux coupes orientées croisées avec $0 < y_{U_1} \leq y_{U_2}$. Alors nous définissons une solution y' elle aussi optimale, égale à y en tous ses coefficients, sauf :

$$\begin{cases} y'_{U_1 \cup U_2} &= y_{U_1 \cup U_2} + y_{U_1} \\ y'_{U_1} &= 0 \\ y'_{U_2} &= y_{U_2} - y_{U_1} \\ y'_{U_1 \cap U_2} &= y_{U_1 \cap U_2} + y_{U_1} \end{cases} \tag{17}$$

La somme des y n'est pas modifiée, donc cette solution, si elle est valide, est optimale. Il suffit maintenant de montrer que y' est réalisable : pour chaque arc $e \in E$, selon l'appartenance de e à $\delta^+(U_1)$, $\delta^+(U_2)$, $\delta^+(U_1 \cup U_2)$ et $\delta^+(U_1 \cap U_2)$ (il y a 5 cas car $\delta(U_1 \setminus U_2, U_2 \setminus U_1)$ est vide) :

$$\sum_{U, e \in \delta^+(U), \delta^-(U) = \emptyset} y'_U = \sum_{U, e \in \delta^+(U), \delta^-(U) = \emptyset} y_U \leq c_e$$

Nous devons montrer que le processus de décroisement termine. Pour cela, nous pourrions prouver que la somme $\sum_{U_1, U_2} y_{U_1} y_{U_2}$ prise sur les ensembles croisés décroit strictement à chaque étape. Nous avons donc une solution optimale correspondant à une famille laminaire. Or, la matrice d'incidence d'une famille laminaire est totalement unimodulaire (voir par exemple [30]). Le système est LSTU, et donc TDI. □

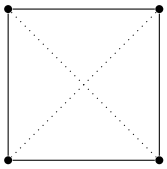


FIGURE 1 – *Un exemple pour lequel la condition de coupe n'est pas suffisante pour l'existence d'une solution entière. Les arêtes de demande sont en pointillé.*

0.5 Mineurs de graphes et condition de coupe

La plupart des problèmes qui nous intéressent sont NP-complets, donc intrinsèquement difficiles à résoudre en toute généralité. Comme nous le verrons pour les problèmes de multiflots entiers, en restreignant les instances, par exemple à des classes de graphes particulières, nous pouvons obtenir des cas de polynomialité. En pratique, les problèmes industriels que nous pouvons être amenés à résoudre possèdent des structures propres, ce qui rend légitime cette approche de chercher à caractériser les instances pour lesquelles une bonne caractérisation est valide.

Pour rester dans le cas des flots, nous sommes tentés de déterminer pour quels graphes une caractérisation *à la Menger* est correcte. Plus précisément, plaçons-nous dans les graphes non-orientés. Nous dirons qu'une instance (G, H, c, r) de multiflot entier vérifie la *condition de coupe* si pour toute coupe $\delta(U)$ de $G + H$, $\sum_{e \in E(G) \cap \delta(U)} c(e) \geq \sum_{e \in E(H) \cap \delta(U)} r(e)$. La condition de coupe est un candidat pour une bonne caractérisation puisque si elle n'est pas vérifiée par une coupe $\delta(U)$, les demandes $E(H) \cap \delta(U)$ ne peuvent toutes être satisfaites simultanément. Malheureusement, cette condition n'est pas suffisante pour l'existence d'un multiflot entier, comme le montre l'exemple de la Figure 1.

Quels sont les graphes pour lesquels la condition de coupe est suffisante ? Bien sûr le choix d'étudier la condition de coupe est arbitraire, et nous pourrions tenter de faire un travail équivalent à partir de n'importe quelle condition nécessaire. Ces questions admettent généralement deux types de réponses. Nous pouvons caractériser ces graphes, soit en donnant une méthode de construction, en déterminant quelques graphes simples vérifiant la propriété et des opérations permettant de les combiner en préservant la propriété, soit en donnant des obstructions, c'est-à-dire une liste la plus simple

possible de graphes qui *apparaissent* exactement dans les graphes ne vérifiant pas la propriété. La notion d'*apparition* est relative à la propriété étudié, et définit une *opération de mineur*.

Pour la condition de coupe, nous définissons la *suppression d'un sommet* comme l'opération qui à un graphe $G = (V, E)$ et un sommet $x \in V$, associe le graphe $(V \setminus \{x\}, \{uv \in E \ : \ u \neq x \neq v\})$, et la *suppression d'une arête* comme l'opération qui à un graphe $G = (V, E)$ et une arête $e \in E$ associe le graphe $(V, E \setminus \{e\})$. La *contraction d'une arête* associe à G et $e = uv$ le graphe obtenu en supprimant l'arête e puis en identifiant u et v.
Un graphe H est :

- *sous-graphe* d'un graphe G si H s'obtient depuis G par une succession de suppressions de sommets ou d'arêtes,
- *sous-graphe induit*, ou *induit* d'un graphe G si H s'obtient depuis G par une succession de suppressions de sommet,
- *mineur* (ou *mineur d'arête*) d'un graphe G si H s'obtient depuis G par une succession de suppressions et de contractions d'arête.

Le paragraphe suivant est donné pour illustrer notre propos, mais les définitions et résultats introduits ne seront pas utilisés par la suite. Un *graphe signé* $G = (V, E, \Sigma)$ est un graphe dont un ensemble $\Sigma \subseteq E$ d'arêtes est distingué. Les graphes signés sont un formalisme très utiles dans l'étude des coupes des graphes, et permet aussi de coder des instances de multiflots : les arêtes signées correspondent alors aux arêtes de demande. Les graphes signés sont considérés comme quotientés par la relation d'équivalence : $(V, E, \sigma) \equiv (V, E, \Sigma')$ si $\Sigma \Delta \Sigma'$ est une coupe de (V, E). Dans un graphe signé, la *suppression d'un sommet* et la *suppression d'une arête* sont définies comme dans un graphe. La *contraction d'une arête* appliquée à l'arête $e = uv$ d'un graphe signé (V, E, Σ) est (V', E', Σ') avec V' et E' obtenu par suppression de e et identification de u et v, et :

- si $e \in \Sigma$, $\Sigma' = \Sigma \Delta \delta(u)$,
- si $e \notin \Sigma$, $\Sigma' = \Sigma$.

Un graphe signé H est *mineur* du graphe signé G si H s'obtient depuis G par une succession de suppressions de sommet ou d'arête et de contractions d'arête. Nous appelons *impair-K_n* le graphe signé $(V(K_n), E(K_n), E(K_n))$. Le théorème suivant de Geelen et Guenin [19] caractérise les graphes pour lesquels la condition de coupe est suffisante (la condition d'Euler est $c(\delta(v))$ est pair pour tout $v \in V$) :

Théorème 0.5.1 (Geelen, Guenin). *Soit $G = (V, E, \Sigma)$ un graphe signé*

sans mineur impair-K_5. Alors pour toute fonction de capacité $c : E \to \mathbb{R}^+$, la condition de coupe implique l'existence d'un multiflot rationnel. Si en plus c est entier, il existe un multiflot demi-entier, et si c vérifie la condition d'Euler, il existe un multiflot entier.

Ce théorème généralise un résultat de Seymour que nous mentionnerons à nouveau par la suite.

Théorème 0.5.2 (Seymour)**.** *Un graphe (V, E) n'a pas de mineur K_5 ssi pour tout $\Sigma \subset E$ et toute fonction $c : E \to \mathbb{R}^+$, la condition de coupe implique l'existence d'un multiflot. Dans ce cas, si c est entier, la condition de coupe implique l'existence d'un multiflot demi-entier, et si en plus c vérifie la condition d'Euler, l'existence d'un multiflot entier.*

Deuxième partie

Multiflots entiers

Introduction.

S'il est possible de faire remonter l'histoire des flots au XIX^{e}, Schrijver [57] fait débuter celle des multiflots seulement à l'entre-deux-guerres, à un article de Kantorovich [27] de 1939. Le contexte était le routage de plusieurs trains dans un même réseau ferré. Rapidement, de nombreuses applications sont apparues, notamment dans la télécommunication (problèmes de routages) et dans la microélectronique (problèmes de placement). Récemment encore, de nouveaux problèmes se posent sur des multiflots contraints, en particulier liés au développement de l'Internet.
Les problèmes de multiflots peuvent être résolus par des approches de programmation linéaire en général, tant que nous ne sommes intéressés que par des solutions éventuellement fractionnaires. Malheureusement, dès l'essor de la théorie de la complexité dans les années 1970, la restriction à la recherche des solutions entières a été démontrée difficile (par Knuth, Lynch, Karp par exemple, nous y reviendrons). Pour contourner cela, une approche est de s'intéresser à la topologie du graphe, ce qui au vu des applications pratiques (nous pensons au dessin de circuits électroniques) semble assez naturel : graphes planaires, grilles, graphes série-parallèles viennent prendre leur place dans ce schéma.
La multiplicité des problèmes nous a amené à tenter une classification d'une sous-partie des problèmes de multiflots entiers, sous forme d'un tableau présenté dans le chapitre suivant. Celui-ci met en avant plusieurs problèmes ouverts, auquels nous nous sommes attaqués, parfois avec succès, en particulier pour les flots à deux commodités dans le plan qui font l'objet du Chapitre 4. Nous espérons que notre tableau permettra de populariser les problèmes encore ouverts.

Chapitre 1

Tableau des problèmes

1.1 Présentation du tableau

Dans cette section, nous présentons un outil de travail, publié dans [46], permettant de classifier les différents problèmes de multiflots entiers qui nous ont intéressés. La multiplicité de paramètres du problème, pouvant être combinés entre eux, permet de le décliner en de nombreux cas, tant et si bien que, parmi la centaine de possibilités, il est parfois difficile de distinguer les résultats les uns des autres, ainsi que de repérer les problèmes encore ouverts.

Nous avons donc choisi de construire un tableau, avec pas moins de 6 dimensions, pour répertorier le plus soigneusement possible tous les théorèmes et leurs conséquences. L'ensemble tient sur une seule page, permettant d'identifier d'un coup d'œil la complexité du problème de multiflot avec les paramètres désirés. Bien sûr, le choix de ces paramètres supplémentaires peut paraître partiellement arbitraire. Nous avons choisi ceux qui nous semblaient intéressants dans le cadre de notre étude. D'autre part, nous nous sommes restreints à traiter du seul cas de l'existence de multiflots entiers, laissant à d'autres la possibilité de traiter de la même manière la question de la maximisation de multiflots, par exemple [2].

Au final, le tableau contient pas moins de 189 problèmes, dont 21 sont encore ouverts. Une quinzaine de théorèmes permettent, par le jeu de réductions bien connues, de déterminer la complexité des 168 problèmes résolus. Ces théorèmes seront rappelés par la suite. Depuis l'élaboration de la première version de ce tableau, 14 cases ont pu être complétées, grâce aux résultats de Schwärzler [58] et à ceux présentés dans ce document. En ayant

permis de mettre en avant les problèmes encore ouverts, nous espérons que prochainement la complexité de chacun des problèmes du tableau sera connue.

			orienté			orienté acyclique			non-orienté		
			arc-disjoint		sommet-	arc-disjoint		sommet-	arête-disjoint		sommet-
G	$\|E(H)\|$	r	gen	eulérien	disjoint	gen	eulérien	disjoint	gen	eulérian	disjoint
gen	arb	bin	NPC	NPC	NPC	NPC	NPC	NPC	NPC	NPC	**NPC** [4]
		un	NPC	NPC		NPC	NPC		NPC	NPC	
	fix	bin	NPC	NPC	NPC	NPC	NPC	**P** [2]	NPC	NPC	**P** [12]
		un	NPC	NPC		NPC	**NPC** [19]		NPC	NPC	
		fix	NPC	**???** [3]		P	P		P	P	
	2	bin	NPC	**P** [10]	**NPC** [2]	NPC	P	P	NPC	**P** [13]	P
		un	NPC	P		**NPC** [1]	P		**NPC** [1]	P	
		fix	NPC	P		P	P		P	P	
		2	**NPC** [2]	P		P	P		P	P	
plan	arb	bin	NPC	NPC	NPC	NPC	NPC	NPC	NPC	NPC	**NPC** [7]
		un	NPC	NPC		**NPC** [19]	**NPC** [21]		**NPC** [5]	NPC	
	fix	bin	NPC	???	**P** [14]	NPC	???	P	NPC	???	**P** [15]
		un	NPC	???		**NPC** [16]	**P** [22]		**NPC** [16]	???	
		fix	???	???		P	P		P	P	
	2	bin	NPC	P	P	NPC	P	P	NPC	P	P
		un	**NPC** [9]	P		**NPC** [11]	P		**NPC** [11]	P	
		fix	???	P		P	P		P	P	
		2	???	P		P	P		P	P	
$G+H$ plan	arb	bin	NPC	???	NPC	**P** [6]	P	**NPC** [20]	NPC	**P** [18]	**NPC** [8]
		un	NPC	???		P	P		**NPC** [8]	P	
	fix	bin	NPC	???	P	P	P	P	**P** [17]	P	P
		un	NPC	???		P	P		P	P	
		fix	???	???		P	P		P	P	
	2	bin	NPC	P	P	P	P	P	P	P	P
		un	**NPC** [11]	P		P	P		P	P	
		fix	???	P		P	P		P	P	
		2	???	P		P	P		P	P	

TABLE 1.1 – *Complexité de problèmes de multiflots entiers.*

1.2 Définitions, notations, paramètres du tableau

Pour toute la durée de notre étude, les problèmes de multiflots entiers seront encodés à l'aide de deux graphes sur un même ensemble de sommets. L'un est orienté si l'autre l'est. Le premier, que nous noterons usuellement G et qui sera appelé *graphe d'offre*, correspond au graphe dans lequel le routage s'effectue. Le deuxième, le *graphe de demande* H, avec $V(H) \subseteq V(G)$, définira les arcs ou arêtes de demande du multiflot, autrement dit, chacun de ses arcs définit une paire origine-destination. Chacun de ces deux graphes peut-être muni d'une fonction de poids sur ses arcs ou arêtes, $c : E(G) \to \mathbb{N}$ (pour *capacité*) définit la capacité de l'offre, et $r : E(H) \to \mathbb{N}$ (pour *requête*) définit la demande, c'est-à-dire la quantité de flot à router pour chaque paire origine-destination. Lorsque r et c ne sont pas précisées, nous admettrons qu'elles sont définies comme valant 1 sur tout élément de leur domaine. Nous parlons dans ce cas de *problème de chemins arêtes-disjoints (ou arc-disjoints)*.
Par convention, H ne possède pas de sommet isolé. Les sommets de H sont

1. Even, Itai et Shamir [11], 1976
2. Fortune, Hopcroft et Wyllie, [14], 1980. De plus, G est acyclique et $|E(H)|$ borné implique la polynomialité.
3. Polynomial pour 3 arêtes de demande (Ibaraki, Poljak [26], 1991)
4. Karp [28], 1975
5. Kramer et van Leeuwen, [31], 1984
6. Lucchesi-Younger, [36], 1978
7. Lynch[37], 1975
8. Middendorf et Pfeiffer, [41], 1993, même en restreignant le degré maximum à 3
9. Müller, [42], 2005
10. Frank, [16], 1989, voir aussi la preuve de Nash-Williams du théorème de Hu [44], 1969
11. Naves, [45], 2008
12. Robertson, Seymour, [51], 1990
13. Rothschild-Whinston [52], 1966. La polynomialité et la suffisance de la condition de coupe s'étendent au cas où H est l'union de deux étoiles, K_4 ou C_5 (Lomonosov [34], 1985)
14. Schrijver, [56], 1992
15. Schrijver [55], 1990, polynomial si les terminaux sont sur le bord d'un nombre borné de faces.
16. Schwärzler, [58], 2007, voir le Théorème 3.0.11
17. Sebő [59], 1993
18. Seymour [61], 1981. Polynomial dans les graphes $G + H$ sans mineur K_5.
19. Vygen [63], 1995
20. Voir le Théorème 2.1.1
21. Indépendamment, Pfeiffer [49], 1990, et Marx [40], 2004
22. Voir le Théorème 2.3.8

alors les extrémités des arêtes (ou arcs) de demande, et sont appelés *terminaux*.
Le problème de trouver un multiflot entier consiste, étant donné G et H, à trouver un multi-ensemble de cycles $\mathcal{C}$ du graphe $G + H$, répondant aux conditions suivantes :

(i) pour tout $C \in \mathcal{C}$, $|C \cap E(H)| = 1$,

(ii) pour tout $e \in E(G)$, au plus $c(e)$ cycles de $\mathcal{C}$ contiennent e,

(iii) pour tout $d \in E(H)$, exactement $r(d)$ cycles de $\mathcal{C}$ contiennent d

Par la suite (et avec des conséquences sur la complexité du problème), les multi-ensembles seront codés par des ensembles munis d'une fonction de poids à valeur entière. Notons que retirer cette contrainte d'intégralité revient à définir le problème du multiflot fractionnaire.

Cette définition du problème ne prend pas en compte le cas des chemins sommet-disjoints. Nous pourrions de la même manière donner des capacités aux sommets, et imposer que chaque sommet appartienne à un nombre limité de cycles. Dans la littérature, de nombreux résultats considérent que chaque sommet a une telle capacité de 1 (donc c'est véritablement le problème de chemins sommet-disjoints). Nous notons cependant une certaine dissymétrie dans cette définition. En effet, *la demande est déterminée par des arêtes, là où l'offre correspond aux sommets*. Pour y remédier, il suffit de subdiviser chaque arête de demande en deux, créant un nouveau sommet de degré 2, qui portera la demande (cette transformation ne préserve pas la planarité de G, si G est planaire mais pas $G+H$). Dans le tableau, nous avons seulement donné le cas des chemins sommet-disjoints à titre indicatif. *Déterminer les complexités pour le problème plus général que nous venons d'esquisser pourrait constituer un nouveau sujet de recherche, car à notre connaissance, ce formalisme n'a pas été considéré jusqu'à présent.*

Pour rentrer plus précisément dans les paramètres du tableau, nous avons gardé ceux qui nous paraissaient les plus naturels à imposer sur ces problèmes de multiflots entiers. Une première distinction porte sur la nature des graphes G et H. Ils peuvent être orientés ou non, et lorsque G est orienté, il peut être acyclique, ce qui définit trois cas, représentant les trois grandes subdivisions verticales. Pour chacun des cas, nous distinguons les problèmes sommet-disjoints et arc-disjoints (ou arêtes disjoints), ainsi que le cas arc-disjoints (ou arêtes-disjoints) dans le cas particulier où $G + H$ avec leurs poids r et c forment un *graphe Eulérien*. Par Eulérien, nous entendons qu'ils vérifient la *condition d'Euler* :

– dans le cas orienté, pour chaque sommet la somme des capacités et requêtes entrantes est égale à la somme des capacités et requêtes sortantes (loi de Kirchhoff), plus exactement, pour tout sommet v :

$$\sum_{e \in \delta_G^-(v)} c(e) + \sum_{d \in \delta_H^-(v)} r(d) = \sum_{e \in \delta_G^+(v)} c(e) + \sum_{d \in \delta_H^+(v)} r(d) \qquad (1.1)$$

– dans le cas non-orienté, la somme des capacités et requêtes de chaque sommet est paire, pour tout sommet v :

$$\sum_{e \in \delta_G(v)} c(e) + \sum_{d \in \delta_H(v)} r(d) \text{ est pair} \qquad (1.2)$$

Ces conditions sont particulièrement naturelles lorsque l'on considère qu'une solution du problème de multiflot entier n'est rien d'autre qu'un packing de cycles, donc définit un graphe Eulérien. Un coup d'œil rapide au tableau permet de confirmer que cette condition simplifie bien souvent le problème, permettant l'existence d'algorithmes polynomiaux dans de nombreux cas.

Une autre distinction est faite dans le tableau selon la planarité de l'instance. Nous nous intéressons au cas où l'offre est constituée d'un graphe planaire, correspondant en particulier aux applications pratiques notamment dans le domaine de l'intégration à large échelle (VLSI), et celui encore plus spécialisé où l'union $G+H$ est elle-même planaire (évidemment, dans ce cas G est planaire aussi). Cette dernière distinction permet d'ajouter au tableau des résultats très important, comme les résultats sur les graphes signés de Seymour [60], ou le théorème de Lucchesi-Younger [36], qui trouve ici une belle application dans les graphes planaires. Lorsque seul G est planaire, les outils topologiques permettent à Schrijver [55], [56] d'offrir deux résultats dans le cas sommet-disjoints, utilisant l'impossibilité d'avoir deux chemins qui se croisent.

Enfin, des arguments purement comptables viennent compléter le tableau. Nous prenons alors en compte le nombre d'arcs (ou arêtes) de demande $|E(H)|$, en distinguant les trois possibilités suivantes : nombre non borné d'arcs, nombre borné d'arcs, et seulement 2 arcs. Il serait certainement possible de retenir plus de cas : la complexité est-elle la même pour 3 arcs et pour un nombre fixe d'arcs ? Cette question n'a pas de réponse générale, mais l'expérience montre qu'en général c'est bien le cas.

Le dernier paramètre utilisé, probablement le plus technique, permet de distinguer la forme des fonctions de poids, avec principalement deux conséquences. Premièrement, cela permet de distinguer le cas arc-disjoint (ou arête-disjoint), du cas plus général des multiflots entiers. La transformation consistant à remplacer chaque arc par un nombre d'arcs parallèles égal au poids de l'original permet d'obtenir un problème de chemins arc-disjoints depuis tout problème de multiflot entier. Malheureusement, les deux formulations ne sont pas équivalentes, puisque la représentation en mémoire des arcs parallèles, utilisée par le problème de chemins arc-disjoints, est exponentiellement plus gourmande que celle par capacité (ceci correspond à la notion usuelle de pseudo-polynomialité *versus* NP-complétude forte dans la littérature). Ainsi, *bin* (pour binaire) désigne un encodage des poids par des entiers, avec un code de longueur logarithmique en la valeur des poids, alors que *un* (pour unaire) représente les problèmes codés par arcs parallèles (donc en un code de longueur polynomiale en la valeur des poids). Si en pratique il ne semble pas y avoir de différence, il est important de noter que les problèmes ne sont pas exactement les mêmes dans les deux formalismes. Deuxièmement, nous traitons aussi les cas avec une requête totale bornée (*fix*), c'est-à-dire $\sum_{d \in E(H)} r(d)$ doit être inférieur à une constante donnée, et le cas très particulier de rechercher seulement deux chemins disjoints.

D'autres paramètres auraient pu retenir notre attention. Dans le cas d'un graphe G planaire, il eut été intéressant de traiter à part les cas où les terminaux sont sur les bords d'un petit nombre de faces. On considère les graphes pouvant se plonger dans le plan, qu'en est-il des graphes sur des surfaces plus complexes ? Malheureusement, le souci de clarté et de concision impose le choix d'un nombre réduit de paramètres, et ceux retenus nous ont semblé être les plus pertinents. Pour autant, des cas spéciaux peuvent aussi bien être étudiés par ailleurs, et nous ne nous en priverons pas. Pour ces problèmes, la comparaison avec ceux du tableau permet toujours de donner une mesure, certes grossière, de l'intérêt d'un résultat.

1.3 Réductions folkloriques

Afin d'exploiter au maximum chaque théorème pour remplir le tableau, nous utilisons plusieurs réductions bien connues, certaines étant classiques au point de ne pas avoir de paternité à revendiquer.

La plus simple est celle permettant de réduire les problèmes non-orientés aux problèmes orientés. Pour cela, chaque arête est remplacée par le gadget introduit dans la Figure 1.1. Bien que cette réduction préserve la planarité de G et $G + H$, ainsi que le nombre de demande, le graphe obtenu n'est ni Eulérien, ni acyclique.

FIGURE 1.1 – *Réduction du cas non-orienté au cas orienté. Dans le gadget de gauche, il y a au plus un chemin entre les deux sommets originaux, de gauche à droite ou de droite à gauche.*

Passer d'un problème de chemins arête-disjoints à un problème sommet-disjoints se fait naturellement en prenant le *graphe adjoint* (*line graph*). Malheureusement, *la planarité de G n'est pas conservée, ni l'Euléricité*, ce qui limite la portée de cette réduction. Dans le cas orienté, prendre le graphe adjoint orienté (remplaçant les sommets et leurs arcs incidents par des graphes complets bipartis) s'applique aussi.

Dans le cas des graphes non-orientés, le degré maximal de G peut toujours être ramené à 4 (mais cela n'a pas de sens avec des capacités codées en binaire). Pour cela, on utilise la réduction de la Figure 1.2. Cette réduction conserve la planarité et l'Euléricité de $G+H$. Si $G+H$ est planaire, le degré maximum de $G + H$ peut même être réduit à seulement 3 en suivant une remarque de Middendorf et Pfeiffer [41], sans conserver l'Euléricité. Pour cela, il faut noter que si $G+H$ est planaire, les chemins formant les solutions peuvent être supposés *non-croisés*, et chaque sommet de degré 4 peut être remplacé par un cycle de longueur 4, comme en Figure 1.3. Dans ce cas, le problème de trouver des chemins arête-disjoints revient à trouver des chemins sommet-disjoints.

Une autre réduction remarquée par Middendorf et Pfeiffer dans le même article permet d'ajouter aux problèmes de chemins sommet-disjoints avec $G+H$ planaire, la restriction que le degré maximum est 3, à la condition d'ajouter des arêtes de demande supplémentaires. La Figure 1.4 introduit cette

FIGURE 1.2 – *En ajoutant quelques sommets bien placés, selon le schéma ci-dessus, on peut diminuer le degré de tout sommet à moins de* 4*. Remarquons que tous les routages entre les arêtes inférieures sont possibles dans le graphe de droite.*

FIGURE 1.3 – *Si $G+H$ est planaire, dans le cas arête-disjoints, on peut remplacer les sommets de degré* 4 *par des sommets de degré* 3*, car s'il existe un solution, il existe une solution pour laquelle les chemins ne se croisent pas.*

transformation (qu'il convient d'appliquer plusieurs fois, mais tout cela reste bien polynomial). L'intérêt principal est de ramener le problème sommet-disjoints au problème arête-disjoint, puisque lorsque les degrés sont au plus 3 (sans capacité), les deux problèmes sont identiques (sommet-disjoints est alors équivalent à arête-disjoints).

Pour clore cette partie sur les réductions, rappellons la correspondance entre les chemins arc-disjoints dans un digraphe acyclique Eulérien, et les chemins arête-disjoints dans un graphe non-orienté Eulérien, mise en évidence par Vygen [63].

Lemme 1.3.1 (Vygen, 1995)**.** *Soit (G, H) une instance du problème de multiflot entier orienté telle que $G + H$ est Eulérien et G est acyclique. Soit (G', H') l'instance du problème de multiflot entier non-orienté obtenue à partir de (G, H) en négligeant l'orientation. Alors, il existe une correspondance un-à-un entre les solutions de (G, H) et celles de (G', H').*

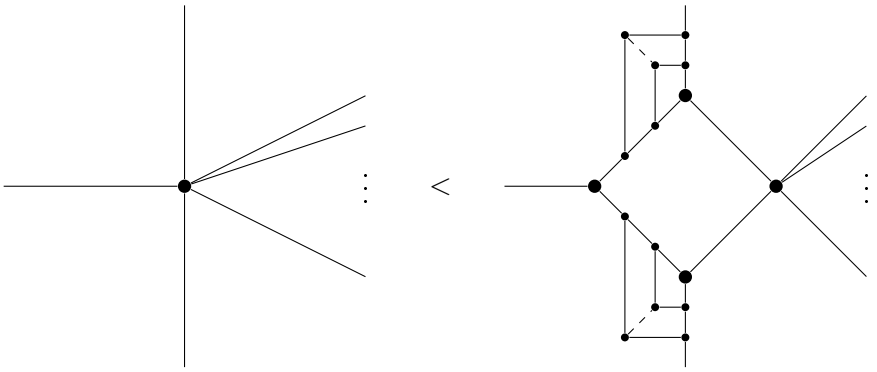


FIGURE 1.4 – *Si $G+H$ est planaire, dans le cas sommet-disjoints, on peut réduire les sommets de degré au moins 4, jusqu'à l'obtention de sommets de degré 3, au prix de l'ajout de nouvelles demandes (ici en tirets). Il est aisé de vérifier qu'un seul chemin avec des terminaux extérieurs au gadget peut y passer.*

Chapitre 2

Routage dans les graphes orientés acycliques

Nous présentons ici des résultats originaux sur les digraphes acycliques planaires. L'acyclicité apporte plusieurs résultats de polynomialité et de bonnes caractérisation. Le Théorème 0.4.5 de Lucchesi et Younger [36] trouve dans les graphes planaires une application particulièrement intéressante. Rappelons leur théorème :

Théorème 2.0.2 (Lucchesi et Younger, 1978)**.** *La cardinalité minimum d'un transversal des coupes orientées d'un digraphe est égale au nombre maximum de coupes orientées arc-disjointes.*

La dualité des graphes planaires permet de traduire coupe orientée par cycle orienté dans le graphe dual. Le théorème se réécrit alors dans le cas planaire de la manière suivante :

Corollaire 2.0.3. *La cardinalité minimum d'un transversal des cycles orientés d'un digraphe planaire est égale au nombre maximum de cycles orientés arcs-disjoints.*

Si $G + H$ est planaire et G acyclique, H est un bloqueur des cycles orientés. S'il existe un packing de cycles de taille $|H|$, le problème est résolu puisque chaque cycle intersectera H une et une seule fois. Sinon, il existe un bloqueur $B \subset E(G)$ de cardinalité strictement inférieure à H, constituant une bonne caractérisation.

Un autre résultat de polynomialité intéressant sur les digraphes acycliques est donné par Fortune, Hopcroft et Wyllie [14], sur le problème de

chemins sommet-disjoints. Leur solution exploite un algorithme dynamique basé sur le fait que le nombre de chemins est fixé.

2.1 Si $G + H$ est planaire

Lorsque $G + H$ est planaire, ces deux précédents résultats laissent en suspens la question de la complexité du cas sommet-disjoints avec un nombre arbitraire de chemins. Middendorf et Pfeiffer [41] donnent une réponse dans le cas où $G+H$ n'est pas acyclique. Cependant, leur preuve se dérive facilement vers le cas acyclique, ce que nous avons montré dans [46]. Nous reprenons ici cette preuve, qui est une réduction depuis Planar 3-Sat, qui est 3-Sat restreint aux instances dont le graphe d'appartenance des variables aux clauses est planaire [33].

Théorème 2.1.1. *Le problème des chemins sommet-disjoints est NP-complet dans les digraphes acycliques, même sous la restriction $G + H$ planaire.*

Preuve. Nous réduisons Planar 3-Sat à notre problème. Soit φ une formule dont le graphe biparti (C, V, F) associé (les sommets sont les variables V et les clauses C, les arêtes F correspondent à la présence d'une variable dans une clause) est planaire. Sans perte de généralité, nous supposons que chaque variable apparait au plus trois fois, dont exactement une fois négativement, et qu'aucune clause ne contient deux fois la même variable.
Nous subdivisons chaque arête xc, avec c une clause et x une variable apparaissant dans cette clause, en deux en ajoutant un nouveau sommet, nommé v_{cx}. Prenons un ordre total quelconque sur les variables, et remplaçons chaque sommet c de C par un gadget, construit de la manière suivante. Soit z l'une des variables apparaissant dans la clause, x et y les deux autres avec $x < y$ pour l'ordre choisi. Nous enlèvons c, et ajoutons le gadget sur la gauche de la Figure 2.1 liant les sommets v_{xc}, v_{yc} et v_{zc}. Notons que cette opération préserve la planarité. De plus, il existe un unique chemin entre deux des précédents sommets, de v_{xc} vers v_{yc} (ceci nous permettra de montrer l'acyclicité).
Nous remplaçons maintenant de manière similaire chaque sommet x de variable : notons a et b les clauses dans lesquelles x apparaît positivement, c celle où il apparaît négativement. x est alors remplacé par le gadget du côté droit de la Figure 2.1. Si une variable n'apparaît que deux fois, le sommet v_{xb} de la Figure correspond à un nouveau sommet.

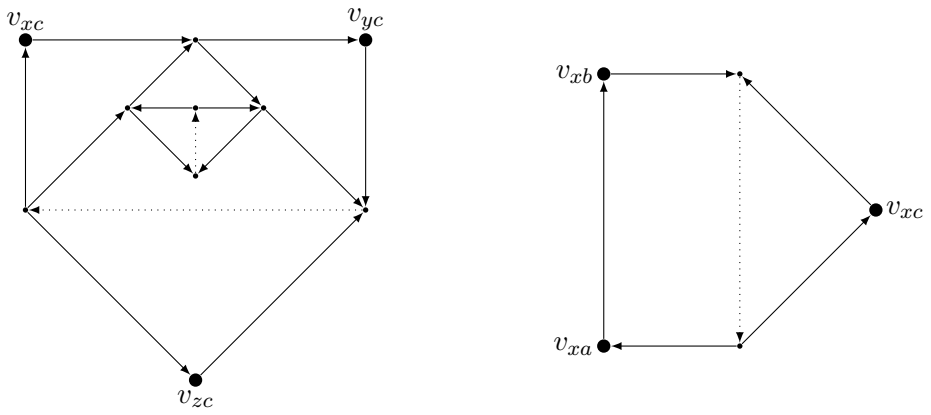


FIGURE 2.1 – *Le gadget pour les clauses est à gauche, celui pour les variables est à droite. Les sommets en gras sont ceux provenant de la subdivision des arêtes initiales. Les arcs en pointillés sont des arcs de demande.*

Soient G_φ, H_φ les graphes d'offre et de demande ainsi construits. Nous prouvons qu'il existe un ensemble de chemins sommet-disjoints dans (G_φ, H_φ) ssi φ est satisfiable.
Soit une solution aux problèmes de chemins disjoints. On peut supposer que la demande d'un gagdet de variable est satisfaite par un chemin de longueur 3 (passant pas v_{xa} et v_{xb}, correspondant à x faux) ou 2 (par v_{xc}, correspondant à x vrai), puisque les chemins peuvent être choisis minimaux par inclusion en terme de sommets. Les deux demandes d'un gadget de clause ne peuvent être simultanément satisfaites que si au moins un des trois sommets en gras dans la Figure 2.1 n'est pas utilisé par un chemin dont la demande est extérieure à ce gadget. Ceci s'interprète, étant donné notre choix d'assignation des variables, comme avoir une variable dont l'assignation valide la clause en question. Il reste à prouver que le graphe G_φ est acyclique. Pour cela, on remarque que chaque gadget est lui-même acyclique, donc s'il existe un cycle, celui-ci passe par plusieurs gadgets, alternativement de clauses et de variables. Comme il existe un seul chemin possible dans les gadgets de clauses entre deux sommets partagés par le reste du graphe (les sommets gras), ce cycle doit passer par des gadgets de variables strictement croissantes selon l'ordre qui a été préalablement choisi, ce qui contredit le fait qu'il s'agit d'un cycle. □

Selon les paramètres choisis pour le tableau, ce résultat termine l'étude du cas $G + H$ planaire, G orienté acyclique. Le dernier théorème implique

bien sûr la NP-complétude pour G planaire ou G quelconque.

2.2 Si G seulement est planaire

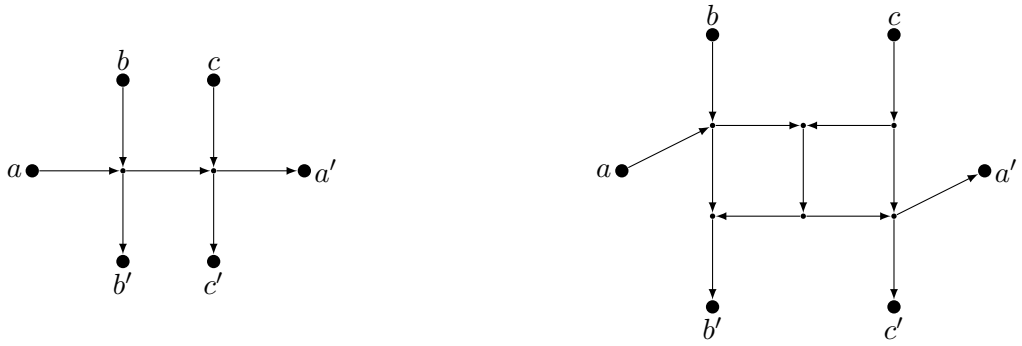
Lorsque G seulement est planaire, la difficulté du problème augmente très vite. Vygen [63] a ainsi montré que le problème du multiflot entier dans les digraphes acycliques est NP-complet même avec G planaire. Pfeiffer [49], et indépendemment Marx [40], ont montré la NP-difficulté du même problème, avec l'hypothèse supplémentaire que $G+H$ est en plus Eulérien. Même sans cette hypothèse, ces résultats utilisent un nombre non limité d'arcs de demande, et jusqu'à très récemment, la complexité du problème avec un nombre fixe d'arcs de demande restait indéterminée. Bien sûr, si la demande totale est bornée, les résultats de Robertson et Seymour [51] s'appliquent pour obtenir la polynomialité, mais cela n'est pas suffisant lorsque les capacités sont arbitraires. Une première réponse a été énoncée indirectement par Schwärzler en 2008, en répondant à une question ouverte autour du problème d'Okamura-Seymour. Sa preuve, après une modification mineure, permettait de démontrer la NP-difficulté du problème avec seulement 3 arcs de demande, repoussant l'inconnu au cas où seuls deux arcs de demande subsistent. Voici donc la solution pour deux arcs de demande, qui renforce aussi un résultat de Müller lorsque G n'est pas acyclique.

Théorème 2.2.1. *Le problème de multiflot entier avec G planaire orienté acyclique et H réduit à deux arcs de demande est fortement NP-complet.*

Preuve. Nous réduisons depuis Satisfaisabilité. Soit $C_1, \dots C_n$ une formule comprenant n clauses, sur l'ensembles des variables $\{X_1, \dots, X_p\}$. Soit G_1 une grille avec n colonnes et $2p$ lignes, où chaque intersection est un sous-graphe spécial, défini par la règle suivante (se reporter à la Figure 2.2) :

- $G_1(2i-1, j)$ est le graphe YES si X_i apparaît positivement dans C_j,
- $G_1(2i, j)$ est le graphe YES si X_i apparaît négativement dans C_j,
- $G_1(i, j)$ est le graphe NO dans tous les autres cas.

Les connections entre les sous-graphes sont créées en ajoutant des arcs entre les sommets comme suit : du sommet a' de $G_1(i, j)$ vers a de $G_1(i, j+1)$, des sommets b' et c' de $G_1(i, j)$ vers b et c dans $G_1(i+1, j)$ respectivement, pour toutes les valeurs cohérentes de i et j. Nous notons ligne(i) le sous-graphe induit par les sommets de $G_1(i, j), j \in [\![1, n]\!]$, et col$(j)$, le sous-graphe induit par les sommets de $G_1(i, j), i \in [\![1, 2p]\!]$. La $k^{\text{ième}}$ coupe verticale est définie par $\delta(\bigcup_{i \in [\![1,k]\!]} \text{ligne}(i))$, et la $k^{\text{ième}}$ coupe horizontale est définie par $\delta(\bigcup_{j \in [\![1,k]\!]} \text{col}(j))$.



Graphe NO Graphe YES

FIGURE 2.2 – *Deux gadgets pour la réduction. Le graphe de gauche ne permet pas de chemins entre c et b', celui de droite l'autorise mais seulement s'il n'y a pas d'autres chemins, en particulier de a vers a'.*

Affirmation 1. La formule booléenne est satisfiable ssi il existe un ensemble de chemins disjoints dans G_1 tels que :

(i) Pour tout $j \in [\![1, n]\!]$, il existe un chemin P_j dans $\mathcal{P}$ depuis $c \in G_1(1, j)$ vers $b' \in G_1(2p, j)$,

(ii) Pour tout $i \in [\![1, p]\!]$, il existe un chemin Q_i dans $\mathcal{P}$ soit depuis $a \in G_1(2i-1, 1)$ vers $a' \in G_1(2i-1, n)$, soit depuis $a \in G_1(2i, 1)$ vers $a' \in G_1(2i, n)$.

Preuve. Supposons que $\mathcal{P}$ existe, pour tout $i \in [\![1, p]\!]$, si Q_i a pour extrémité $a \in G_1(2i-1, 1)$ et $a' \in G_1(2i-1, n)$, alors assignons la valeur *faux* à X_i, sinon, assignons-lui *vrai*. Pour tout $j \in [\![1, n]\!]$, la $j^{\text{ième}}$ coupe horizontale est une coupe orientée, ainsi tout chemin Q_i, ($i \in [\![1, p]\!]$) ne peut intersecter qu'une seule ligne. Symétriquement tout chemin $P_j, j \in [\![1, n]\!]$ est contenu dans une seule colonne. Soit un chemin $P_j, j \in [\![1, n]\!]$, soit i l'indice de la première ligne telle que P_j passe par $b' \in G_1(i, j)$. Par construction du graphe, il faut pour cela que P_j soit le seul chemin qui passe par $G_1(i, j)$, et que $G_1(i, j)$ soit un graphe YES. Si i est pair, cela signifie que $X_{\frac{i}{2}}$ apparaît négativement dans C_j et que cette variable a valeur *faux*, donc la clause C_j est validée. Sinon $X_{\frac{i+1}{2}}$ apparaît positivement dans C_j et cette variable a valeur *vrai*, là encore C_j est validée. Puisque c'est vrai pour toute clause, la formule est satisfaite par l'assignement choisi. La réciproque se montre de la manière suivante : depuis une assignation satisfaisant la formule, on construit

les chemins horizontaux de manière cohérente avec la preuve ci-dessus. Les chemins verticaux passent par la droite de leur colonne jusqu'à arrivé à la première ligne de variable satisfaisant la clause, puis restent du côté droit. □

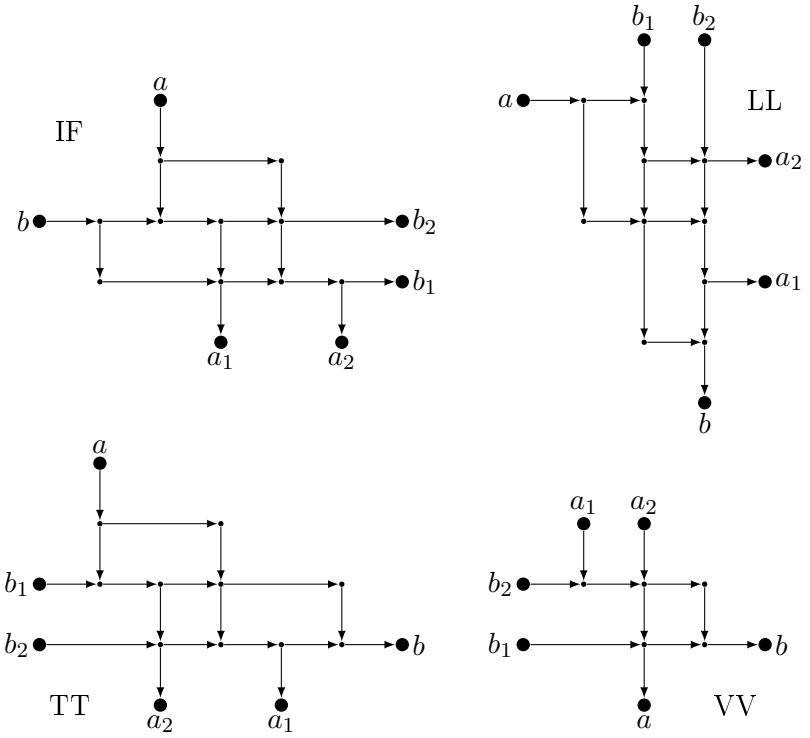


FIGURE 2.3 – *Quatre gadgets, de haut en bas et de droite à gauche : IF, LL, TT, VV. Les trois premiers, les routeurs, possèdent la propriété suivante : s'il y a deux chemins disjoints, l'un entre a et a_i, l'autre entre b et b_j, alors $i = j$. Le dernier, VV, assure qu'il n'y a pas un chemin entrant par a_1 et un autre entrant par b_2.*

Il nous suffit maintenant de forcer des chemins à se comporter comme le demande l'affirmation précédente. La condition (i) est faible,mais la condition (ii) demande l'utilisation de gadgets, décrits en Figure 2.3. Ces gadgets vont être ajoutés autour de G_1, comme le montre la Figure 2.5. Le principe de fonctionnement de ces ajouts est relativement simple : des chemins vont transportés depuis la droite et la gauche de G_1, l'information utile pour vérifier la condition (ii), jusqu'à un gagdet qui ne sera routable que si les infor-

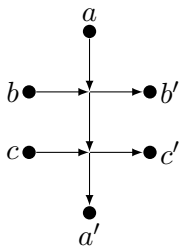


FIGURE 2.4 – *Le gadget ON, qui est symétrique de NO, et possède donc les mêmes propriétés.*

mations en question sont compatibles. Les gadgets de la Figure 2.3 agissent comme des routeurs, en recevant une information d'un côté, et en la retransmettant de l'autre. L'information est codée par un chemin, qui passe soit du côté droit de sa colonne, soit du côté gauche (haut et bas sur les lignes). Les gadgets ON et NO permettent de conserver cette "information".

Affirmation 2. Soit P_1 un chemin entre a et l'un parmi a_i, $i \in \{1, 2\}$, et soit P_2 un chemin entre b et l'un des b_j, $j \in \{1, 2\}$, dans IF, LL ou TT. Si P_1 et P_2 sont arc-disjoints, alors $i = j$. □

Affirmation 3. Il n'y a pas deux chemins arc-disjoints dans VV, tel que l'un va de b_2 à b et l'autre de a_1 à a. □

Les Affirmations 2 et 3 sont triviales. Nous aurons aussi besoin du graphe ON introduit par la Figure 2.4. Nous donnons maintenant la réduction complète, définissant un graphe de demande G. G est construit à partir de G_1, sous la forme décrite en Figure 2.5. G correspond à une grille de sous-graphes, avec $2p + n$ colonnes et $2p$ lignes. La sous-grille correspondant aux colonnes $p + 1$ à $p + n$ et aux lignes 1 à p est de fait G_1, sachant que deux lignes dans G_1 correspondent à une seule ligne de G. Les intersections $G(i, i)$, $G(p + 1 - i, n + p + i)$, $G(2p + 1 - i, i)$ et $G(p + i, n + p + i)$, pour tout $i \in [\![1, p]\!]$, sont les graphes IF, TT, LL et VV respectivement. Tous les autres sous-graphes non-définis sont des NO ou bien des ON, selon la figure. Tout comme pendant la construction de G_1, nous ajoutons des arcs entre les sommets de degré 1 dans les gadgets adjacents. Les lignes, colonnes et coupes verticales ou horizontales sont définies de la même manière que pour G_1.
Enfin, on ajoute quatre terminaux, un de chaque côté de la grille (se référer à la figure). La demande consiste en deux arcs, de s_1 à s_2 avec requête $2p$, de t_1 à t_2 avec requête $2p + n$.

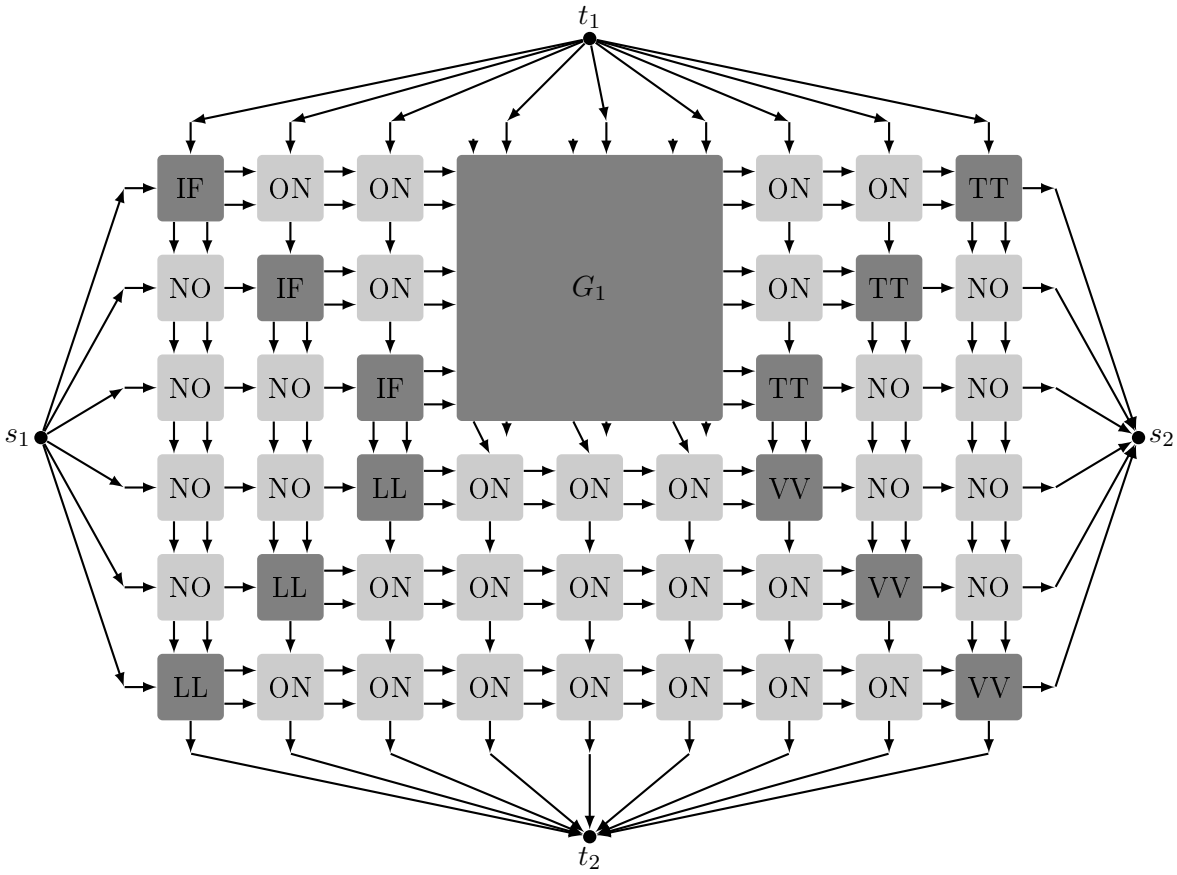


FIGURE 2.5 – *Schéma pour la réduction complète. Les coupes horizontales et verticales sont clairement acycliques. Ici, la formule contient* 3 *clauses sur* 3 *variables. Avec* p *variables et* n *clauses, il y aurait* $2p + n$ *colonnes et* $2p$ *lignes, et la demande serait* $2p + n$ *de* t_1 *vers* t_2*, et* $2p$ *de* s_1 *vers* s_2*.*

Affirmation 4. G est acyclique.

Il suffit d'observer que tous les arcs entre les gadgets sont orientés vers la droite ou vers le bas, et chaque gadget est acyclique.

Affirmation 5. Dans toute solution, pour chaque colonne de G, il existe exactement un chemin parcourant cette colonne de haut en bas, et qui ne traverse que des sommets de cette colonne. De même pour chaque ligne de G.

Preuve. En effet, les coupes horizontales et verticales sont orientés, et les coupes $\{s_1\}$, $\{s_2\}$, $\{t_1\}$, $\{t_2\}$ sont serrées. □

L'affirmation suivante est le point principal de la preuve, il s'agit de montrer qu'effectivement les chemins horizontaux vérifient la condition (ii).

Affirmation 6. Si un chemin horizontal quitte G_1 par l'arc inférieur de sa ligne (dans la $n + p^{\text{ième}}$ coupe verticale), alors, il rentre dans G_1 par l'arc inférieur (dans la $p^{\text{ième}}$ coupe verticale).

Preuve. Dans chaque sous-graphe ON ou NO, passent exactement un chemin de b à b' ou de c à c' et un chemin de a à a'. Ainsi, un chemin vertical quitte un sous-graphe IF par la droite ssi il entre dans le gadget LL de la même colonne par la droite. De même, sur la ligne entre chaque LL et VV, ou entre G_1 et IF ou bien encore entre G_1 et TT, le chemin horizontal reste au même niveau, et entre TT et VV, le chemin vertical reste sur le même côté.
Du coup, si un chemin quitte G_1 par l'arc inférieur de sa ligne (ligne $i \in [\![1, p]\!]$), alors,il entre dans le sous-graphe $G(i, 2p + n + 1 - i)$ par le sommet b_2, et le chemin vertical de la colonne $2p + n + 1 - i$ quitte donc ce gadget par le sommet a_2, par l'Affirmation 2. Donc ce même chemin rentre dans $G(2p+i, 2p+n+1-i)$ par le sommet a_1. Ainsi, par l'Affirmation 3, le chemin horizontal de la ligne $2p+1-i$ utilise les arcs du bas de sa ligne entre LL et VV. Par l'Affirmation 2, le chemin vertical de la colonne i utilisent les arcs de gauche entre IF et LL, et donc le chemin horizontal de la ligne i utilise les arcs inférieurs entre IF et G_1. □

Affirmation 7. Il existe une solution au problème de chemins arc-disjoints dans (G, H) ssi la formule est satisfiable.

Preuve. L'Affirmation 6 prouve que les chemins dans G_1 satisfont aux conditions de l'Affirmation 1 : s'il existe une solution au problème des chemins, la formule est satisfiable. Réciproquement, si la formule est satisfiable, la solution pour G_1 s'étend naturellement à G. □

Cette construction de Karp est bien polynomiale, le problème des chemins arc-disjoints étant dans NP, le théorème est prouvé. □

Nous pouvons terminer la réduction de façon légèrement différente, en identifiant les sommets t_1 et s_2 d'une part, et les sommets s_1 et t_2 d'autre part. Cela ne modifie pas le reste de la preuve, mais l'énoncé change :

Théorème 2.2.2. *Le problème de multiflot entier avec G digraphe, $G+H$ planaire et H restreint à deux arcs d'orientation opposée $\{st, ts\}$, est NP-complet.*

Une meilleure possibilité serait de ne faire passer qu'un chemin pour la demande t_1t_2. Pour cela, il suffit de remplacer les arcs incidents à t_1 et t_2 par un arc entre le bas d'une colonne et le haut de la colonne à sa gauche. Nous cherchons alors autant de chemins entre s_1 et s_2, plus un chemin entre le haut de la colonne de droite et la bas de la colonne de gauche, que nous pouvons identifier à s_2 et s_1 respectivement. Comme chacun des nouveaux arcs appartient à une coupe serrée, nous avons alors :

Théorème 2.2.3. *Le problème de multiflot entier avec G digraphe, $G+H$ planaire, $E(H) = \{st, ts\}$ et $r(st) = 1$ est NP-complet.*

Malheureusement, ces réductions n'utilisent pas de graphes Eulériens. Ainsi, sur les digraphes planaires acycliques, il reste à déterminer la complexité du problème de multiflot entier avec capacité dans le cas Eulérien lorsque le nombre d'arcs de demande est fixe. La prochaine section donne une réponse partielle à cette question.

2.3 Si G est planaire avec loi de conservation

Nous étudions maintenant le cas G planaire acyclique, $G+H$ Eulérien, dans le cas d'un nombre d'arcs de demande fixe. Nous allons décrire un algorithme pseudopolynomial pour ce problème (section 1.2). Nous supposerons pour des raisons techniques que les terminaux des chemins sont tous distincts, et tous des puits ou des sources. Ceci sans perte de généralité, puisque nous pouvons toujours augmenter G avec des arcs vers ou depuis des sommets de degré 1, et ainsi déporter les terminaux vers de nouveaux sommets.

Nous introduisons quelques outils pour traiter de la topologie des chemins dans le plan. Soient $e_1, \ldots, e_n$ une famille d'objets indicés par un intervalle $[\![1, n]\!]$, e_i et e_j sont dits *consécutifs* si $j = i+1$, ou bien si $i = n$ et $j = 1$. Pour trois éléments distincts a, b et c de cette famille, b est *entre* a et c s'il existe une suite d'éléments consécutifs distincts $a = u_1, u_2, \ldots u_k = c$ tel que b est l'un de ces éléments. Notons que si b est entre a et c, il n'est pas entre

c et a, et c est entre b et a. Ceci définit ce que l'on appelle usuellement un *ordre cyclique*. Un *intervalle* d'un ordre cyclique est un ensemble d'éléments entre deux éléments (non nécessairement distincts), ou bien l'ensemble de tous les éléments. Dans le premier cas, les deux éléments distingués sont les *extrémités* de l'intervalle. Il existe deux intervalles distincts ayant les mêmes extrémités lorsque celles-ci sont distinctes. Le complémentaire d'un intervalle est un intervalle.
L'ordre d'apparition des arcs autour d'un sommet v d'un graphe planaire induit un ordre cyclique ; y est entre x et z si x, y, z apparaissent dans cet ordre dans le *sens positif* (*trigonométrique* ou *sens inverse des aiguilles d'une montre*) autour du sommet v. Deux arcs sont alors consécutifs s'ils appartiennent à une même face. Nous pouvons aussi considérer l'ordre cyclique des arcs entrants ou des arcs sortants. Les chemins de mêmes extrémités seront considérés avec l'ordre cyclique de leur premier arc autour de la source de ces chemins, les indices adoptés par la suite respecteront l'ordre cycliques : P_i et P_{i+1} sont consécutifs, et si n est le plus grand indice, P_n et P_1 sont consécutifs.

Pour des raisons de commodité, et puisque nous allons montrer l'existence d'un algorithme pseudopolynomial, nous supposons que les arcs ont tous capacité 1, mais les arcs multiples sont permis. En particulier, les chemins sont arcs-disjoints, et nous considérons seulement les chemins simples. Soit P un chemin passant par le sommet v, nous notons P_v^- l'arc de P entrant dans v, et P_v^+ l'arc de P sortant de v, s'ils existent.
Soit P et Q deux chemins passant par un sommet v. Nous disons que :

- P et Q *se croisent* en v si les arcs de $\delta(v) \cap P$ n'appartiennent pas au même intervalle d'extrémités $\delta(v) \cap Q$.
- P *passe à droite de* Q en v si P et Q ne se croisent pas, et P_v^- apparaît avant P_v^+ dans le sens positif autour de v partant d'un arc de Q.
- P *passe à gauche de* Q en v sinon.

Remarquons que P passe à gauche (respectivement à droite) de Q ssi P passe à gauche (à droite) de Q^{-1} ssi P^{-1} passe à droite (à gauche) de Q, et que P et Q se croisent ssi Q et P se croisent. La connaissance pour P, Q, v consistant à savoir si P croise, passe à droite ou passe à gauche de Q en v est appelé *comportement relatif de P et Q en v*. Une des principales idées de l'algorithme que nous présentons, est que le comportement relatif de chaque paire de chemins en chaque sommet suffit pour déterminer les chemins.
Nous rappelons qu'un *ordre cohérent* (ou *topologique*) sur un graphe orienté est un ordre total sur les sommets de ce graphe, tel que si u est plus petit que v, il n'existe pas de chemins de v vers u. Un graphe admet un ordre

cohérent si et seulement si il est acyclique. Nous appelons *premier sommet commun* de P et Q le plus petit sommet de $V(P) \cap V(Q)$ (il ne dépend pas de l'ordre choisi). Par la suite, nous considérons qu'un ordre cohérent nous est donné (en trouver un est un problème facile).

Lemme 2.3.1. *Soit G un graphe planaire acyclique et $\mathcal{P}$ un ensemble de chemins arc-disjoints de G. Alors, il existe un ensemble $\mathcal{P}'$ de chemins arc-disjoints de même extrémités que $\mathcal{P}$ tel que :*

- *pour toute paire de chemins ayant une extrémité commune, ces deux chemins ne se croisent pas,*
- *pour toute paire de chemins se croisant, les deux chemins se croisent une seule fois sur leur premier sommet commun.*

Preuve. Soient P un (s,t)-chemin et Q un (s',t')-chemin distinct de $\mathcal{P}$, $u < v \in V(P) \cap V(Q)$, tels que P et Q se croisent en v, ou bien se croisent seulement en u et v est leur destination commune. Considérons la transformation qui remplace P et P par $P' = P_{su} \cup Q_{uv} \cup P_{vt}$ et $Q' = Q_{s'u} \cup P_{uv} \cup Q_{vt'}$. Nous cherchons donc à montrer l'existence d'un ensemble de chemins n'admettant pas de telle transformation. Considérons une solution minimisant le vecteur de croisements, en ordre lexicographique, c'est-à-dire le vecteur $(c_1, \ldots, c_n)$, avec c_i le nombre de paires de chemins se croisant dans le $i^{\text{ième}}$ plus grand sommet de G
Pour une transformation donnée, le nombre de croisements ne change pas dans les sommets autres que u et v, puisque les chemins n'y sont pas modifiés localement, et si u ou v est une extrémité de P et Q non plus.
Après transformation, le nombre de croisements entre P et Q en u et v diminue. De plus le nombre de croisements de tout autre chemin en u et v ne peut augmenter, puisque s'il croisait l'un de P et Q, il croise aussi l'un de P' et Q', et s'il ne croise ni P ni Q, il ne croise pas non plus P' et Q'. Les transformations diminuent donc le vecteur de croisements. □

Sans perte de généralité, *nous considérons G plongé sur une sphère.* Soit deux chemins P et Q de mêmes extrémités dans une solution décroisée. Puisque les chemins sont arc-disjoints, PQ^{-1} définit une courbe de la sphère, simple à une déformation continue arbitrairement petite près, séparant la sphère en deux parties connexes par le théorème de Jordan. Nous appelons *intérieur* de PQ^{-1} la partie connexe telle que PQ^{-1} parcourt son bord dans le sens positif, et *extérieur* l'autre partie connexe.
Si P et Q sont en plus consécutifs, aucun chemin de mêmes extrémités que P et Q ne passe par l'intérieur de PQ^{-1}, puisque tous ces autres chemins sont

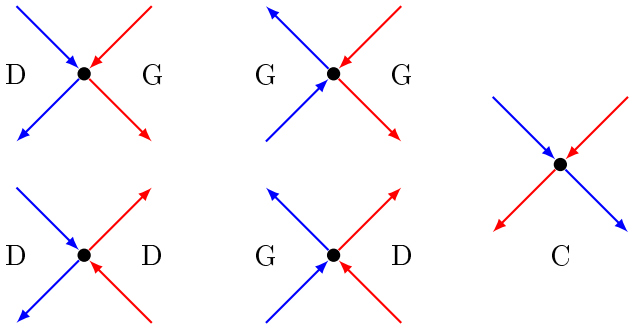


FIGURE 2.6 – *Les cinq possibilités de comportement de deux chemins arcs-disjoints en un sommet. G et D désigne si le chemin passe à gauche ou à droite du second chemin.*

à l'extérieur de PQ^{-1} à leur origine, et ne peuvent pas passer de l'extérieur à l'intérieur de PQ^{-1} sans croiser P ou Q.
Soit $P_1, \dots, P_k$ les $k = r(h)$ chemins d'une solution $\mathcal{P}$ décroisée pour l'arc de demande h. Nous pouvons donc partitionner la sphère (moins ces chemins) en k régions, les intérieurs de $P_i P_{i+1}^{-1}$ pour $i \in [\![1, k-1]\!]$ et l'intérieur de $P_k P^{-1}$, et que nous notons respectivement $\mathcal{K}_i^h$, $i \in [\![1, k]\!]$ (ceci induit donc un ordre cyclique isomorphe à celui des chemins considérés). Hormis les deux extrémités de l'arc de demande, l'intersection du bord de deux régions est vide si les deux régions ne sont pas consécutives, et est P_{i+1} si les deux régions sont $\mathcal{K}_i^h$ et $\mathcal{K}_{i+1}^h$.

Lemme 2.3.2. *Soient $t_1s_1, t_2s_2 \in E(H)$ deux arcs de demande distincts de capacité respective r_1 et r_2, et une solution décroisée dans le sens du Lemme 2.3.1. Alors il existe un intervalle $\mathcal{P}$ de (s_1, t_1)-chemins et un intervalle $\mathcal{Q}'$ de (s_2, t_2)-chemins tel que P croise Q ssi $P \in \mathcal{P}'$ et $Q \in \mathcal{Q}'$, ou bien $P \notin \mathcal{P}'$ et $Q \notin \mathcal{Q}'$.*

Preuve. Soient $P_1, \dots, P_{r_1}$ les chemins pour la demande $h = t_1s_1$. s_2 appartient à une région $\mathcal{K}_i^h$, et t_2 à $\mathcal{K}_j^h$, avec $i, j \in [\![1, r_1]\!]$. Notons que par hypothèse, les sommets s_1, s_2, t_1 et t_2 sont distincts. Puisque aucun (s_2, t_2)-chemin ne peut passer par s_1 (qui est une source) et t_1 (qui est un puits), tout (s_2, t_2)-chemins doit franchir les bords de régions consécutives de h, et ne peut pas franchir deux fois le même bord par décroisement, puisque un bord commun à deux régions est un (s_1, t_1)-chemin. Le graphe d'adjacence des régions de h étant un cycle, tout (s_2, t_2)-chemin doit donc suivre un des deux chemins entre la région contenant s_2 et celle contenant c_2, ce qui déter-

mine $\mathcal{P}$ et $\mathcal{Q}$, et $\mathcal{Q}$ est un intervalle. Par symétrie, $\mathcal{P}$ est aussi un intervalle. □

Lemme 2.3.3. *Soient P et Q deux chemins d'une solution décroisée, $u < v$ deux sommets distincts de $V(P) \cap V(Q)$. Alors P passe à gauche de Q en v ssi la destination t de P se trouve à l'intérieur de $P_{uv}Q_{uv}^{-1}$.*

Preuve. Notons que P et Q ne peuvent pas se croiser en v par décroisement. Puisque P ne peut pas croiser $\mathcal{C} = P_{uv}Q_{uv}^{-1}$ après v, P reste entièrement à l'intérieur ou entièrement à l'extérieur de $\mathcal{C}$ après v, notamment au sommet suivant v et à sa destination. Ceci détermine donc le comportement de P par rapport à Q en v. □

Ce lemme montre que le comportement relatif de deux chemins en un sommet qui n'est pas le premier sommet de croisement, est déterminé par les sous-chemins entre le précédent sommet de croisement et celui considéré. En fait, en utilisant l'acyclicité, les sommets de $\mathcal{C}$ sont inaccessibles depuis les sommets de $\{w \ : \ vw \in E(G)\}$ à l'intérieur (ou l'extérieur) de $\mathcal{C}$. Il suffit donc de connaitre les sommets à partir desquels chaque destination est accessible pour déterminer les comportements relatifs aux intersections autres que premières.

Lemme 2.3.4. *Soient C et D deux cycles d'intérieurs disjoints, et $v \in V(C)$ un sommet apparaissant exactement une fois dans D. Alors D passe à gauche de C en v.*

Preuve. Si D passe à droite de C en v, C passe par l'intérieur de D, contradiction. □

Lemme 2.3.5. *Soient P et Q deux chemins de mêmes extrémités, et $\mathcal{P} = \{P_1, \ldots, P_n\}$ un ensemble de (s,t)-chemins, avec s, t à l'extérieur de $C = PQ^{-1}$ et $C \subset \mathcal{K}_n^{st}$, tels que ces chemins ne se croisent pas. L'ensemble des chemins de $\mathcal{P}$ qui passent à droite (respectivement à gauche) de C en leurs premiers sommets communs est un intervalle.*

Preuve. Soient $i < j$ tel que P_j passe à gauche de C en sa première intersection v avec C. Soit u la première intersection de P_i et C (en admettant qu'elle existe).
Si $u = v$, puisque $C \subset \mathcal{K}_n^{st}$, P_i passe à gauche de C en v.
Sinon, soit w le plus petit sommet commun de $V(P_i) \cap V(P_j)$ strictement plus grand que u et v. Considérons le cycle D défini par $P_i'P_j'^{-1}$, avec P_i' et P_j' sous-chemins respectifs de P_i et P_j entre s et w. Puisque $C \subset \mathcal{K}_n^{st}$, l'intérieur de D ne contient pas C. De plus, D ne passe qu'une seule fois

par u, car sinon P_j passe par u et le cycle formée par P_i et P_j^{-1} entre s et u contiendrait C dans son intérieur. Donc par le Lemme 2.3.4, P_i passe à gauche de D en u.
Enfin, tous les chemins P_k avec $i < k < j$ doivent passer par u ou v, donc intersecter C. Ainsi, l'ensemble des sommets passant à gauche de C en leur première intersection est un intervalle. Symétriquement pour ceux passant à droite. □

Nous appelons *schéma de routage* la donnée pour toute paire de chemins du comportement relatif de ces deux chemins en leur premier sommet. Notons $R = \max_{h \in E(H)} r(h)$.

Lemme 2.3.6. *Pour toute instance (G, H, r, c), le nombre de schéma de routages possibles est au plus $O(((R+1)^4)^{|E(H)|^2})$, et ces routages peuvent être énumérés polynomialement.*

Preuve. Par le Lemme 2.3.2 et le Lemme 2.3.5, pour chaque couple d'arcs $(h_1, h_2) \in E(H)^2$, il suffit de déterminer quatre intervalles disjoints parmi les chemins pour h_1 pour déterminer le comportement de n'importe quel h_1-chemin lors de sa première intersection avec un h_2-chemin. Il y a trivialement moins de R^4 telles possibilités, ce qui donne en comptant chaque couple $O((R^4)^{|E(H)|^2})$. □

Notre algorithme consiste à tester tous les schémas de routage possibles. Pour chacun, nous routons consécutivement en chaque sommet, pris dans un ordre cohérent, les chemins passant par le sommet considéré. Nous connaissons alors le comportement relatif des chemins s'intersectant au sommet, soit parce qu'il s'agit d'une première intersection, soit par le Lemme 2.3.3 (le comportement pouvant alors être calculé en temps linéaire). Il nous faut montrer que cela suffit pour déterminer un unique routage par ce sommet, ou l'inexistence d'un routage sous ces contraintes. Pour cela nous aurons besoin de l'hypothèse d'Euléricité.

Lemme 2.3.7. *Soit v un sommet de G avec n arcs entrants et n arcs sortants, et n chemins passant par ce sommet. Alors, l'arc sortant utilisé par chaque chemin est uniquement déterminé par les arcs entrants utilisés par chaque chemin et le comportement relatif en v des chemins entre eux.*

Preuve. Soit P_1 un des chemins entrants dans v. Notons $P_2, \ldots, P_n$ les autres chemins entrants dans v, pris dans l'ordre de leurs arcs entrants dans v, notons a_i l'arc entrant du chemin i. Nous posons $a_{n+1} = a_1$. Considérons le comportement relatif de P_1 par rapport à $P_2, \ldots, P_n$. Soit i le plus grand indice tel que P_1 passe à gauche de P_i ($i = 1$ s'il n'en existe pas), et soit j

le plus petit indice tel que P_1 passe à droite de P_j ($j = n+1$ s'il n'en existe pas). P_1 doit sortir de v par un arc entre a_i et a_n (à gauche de P_i), et entre a_1 et a_j (à droite de P_j), donc $i < j$, et P_1 sort par un arc entre a_i et a_j.
Notons g le nombre de chemins desquels P_1 passe à gauche. Pour tout arc sortant a entre u_i et u_j, nous définissons $c(a)$ le nombre de chemins entrant entre a et u_1 qui croisent P_1, $p(a)$ le nombre d'arcs sortants entre u_1 et a. Alors si P_1 sort par l'arc a, le nombre de chemins sortants par un arc entre u_1 et a est $g + c(a) = p(a)$. Or, la quantité $g + c(a) - p(a)$ est strictement décroissante dans le sens positif (puisque $c(a)$ diminue et $p(a)$ augmente, tous les arcs entrants entre u_i et u_j devant croiser P_1), donc elle s'annule en au plus un arc, l'arc par lequel sort P_1, qui est donc uniquement déterminé par le comportement de P_1 vis-à-vis des autres chemins et des arcs entrants de chaque chemin.
Si cette quantité ne s'annule pas, elle est toujours strictement positive ou strictement négative. Dans le premier cas, pour le dernier arc sortant entre a_i et a_j, pour cet arc, $g + c(a) - p(a)$ est positif, contredisant l'existence de chemins suivant le comportement donné. Le deuxième cas est similaire. □

Théorème 2.3.8. *Le problème de chemins arc-disjoints avec G orienté planaire acyclique, $G + H$ eulérien et $|E(H)| = k$ est pseudopolynomial : il existe un algorithme de complexité $O(f(R)^{k^2}k^3n)$, où f est une fonction polynomiale, $n = |V(G)|$ et $R = \max_{h \in E(H)} r(h)$.*

Preuve. L'algorithme consiste à tester tous les schémas de routages possibles. Pour chacun, nous routons successivement les chemins en chaque sommet pris par ordre topologique croissant. Ainsi, lorsqu'un sommet v est traité, les chemins entrants dans ce sommet sont déjà connus. Puisque par le Lemme 2.3.3 et par le choix d'un schéma de routage, le comportement relatif des chemins est connu, il suffit d'appliquer le Lemme 2.3.7. L'algorithme induit par ce lemme est de complexité $O(|\delta^-(v)|^2)$ et donc $O(R^2k^2)$. En comptant les itérations sur chaque sommet, le nombre de routages possibles, le temps de calcul de l'ordre cohérent et le prétraitement ($O(nk)$ trivialement, le graphe étant planaire $|E| = O(n)$) pour calculer l'accessibilité aux destinations des demandes pour chaque sommet (voir la discussion suivant le Lemme 2.3.3), l'algorithme a une complexité de $O(((R+1)^4)^{k^2}R^3k^3n)$. Notons que pour un schéma de routage donné, les échecs possibles sont l'impossibilité de router un sommet selon le routage demandé, ou les chemins trouvés ne forment pas une solution (certains chemins n'ont pas d'arcs de demande correspondants). □

Nous ne savons pas s'il existe un algorithme polynomial pour ce problème. Pour rester proche de notre algorithme, il faudrait montrer que le nombre

de schémas de routage est beaucoup plus petit, ou bien faire une exploration intelligente des schémas pour n'en tester qu'un petit nombre (cela ne semble pas évident). Il faudrait aussi améliorer l'algorithme de routage en un sommet, pour pouvoir manipuler non pas des chemins, mais des faisceaux de chemins identiques, et montrer qu'il n'y en a pas trop (cette partie semble plus réaliste).
Notons que dans certains cas particuliers, un seul schéma de routage est possible, permettant un algorithme polynomial, avec de bonnes hypothèses sur G et H. C'est essentiellement ainsi qu'Ibaraki et Nagamochi [43] résolvèrent le cas particulier où les destinations des demandes sont toutes sur le bord de la face externe, et qu'il existe un ordre d'élimination des sommets qui permette de retirer successivement des sources de la face extérieure. Dans ce cas, il est facile de déterminer en chaque sommet s'il faut ou non croiser les chemins, selon l'ordre d'apparition des destinations sur la face externe.

Chapitre 3

Autour du théorème d'Okamura-Seymour

Comme le montre le tableau, les problèmes de multiflots entiers sont bien compris lorsque $G + H$ est planaire, notamment grâce aux travaux sur les graphes et matroïdes signés. De même, dans un cadre très général, les résultats de Lomonosov d'un côté, et de Robertson et Seymour d'un autre, ont permis de bien distinguer les cas polynomiaux lorsque G peut être quelconque. Notre tableau laissait pourtant entrevoir plusieurs lacunes dans le cas où G est planaire. Et malgré des avancées réalisées durant ce travail, il reste toujours des cases vides.
Plusieurs réponses partielles ont été apportées sur ces problèmes. Okamura et Seymour [47] ont ainsi prouvé le théorème suivant :

Théorème 3.0.9 (Okamura, Seymour, 1981)**.** *Soit G un graphe planaire, plongé dans le plan. Soit H un graphe de demande pour G, tel que tous les terminaux sont situés sur le bord de la face extérieure de G. Soit r et c des fonctions de capacité telles que (G, H, r, c) est Eulérien. Le problème de multiflot entier ainsi défini admet une solution ssi la condition de coupe est réalisée sur cette instance.*

Une preuve algorithmique originale est donnée dans [46]. Ce théorème, qui demande donc que toutes les demandes soient "sur une face" de G, a fait l'objet de plusieurs renforcements. Schrijver [54] a montré qu'on pouvait en fait demander de mettre les demandes sur 2 faces, puis dans le même sens Sebő [59] l'a montré pour 3 faces. Frank [15] a pris une autre direction, en montrant qu'on pouvait affaiblir la condition d'Euléricité :

Théorème 3.0.10. *Soit G un graphe planaire plongé dans le plan. Soit H un graphe de demande pour G, tel que tous les terminaux sont situés sur le bord de la face extérieure de G. Soit r et c des fonctions de capacité telles que pour tout noeud interne (pas sur le bord de la face extérieure) la loi de Kirchhoff soit vérifiée. Alors, il existe une solution au problème du multiflot entier (G, H, r, c) ssi :*

$$\sum_{X \in \mathcal{X}} (c(\delta_G(X)) - r(\delta_H(X))) \geq \frac{q}{2} \tag{3.1}$$

pour toute collection $\mathcal{X}$ d'ensembles disjoints de sommets, avec q le nombre de composantes connexes K de $G - (\bigcup_{X \in \mathcal{X}} \delta(X))$ telles que $c(\delta_G(K)) + r(\delta_H(K))$ est impair.

L'argument réside dans le fait qu'une solution au problème de multiflot entier laissera inutilisé un T-joint de G, avec T l'ensemble des sommets impairs. Ce T-joint contient des chemins disjoints couplant les sommets de T, on peut donc ajouter un ensemble bien choisi de demandes tel que le problème reste faisable. En fait, la condition exprime qu'on peut effectivement trouver un ensemble de demandes supplémentaires, rendant le graphe Eulérien, sans rompre la condition de coupe.

Dans [57], problème 56, Schrijver pose la question de déterminer la complexité en retirant totalement la condition d'Euléricité. Schwärlzer [58] a récemment résolu ce problème en 2007, en proposant une preuve de NP-complétude étonnamment simple, que nous reprenons ici en esquissant la preuve afin de préparer le résultat suivant. Schwärzler ne montre pas la version avec seulement 3 arêtes de demande, mais cette restriction s'obtient naturellement à partir de sa réduction.

Théorème 3.0.11 (Schwärlzer, 2007)**.** *Le problème de multiflot entier avec graphe d'offre planaire est NP-complet, même si tous les terminaux sont sur le bord d'une même face, et même si le nombre d'arêtes de demande est limité à 3.*

Preuve. (esquisse) De nouveau, la réduction se fait depuis SATISFAISABILITÉ, d'une façon similaire à la preuve du Théorème 2.2.1. Soit φ une formule à n clauses et p variables. Le graphe construit prend la forme d'une grille, comprenant une ligne par variable et une colonne par clause. Selon l'appartenance positive ou négative, ou bien l'absence, d'une variable dans une clause, le sous-graphe définissant l'intersection de la ligne et la colonne correspondante est l'un des trois graphes présentés en Figure 3.1. Nous ajoutons 4 terminaux par colonne, correspondant aux quatres sommets de degré 1

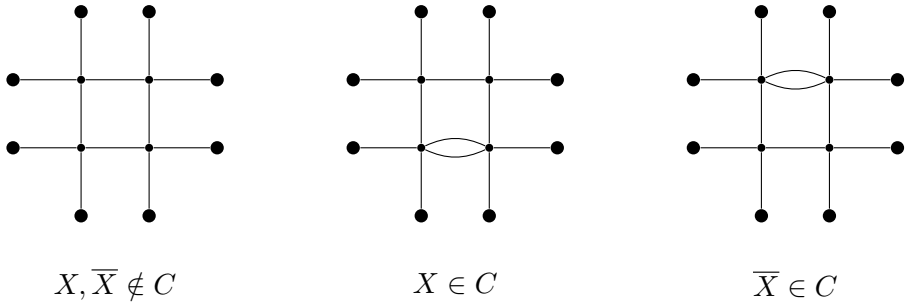


FIGURE 3.1 – *De gauche à droite, le graphe correspondant à l'absence de la variable dans la clause, celui lorsque la variable est présente positivement, celui lorsqu'elle est présente négativement.*

restant dans chaque colonne, et pour chaque ligne, 2 sommets supplémentaires, un de chaque côté, reliés aux deux sommets de chaque extrémité, comme dans la Figure 3.2.

Dans chaque ligne (et donc pour chaque variable), va transiter un chemin, soit par les arêtes du haut (variable mise à *vrai*), soit par les arêtes du bas (variable mise à *faux*). Dans chaque colonne (et donc pour chaque clause), ce sont deux chemins qui vont passer, l'un du côté droit, l'autre du côté gauche. Cependant, les terminaux de ces deux chemins sont inversés, pour obliger ces chemins à changer de côté un nombre impair de fois, donc au moins une fois. Bien entendu, l'idée de cette preuve est de faire en sorte que ce croisement ne puisse se produire que dans la ligne correspondant à une variable validant cette clause. Les demandes sont explicitées sur la Figure 3.2.
L'essentiel de la preuve tient dans les arguments suivants. Les coupes horizontales sont serrées, donc chaque gadget voit effectivement "passer" deux chemins verticaux. Chaque gadget, vu comme ensemble de 4 sommets, définit une coupe de cardinalité 8. 4 de ces arêtes sont utilisées par les chemins verticaux. Pour chaque ligne, nous pourrions montrer itérativement que 2 autres sont utilisés par le chemin horizontal de la ligne correspondante, et les deux dernières sont inutilisées. Enfin, pour conclure la preuve, il suffit de remarquer qu'un chemin horizontal ne peut utiliser d'arête verticale, et que les chemins verticaux de chaque colonne ne peuvent se croiser qu'au niveau des arêtes doubles.

Cette réduction utilise un nombre non-limité de terminaux. Pour atteindre seulement trois terminaux, il nous faut utiliser une astuce supplémen-

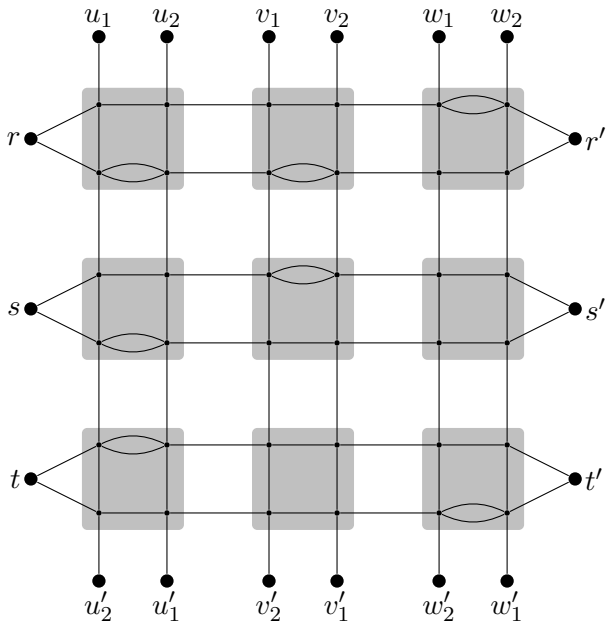


FIGURE 3.2 – *Un exemple de la réduction, depuis la formule* $(X \vee Y \vee \neg Z) \wedge (X \vee \neg Y) \wedge (\neg X \vee Z)$. *Le graphe dessiné est celui d'offre, la demande correspond aux arêtes entre un sommet nommé et sa version prime.*

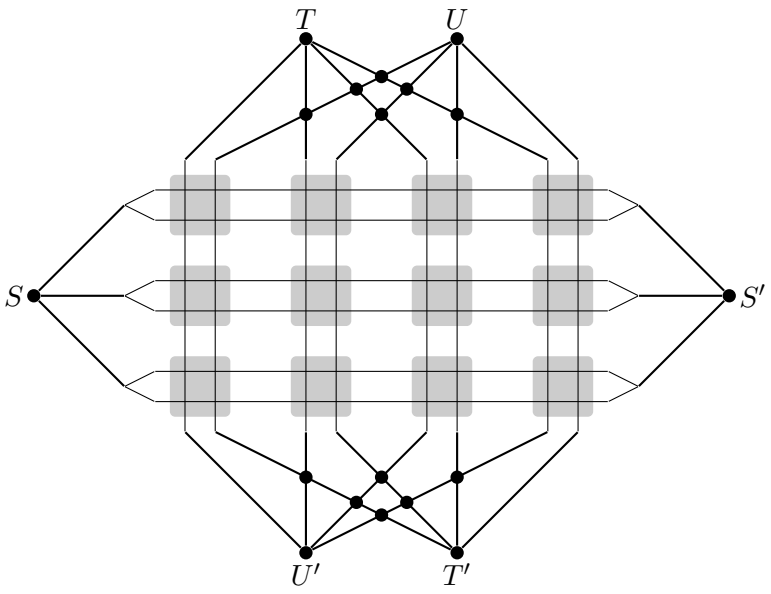


FIGURE 3.3 – *En ajoutant des terminaux, et des arêtes se croisant "en ligne droite", le nombre d'arêtes de demande est réduit à* 3.

taire. Remarquons que cette réduction comprend fondamentalement trois types de chemins, les chemins horizontaux, les chemins verticaux qui vont du haut à droite vers le bas à gauche, et les chemins verticaux dans l'autre sens. Nous ajoutons donc une paire de terminaux pour chacun de ces types de chemins, liés aux anciens terminaux. Malheureusement, la planarité n'est pas conservée par cette opération. Mais en choisissant convenablement comment croiser les nouvelles arêtes, nous allons nous assurer de l'équivalence avec la version non-planaire. Les arêtes sont disposées très simplement, comme illustré sur la Figure 3.3.

La correction de cet ajout provient de l'argument suivant. Tous les chemins de T à T' doivent croiser tous les chemins de U à U'. Comme il y a n chemins de chaque sorte, il faut n^2 croisements de chemins. Comme il ne peut y en avoir qu'un dans chacune des n colonnes, il en faut $n(n-1)$ par ailleurs. C'est exactement le nombre de sommets de degré 4 qui ont été ajoutés. Le routage dans chacun de ces sommets est donc évident. Pour finir, remarquons qu'avec ce routage, les chemins passent bien par les terminaux de la grille, comme dans le début de la preuve. □

Bien sûr, avec une seule demande, le problème consiste en un simple

flot, et même avec deux demandes mais une requête bornée, le problème est polynomial par le résultat de Robertson et Seymour [51]. Il reste donc la complexité pour deux arêtes de demande, sans borne sur la requête totale, à déterminer. C'est l'objet de la prochaine partie.

Chapitre 4

Biflots entiers dans le plan

Dans son article de 2006, Müller [42] montre que le problème de trouver deux flots disjoints dans un graphe orienté planaire est NP-complet, et pose la question pour la version non-orientée. Nous donnons ici la réponse à cette question, en prouvant la NP-complétude du problème.

4.1 Présentation de la réduction

Nous reprenons le schéma de preuve utilisé auparavant, depuis 3-SAT, en utilisant une grille constituée de sous-graphes. Avant d'écrire la preuve détaillée, qui est très technique, tentons de donner les principales idées de la réduction. Prenons une grille avec autant de colonnes que le nombre de clauses, et deux fois plus de lignes que le nombre de variables. Supposons qu'il nous soit donné deux sous-graphes de la forme suivante. Chacun a deux sommets de degré 1 à sa droite et à sa gauche, 4 en haut et en bas. Tout chemin routé entre deux de ses sommets de degré 1, l'est entre un sommet de la droite et un de la gauche, ou bien entre un du haut et un du bas (on respecte donc le comportement effectif constaté dans la preuve du Théorème 3.0.11, nous sommes bien en présence de chemins verticaux et horizontaux).
De plus nos deux sous-graphes possèdent les propriétés suivantes. Le premier, appelons-le XCH, est tel que si un ou deux chemins horizontaux et deux chemins verticaux le traversent, les deux chemins verticaux vont "en diagonale", des deux extrémités en haut à droite vers les deux extrémités en bas à gauche, ou à l'inverse de haut à gauche en bas à droite. Le second sous-graphe, LIC vérifie la même propriété si deux chemins horizontaux et verticaux le traversent. Néanmoins, si un seul chemin horizontal et deux chemins verticaux le traversent, les chemins verticaux peuvent avoir des ex-

trémités quelconques. Les Figures 4.1 et 4.2 résument ces propriétés.

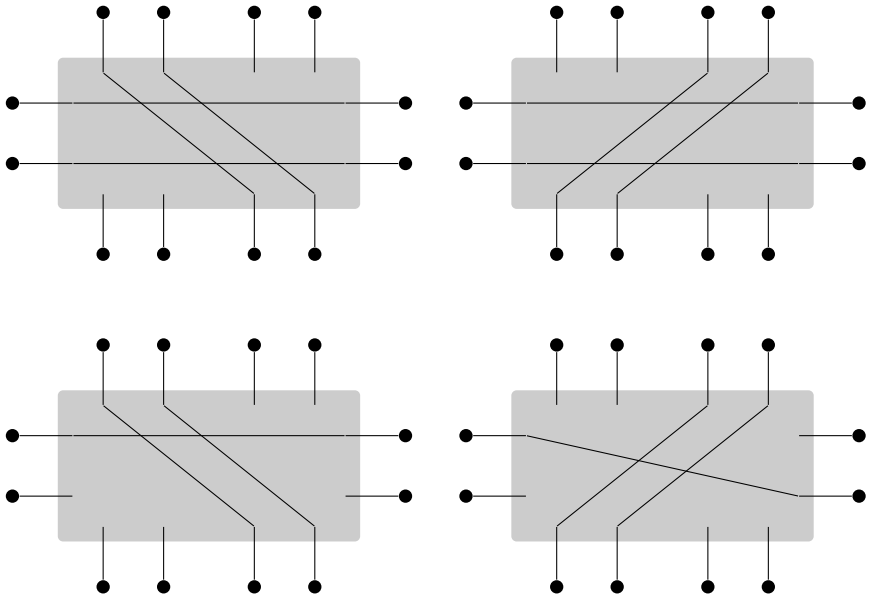


FIGURE 4.1 – *Routages permis par les sous-graphes XCH et LIC. Chaque fois, les deux chemins verticaux changent de côté.*

Le comportement "normal" de ces gadgets est de faire changer les deux chemins verticaux de côté. On dit dans ce cas que le gadget *décale* les chemins. Dans certains cas cependant, les chemins verticaux vont tout droit, on dit que le gadget *garde* les chemins. Nous encodons le fait qu'*une clause est satisfaite ainsi : le chemin a été gardé un nombre impair de fois* dans sa colonne (donc au moins une fois). Comme le nombre de lignes est pair, il a donc été décalé un nombre impair de fois. Il faut donc, pour vérifier la satisfaction d'une clause, router des chemins entre les extrémités droites du haut de la colonne et les extrémités gauches du bas de la colonne.
Bien sûr, pour que cela soit correct, il faut que les chemins soient gardés seulement s'il se trouve une variable qui satisfait la clause. Nous disposons de deux lignes pour encoder chaque variable (ces deux lignes sont prises consécutives). Nous allons y router trois demandes, qui vont donc se repartir soit deux sur le première ligne et une sur la seconde, soit l'inverse. Cela correspond dans le premier cas à assigner la valeur *vrai* à la variable, dans le second cas *faux*. Puisque pour qu'un gadget garde les chemins verticaux, il faut à la fois que ce soit un LIC et qu'un seul chemin horizontal y passe, il

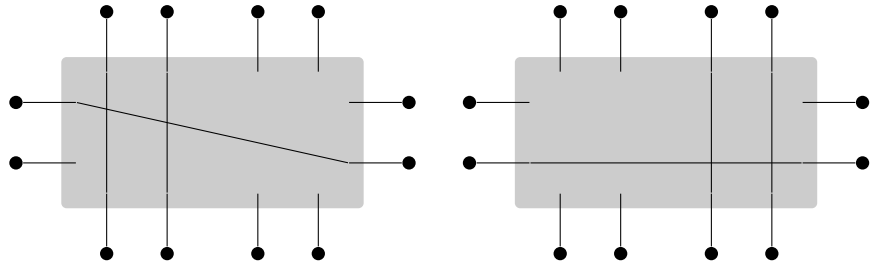


FIGURE 4.2 – *Routages supplémentaires permis par le sous-graphe LIC. Lorsqu'un seul chemin horizontal passe, les deux chemins verticaux peuvent rester du même côté.*

suffit de choisir maintenant les gadgets de la manière suivante. Si une variable apparaît positivement dans une clause, l'intersection de la deuxième ligne de la variable avec la colonne de cette clause est un LIC. Ainsi, si on choisit l'assignation *vrai* pour cette variable, on pourra garder les chemins dans ce sous-graphe. De même, si une variable apparaît négativement dans une clause, l'intersection de la colonne de cette clause et de la première ligne de cette variable est aussi un LIC. Dans tous les autres cas, nous plaçons un XCH, ce qui oblige les chemins à être décalés.
L'équivalence entre existence de chemins et satisfaisabilité de la formule provient donc de *l'obligation de garder au moins une fois dans chaque colonne* (notons qu'il n'est jamais obligatoire de décaler, on peut juste garder les chemins verticaux dans la première ligne où les conditions sont réunies, puis toujours les décaler ensuite).
Finalement, nous pouvons facilement ajouter quatre nouveaux sommets, qui seraient les terminaux des arêtes de demande, une arête pour les demandes de chemins verticaux, et une autre pour les chemins horizontaux. Malheureusement, ce schéma de preuve relativement simple ne passe pas en pratique, car nous ne connaissons pas de sous-graphes avec les propriétés de XCH et LIC telles que nous les avons énoncées. La principale difficulté provient de la possibilité pour les chemins verticaux d'utiliser des arêtes horizontales, et symétriquement pour les chemins horizontaux d'utiliser des arêtes verticales. Là où dans la preuve du théorème de Schwärzler, les coupes verticales sont serrées, les nôtres possèdent deux fois plus d'arêtes d'offre que nécessaire, et les coupes horizontales ne sont pas serrées non plus.
Pour résoudre ces difficultés, nous adoptons une solution en deux points. Premièrement, nous allons très fortement augmenter le nombre de lignes. En

effet, une étude plus précise des gadgets va montrer que la liberté donnée par le surplus d'arêtes dans les coupes horizontales est en fait assez limité, permettant des variations relativement faibles (Lemme 4.4.5). *En ajoutant de nombreuses lignes, qui vont avoir le rôle de tampon entre les lignes de chaque variable*, ces perturbations vont rester locales et maitrisables. Deuxièmement, pour faciliter l'étude de ces perturbations, nous allons construire *un graphe "presque" Eulérien, avec très peu de sommets impairs.* En étudiant les arêtes qui ne sont pas utilisées dans une solution, cette hypothèse va nous permettre de rétablir des propriétés sur les coupes verticales (Lemme 4.6.2). Ceci se révèlera fondamental pour montrer que *les chemins verticaux n'utilisent pas d'arêtes des coupes horizontales.* Une fois ceci prouvé, quelques lemmes techniques sur le comportement des chemins horizontaux permettront de terminer la preuve.

4.2 Préliminaires

Avant même de rentrer dans les détails de la preuve, nous donnons deux lemmes, le premier pour simplifier les considérations sur le comportement des chemins, et le second est essentiel pour obtenir la parité de degré sur les sommets du graphe de la réduction.

4.2.1 Décroisement de chemins

Lemme 4.2.1. *Soit (G, H) une instance du problème de chemins arête-disjoints avec G planaire. Alors s'il existe une solution à l'instance (G, H), il existe une solution telle que toute paire de chemins de la solution se croisent au plus une fois, et les chemins ayant une extrémité commune ne se croisent pas.*

Preuve. La preuve est similaire à celle du Lemme 2.3.1. □

Dans la suite de ce document, on supposera toujours notre solution décroisée (donc telle que deux chemins se croisent au plus une fois), et constituée de chemins simples. Voici une conséquence du décroisement qui se révèlera utile :

Lemme 4.2.2. *Soit G un graphe planaire, a, b, c et d quatre sommets de la face extérieure de G apparaissant dans cet ordre. Soit $\mathcal{P}$ un ensemble décroisé de (a, c)-chemins et (b, d)-chemins arête-disjoints. Alors, tous les (a, c)-chemins croisent les (b, d)-chemins dans le même ordre.*

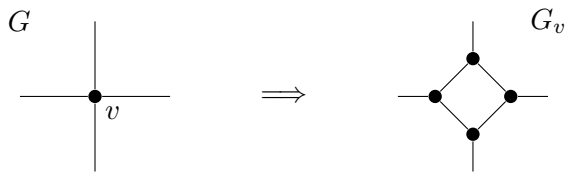


FIGURE 4.3 – *À gauche, le sommet v du graphe G, sur lequel on veut interdire les croisements de chemins. À droite, la transformation effectuée, donnant le graphe G_v.*

Preuve. Les (b, d)-chemins forment un flot entre b et d, donc ces chemins ne sont pas croisés. De même les (a, c)-chemins ne sont pas croisés. Supposons qu'il existe au moins un (b, d)-chemin, et un (a, c)-chemin P. Soit Q le premier chemin croisé par P depuis l'extrémité a. Puisque les (b, d)-chemins ne se croisent pas, Q définit une courbe non-fermée qui sépare a des autres (b, d)-chemins. Donc tous les (a, c)-chemins croisent Q en premier. Par induction sur le nombre de (b, d)-chemins, en enlevant Q, ils croisent aussi tous les autres chemins dans le même ordre. □

4.2.2 Sommets sans croisements

Comme décrit dans la présentation de la preuve, on va vouloir s'assurer que le graphe est essentiellement Eulérien. Malheureusement, il s'agit d'une contrainte très forte (voir le Théorème 3.0.10), qui ne peut s'appliquer directement. Nous allons plutôt construire un graphe dont quasiment tous les sommets sont de degré 4, mais en contrepartie, certains de ces sommets auront la propriété suivante : deux chemins ne peuvent pas se croiser sur l'un de ces sommets. Nous montrons que cette contrainte ne trahit pas la nature du problème, puisqu'un petit gadget permet d'assurer le respect de cette contrainte. Pour cela, il suffit de remplacer chacun de ces sommets par un cycle de longueur 4, dont chaque sommet est incident à l'une des arêtes du sommet remplacé (voir la Figure 4.3).

On définit formellement le problème suivant, avant de montrer son équivalence avec le problème classique.

Problème 4.2.3. *(Chemins arête-disjoints étendu dans le plan)*
ENTRÉE *: un graphe planaire G, un graphe de demande H avec $V(H) \subseteq V(G)$, un sous-ensemble U de sommets de G.*
SORTIE *: Existe-t-il une solution au problème de chemins arête-disjoints*

(G, H) telle que pour tout sommet $u \in U$, les chemins passant par u ne se croisent pas ?

Notons que nous avons déjà déterminé en Section 1.3 que nous pouvons nous restreindre à des graphes de degré maximum 4. De plus dans le problème précédent, si $U = \emptyset$, le problème est exactement le problème de chemins arête-disjoints dans le plan. Nous ne démontrons pas le lemme suivant, prouvant l'équivalence des deux problèmes :

Lemme 4.2.4. *Soit (G, H, U) une instance du problème de chemins arête-disjoints étendu dans le plan. Soit v un sommet de G de degré 4. Alors il existe une solution au problème (G_v, H, U) si et seulement s'il existe une solution au problème $(G, H, U \cup \{v\})$.*

Par la suite, nous utiliserons toujours la version étendue. Les sommets de U seront appelés *sommets sans croisement*. La plupart des sommets de nos graphes seront effectivement sans croisement. Les sommets qui ne sont pas dans U seront représentés en gros, et seront appelés *sommets de croisement*. Remarquons que la preuve qui suit permet de nier l'existence d'un éventuel "théorème de Frank", similaire au Théorème 3.0.10) pour le cas étendu, notre réduction pouvant s'adapter pour être internement Eulérien (eulérien pour tous les sommets sauf ceux sur le bord de la face extérieure).

4.3 Implémentation des gadgets

Nous appelons *non-chemin* dans un graphe G et pour un ensemble de chemins arête-disjoints $\mathcal{P}$, tout chemin du complémentaire de $\mathcal{P}$ dans G. Autrement dit, un non-chemin est un chemin arête-disjoint de $\mathcal{P}$, mais n'est pas soumis à la condition de ne pas croiser d'autre chemins dans des sommets sans croisement.

4.4 Graphes élémentaires

Nous entrons maintenant dans les détails de la preuve. D'abord, nous donnons l'implémentation des sous-graphes LIC et XCH, et détaillons leurs propriétés. Soit donc XCH le graphe présenté en Figure 4.4. Ses sommets de croisement sont a, b, c et d. Nous notons $S = \{s_1, s_2, s_3, s_4\}$, $S' = \{s'_1, s'_2, s'_3, s'_4\}$, $T = \{t_1, t_2\}$ et $T' = \{t'_1, t'_2\}$.

Lemme 4.4.1. *Soit $\mathcal{P} = \{S_1, S_2, T_1, T_2\}$ un ensemble décroisé de chemins arête-disjoints de XCH, tels que :*

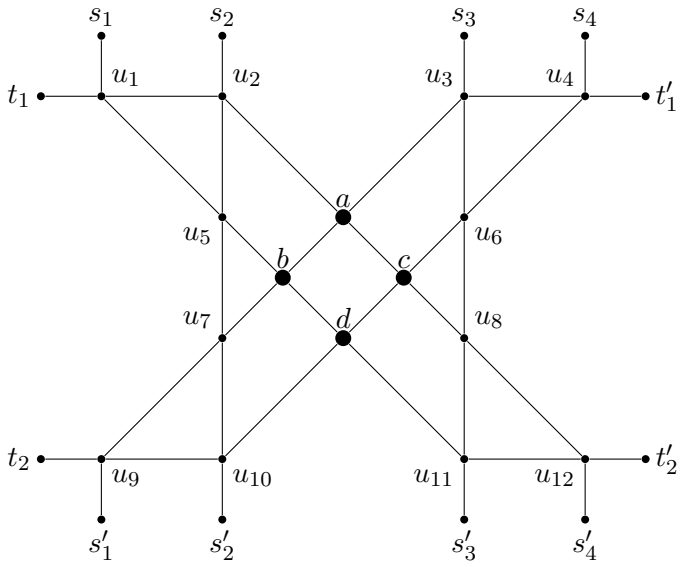


FIGURE 4.4 – *L'implémentation du gadget XCH. Les sommets de croisements sont les quatre sommets centraux, a, b, c et d. Hormis les* 12 *terminaux (de degré* 1*), tous les sommets sont bien de degré* 4*.*

(*i*) *S_1 et S_2 sont des (S, S')-chemins,*

(*ii*) *T_1 et T_2 sont des (T, T')-chemins.*

Alors, S_1 et S_2 sont ou bien un (s_1, s'_3)-chemin et un (s_2, s'_4)-chemin, ou bien un (s_3, s'_1)-chemin et un (s_4, s'_2)-chemin.

Preuve. Soit $\mathcal{P}$ tel que dans l'énoncé du lemme. Chaque (S, S')-chemin doit croiser chaque (T, T')-chemins, ce qui nécessite au moins 4 croisements. Or, XCH possède exactement 4 sommets de croisement, donc chacun est effectivement utilisé pour un croisement.
Supposons que ab appartienne à un (T, T')-chemin, alors c'est aussi le cas de u_3a, bu_7, u_6c, cd et du_{10}, et les arêtes u_2a, ac, cu_8, u_5b, bd et du_{11} appartiennent aux (S, S')-chemins. Du coup, les (S, S')-chemins sont connectés à s_1, s_2, s'_3 et s'_4. Similairement, si ab appartient à un (S, S')-chemin, nous obtenons la seconde solution. □

Ces chemins existent, comme le montre la Figure 4.5.

Lemme 4.4.2. *Soit $\mathcal{P}$ un ensemble de trois chemins arête-disjoints de XCH, tel que :*

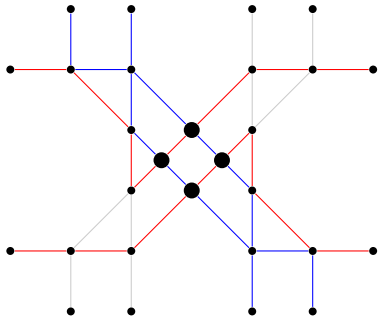


FIGURE 4.5 – *Existence des chemins prescrits par le Lemme 4.4.1, conformément à ce qu'on attend du gadget XCH.*

(*i*) $\mathcal{P}$ *contient exactement* 2 $(\{s_1, s_2\}, S'')$*-chemins, et* S'' *est* $\{s'_1, s'_2\}$ *ou* $\{s'_3, s'_4\}$,

(*ii*) $\mathcal{P}$ *contient exactement* 1 (T, T')*-chemin.*

Alors, $S'' = \{s'_3, s'_4\}$.

On constate déjà ici des différences avec le modèle théorique de la Section 4.1, puisqu'il faut supposer en plus que les chemins verticaux rentrent bien du même côté à chaque fois, s'il n'y a qu'un seul chemin horizontal. Concrètement, ceci sera assuré par le Lemme 4.4.1, puisque les gadget au-dessus et au-dessous de celui-ci auront forcément deux chemins horizontaux.

Preuve. Sinon, $\mathcal{P}$ contiendrait un (T, T')-chemin Q, P_1 un (s_1, s'_1)-chemin, et P_2 un (s_2, s'_2)-chemin. Comme Q doit croiser les deux autres chemins, tous les chemins de $\mathcal{P}$ passe par au moins un sommet parmi a, b, c, d. Donc les arêtes u_2a, u_5b, u_7b et $u_{10}d$ sont utilisées 5 fois en tout, ce qui contredit la disjonction des chemins. □

Soit LIC le graphe de la Figure 4.6. Nous notons encore $S = \{s_1, s_2, s_3, s_4\}$, $S' = \{s'_1, s'_2, s'_3, s'_4\}$, $T = \{t_1, t_2\}$ et $T' = \{t'_1, t'_2\}$.

Lemme 4.4.3. *Soit* $\mathcal{P}$ *un ensemble de chemins arête-disjoints dans LIC, tel que :*

(*i*) $\mathcal{P}$ *contient exactement* 2 $(\{s_1, s_2\}, S'')$*-chemins, où* S'' *est soit* $\{s'_1, s'_2\}$, *soit* $\{s'_3, s'_4\}$,

(*ii*) $\mathcal{P}$ *contient exactement* 2 (T, T')*-chemins.*

Alors, $S'' = \{s'_3, s'_4\}$. *De plus, on ne peut trouver un* $(S \cup T, S' \cup T')$*-chemin disjoint de* $\mathcal{P}$.

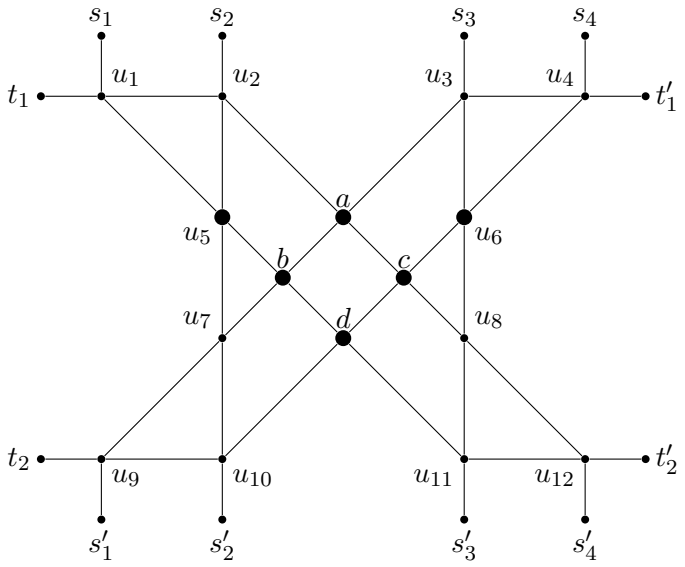


FIGURE 4.6 – *L'implémentation du gadget LIC. Hormis deux sommets de croisements supplémentaires, il est identique au graphe XCH.*

Preuve. Supposons qu'au contraire, $S'' = \{s'_1, s'_2\}$. Notons Q le (t_2, T')-chemins, P_1 le (s_1, s'_1)-chemin et P_2 le (s_2, s'_2)-chemin. Soit C la coupe définie par les sommets $\{u_7, u_9, u_{10}\}$. Ces trois chemins passent par C, qui ne possède pas de sommets de croisement. Comme $d(C) = 6$, u_5u_7 est utilisé par le chemin Q. Quatre chemins passent donc par les sommets $C' = \{u_1, u_2, u_5\}$, or $d(C') = 6$, contradiction. Donc $S'' = \{s'_3, s'_4\}$. Dans ce cas, toutes les arêtes incidentes à a, b, c et d sont utilisées par $\mathcal{P}$ (coupes serrées), et u_6u_8 est utilisé par le (t_2, t'_2)-chemin, il ne peut donc y avoir d'autres chemins entre $S \cup T$ et $S' \cup T'$. □

Lemme 4.4.4. *Il est possible de trouver dans LIC des ensembles de chemins arête-disjoints pour ces demandes :*

1. $s_1s'_1$, $s_2s'_2$, t_1t' *pour tout* $t' \in T'$,
2. $s_3s'_3$, $s_4s'_4$, tt'_1 *pour tout* $t \in T$.

Preuve. Voir la Figure 4.7. □

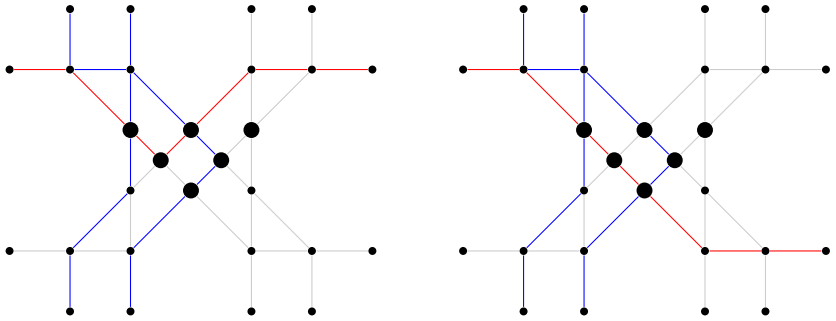


FIGURE 4.7 – *Existence des chemins pour le Lemme 4.4.4. LIC vérifie bien les propriétés décrites dans la Section 4.1.*

4.4.1 Agrégats de gadgets

Nous pouvons agréger les sous-graphes LIC et XCH en grille, comme nous l'avons déjà fait pour d'autres gadgets, en identifiant les sommets de degré 1 qui se correspondent. La Figure 4.8 montre comment constituer ces grilles, et donne les notations qui seront utilisées à présent. De plus, nous notons X l'ensemble des x_i, X' l'ensemble des x'_i, pour $i \in [\![1, 4n]\!]$ avec n le nombre de colonnes, et de même Y et Y' les ensembles des y_j et y'_j, avec $j \in [\![1, 2p]\!]$, p le nombre de ligne.

Lemme 4.4.5. *Soit G une grille de dimension 1×3, constituée exclusivement de LIC. Soit $\mathcal{P}$ un ensemble de chemins arête-disjoints, comprenant :*
- *A et B des (X, X')-chemins,*
- *C un $(\{y_1, y_2\}, y'_1)$-chemin,*
- *D un (y_3, y'_2)-chemin,*
- *E un (y_4, y'_3)-chemin,*
- *F un (y_5, y'_4)-chemin,*
- *H un (y_6, y'_5)-chemin,*
- *I un (X', y'_6)-chemin.*

Alors, dans $G \setminus E(\mathcal{P})$, il n'existe pas de $(\{y_1, y_2\}, X')$-non-chemins.

Notons que le lemme reste vrai en remplaçant certains LIC par des XCH, qui sont moins permissifs. La Figure 4.9 illustre les placements des chemins.

Preuve. Supposons qu'il existe un non-chemin Q. Sans perte de généralité, $\mathcal{P}$ est supposé décroisé.
En raison des coupes serrées, $M(1, 1)$ contient C, $M(1, 2)$ contient E, and $M(1, 3)$ contient H. Dans $M(1, 2)$, E passe par u_6, (u_3 et u_4 sont des sommets

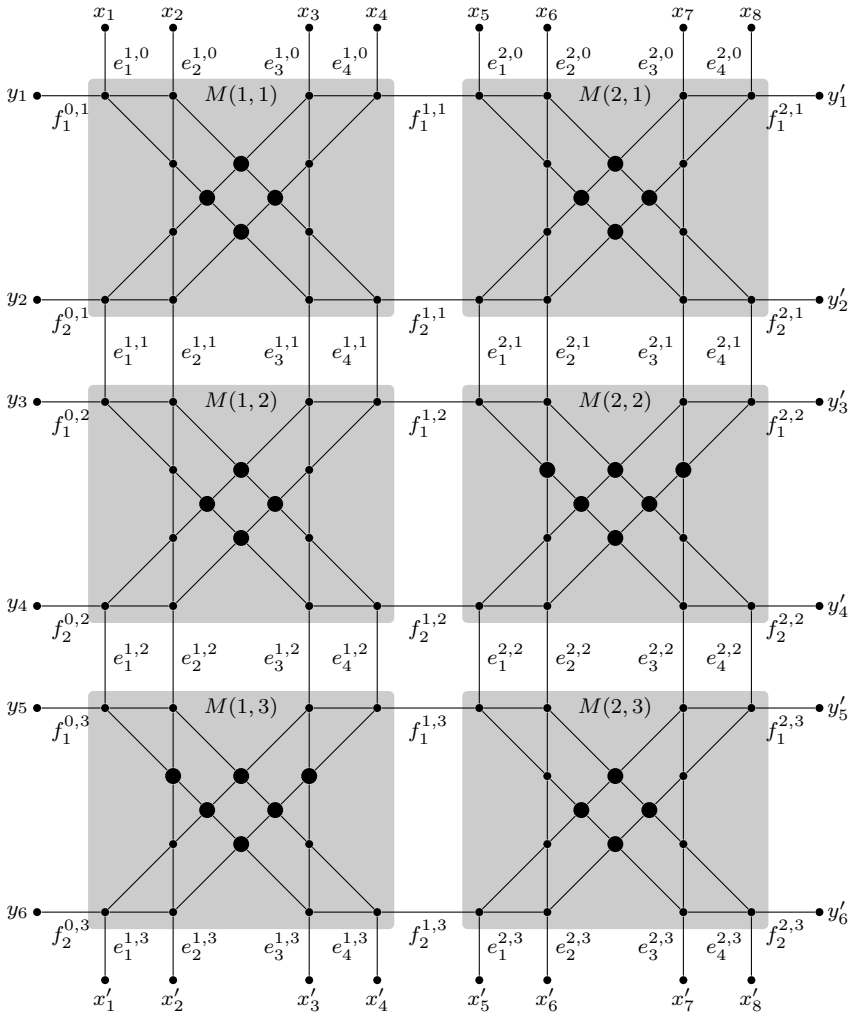


FIGURE 4.8 – *Une grille de dimension* 2×3. *Les notations utilisées par la suite font référence à celle définie ici.*

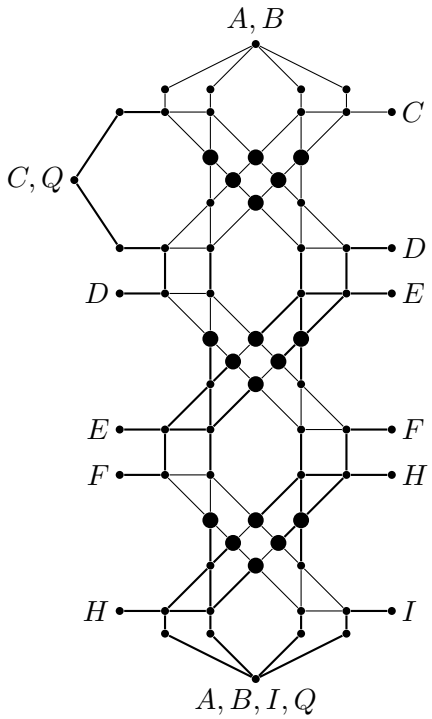


FIGURE 4.9 – *Il n'existe pas de solution à ce problème de chemins arête-disjoints, même si le chemin Q peut croiser les autres chemins dans n'importe quel sommet.*

sans croisement, et l'un des chemins A, B ou D passe par ces sommets) et utilise u_6c ou u_6u_8. De même, E passe par u_7 and utilise u_7u_5 ou u_7b. Considérons dans $M(1,2)$ les coupes $C_1 := \{u_5u_7, u_5b, ab, ac, u_6c, u_6u_8\}$ et $C_2 := \{u_5u_7, u_7b, bd, cd, cu_8, u_6u_8\}$, chacune intersecté par 4 chemins. E doit utilisé au moins trois arêtes dans l'une de ces coupes, puisqu'il ne passe ni par $u_{10}d$ ni par au_3. Donc F ne peut intersecter C_1, prouvant qu'il ne peut croiser A ou B qu'en d dans $M(1,2)$.
Par des arguments similaires, dans $M(1,3)$, en considérant H et les deux mêmes coupes pour $M(1,3)$ (notons-les C_1' et C_2'), F peut croiser A et B en a seulement. Par le Lemme 4.2.2 et en raison de la coupe serrée entre $M(1,2)$ et $M(1,3)$, F croise A en a dans $M(1,3)$ puis croise B en d de $M(1,2)$. Comme F ne peut utiliser d'arêtes dans C_1 et C_2', F doit traverser d de $M(1,2)$ et a de $M(1,3)$ de gauche à droite ou de droite à gauche, un nombre impair de fois, ce qui constitue une contradiction. □

Ce lemme permettra de montrer que les chemins horizontaux ne changent pas trop de lignes. Pour finir l'étude du comportement local des chemins, nous avons besoin d'un dernier lemme, qui permettra de vérifier que les chemins verticaux sont bien systématiquement décalés lorsqu'il n'y a pas de graphes LIC.

Lemme 4.4.6. *Soit G une grille de dimension 1×3, constituée exclusivement de XCH. Alors, il n'existe pas d'ensemble de chemins arête-disjoints comprenant cinq (Y, Y')-chemins, un (x_1, x_1')-chemin et un (x_2, x_2')-chemin.*

Preuve. Soient L et R tels que définis par la Figure 4.10, et supposons que contrairement à l'énoncé du lemme, les chemins existent. Appelons A le (x_1, x_1')-chemin, B le (x_2, x_2')-chemin, et $S_1, \ldots, S_5$ les autres chemins. On suppose que ces chemins sont décroisés, 10 croisements sont quand même nécessaire. Puisqu'il existe seulement 12 sommets de croisement, et qu'un chemin ne peut en croiser qu'au plus 2 autres dans chacun des gadgets (facile à vérifier), A et B sont obligés de passer parmi les sommets de croisement de chaque gadget. Donc $|A \cap (\delta(R) \cup \delta(L))| \geq 6$ et $|B \cap (\delta(R) \cup \delta(L))| \geq 6$. De plus, chacun des S_i utilisent au moins une arête de $\delta(L)$ et $\delta(R)$, donc au moins 22 arêtes sur 24 sont utilisées. Par parité, les deux dernières arêtes sont inutilisées, une dans chaque coupe, et $|A \cap \delta(L)| = |B \cap \delta(R)| = 4$, $|A \cap \delta(R)| = |B \cap \delta(L)| = 4$. Du coup, $|\delta(L) \cap E(M(1,2)) \cap (A \cup B)| = 1$ et $|\delta(R) \cap E(M(1,2)) \cap (A \cup B)| = 3$. Comme une seule arête de $\delta(L)$ est inutilisée, au moins deux chemins parmi les S_i passent par les sommets de croisement de $M(1,3)$, mais un seul peut passer par $\delta(R) \cap E(M(1,2))$, contradiction. □

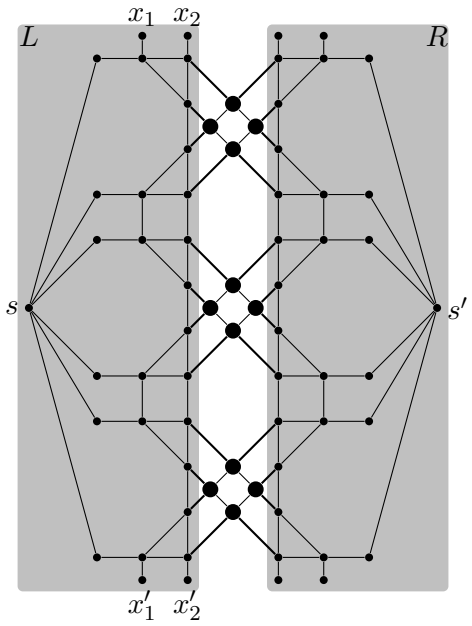


FIGURE 4.10 – *Il n'existe pas de solutions à ce problème de chemins arête-disjoints, avec* 5 *chemins de* s *à* s'*, c'est ce que montre le Lemme 4.4.6*

4.5 Construction du graphe

Soit φ une formule booléenne en forme normale conjonctive, chaque clause se composant de 3 littéraux, comprenant n clauses sur un ensemble de p' variables. Nous encodons cette formule dans un graphe G_φ, construit à partir d'une grille à n colonnes. Nous avons besoin de 2 lignes par variables, et entre chacune de ces lignes nous allons ajouter un tampon de $q = 12n+4np'+2$ lignes. La grille comprend donc $p = 2p'(1+q)$ lignes. La $i^{\text{ième}}$ variable est encodée au lignes $1+2(i-1)(1+q)$ et $q+2+2(i-1)(1+q)$. Les autres lignes sont composées entièrement de graphes XCH. La ligne $1+2(i-1)(1+q)$ code l'assignement *vrai* pour la $i^{\text{ième}}$ variable, et doit donc comporter un graphe LIC dans les colonnes correspondant à une apparition positive de cette variable. Dans les autres colonnes, le sous-graphe choisit est XCH. De même, la ligne $q+2+2(i-1)(1+q)$ code l'assignement *faux* pour la $i^{\text{ième}}$ variable, et doit donc comporter un graphe LIC dans les colonnes correspondant à une apparition négative de cette variable. En résumé, $M(i,j)$ est un LIC ssi :

- soit $j = 1+2(k-1)(1+q)$ et la $k^{\text{ième}}$ variable apparaît positivement dans la $j^{\text{ième}}$ clause.
- soit $j = q+2+2(k-1)(1+q)$ et la $k^{\text{ième}}$ variable apparaît négativement dans la $j^{\text{ième}}$ clause.

Tous les autres gadgets sont des XCH. On ajoute deux terminaux pour les chemins verticaux, x et x', reliés respectivement à x_{4k+1} et x_{4k+2}, et à x'_{4k+3} et x'_{4k+4} pour tout $k \in [\![0, n-1]\!]$. Puisque le nombre de ligne p est pair, on force bien que les chemins verticaux doivent être gardés un nombre impair de fois. On ajoute aussi des arêtes $x_{4k+3}x_{4k+4}$ et $x'_{4k+1}x'_{4k+2}$ pour tout k, afin de réduire le nombre de sommets de degré impair.

Pour tout $i \in [\![1, p']\!]$, nous ajoutons deux nouveaux sommets w_i et w'_i, et les arêtes suivantes : $w_i y_j$ et $w'_i y'_j$ pour tout $j \in [\![4(i-1)(q+1)+1, 4(i-1)(q+1)+2q+4]\!]$. Enfin, on ajoute deux terminaux pour les chemins horizontaux, y et y', et les arêtes yw_i et $y'w'_i$ de capacité $2q+3$ pour tout i, ainsi que des arêtes entre y et tous les sommets de Y encore de degré 1, et entre y' et tous les sommets de Y' encore de degré 1. Notons que chaque w_i et w'_i a une arête de capacité $2q+3$ et $2q+4$ arêtes de capacité 1 incidentes, et est donc impair.

Enfin, nous ajoutons les demandes entre x et x' avec capacité $2n$, et entre y et y' avec capacité $2p-p'$, définissant H_φ. En particulier, les coupes $\{x\}$, $\{y\}$, $\{x'\}$ et $\{y'\}$ sont serrées, et seuls les sommets w_i et w'_i sont impairs. La Figure 4.11 décrit le placement des terminaux pour les lignes correspondant à la première variable.

On appelle chemins horizontaux ceux répondant à la demande yy', et ver-

ticaux ceux répondant à la demande xx'. Deux chemins sont dit parallèles s'ils satisfont la même demande, et perpendiculaires sinon.

4.6 Non-chemins

Soit une solution $\mathcal{P}$ au problème de chemins arête-disjoints (G_φ, H_φ). $W \cup W'$ est l'ensemble des sommets impairs de $G_\varphi + H_\varphi$, avec $W = \{w_i \ : \ i \in [\![1, p']\!]\}$ et $W' = \{w'_i \ : \ i \in [\![1, p']\!]\}$. On considère la solution $\mathcal{P}$ comme un ensemble de cycles contenant exactement une arête de H_φ chacun. Dans ce cas, les sommets dans $G_\varphi + H_\varphi \setminus E(\mathcal{P})$ ont même parité que les sommets de $G_\varphi + H_\varphi$, autrement dit le complémentaire de $\mathcal{P}$ forme un $W \cup W'$-joint Q dans G_φ. Nous montrons que Q se décompose en un ensemble de (w_i, w'_i)-chemins plus des cycles de G_φ.

Lemme 4.6.1. *Soit G une grille de dimension $n \times p$. Soit $\mathcal{P}$ un ensemble décroisé de (X, X')-chemins et (Y, Y')-chemins arête-disjoints. Supposons qu'il existe $i \in [\![2, p-2]\!]$ tel que pour tout $j \in [\![1, n]\!]$, $M(i, j)$ et $M(i+1, j)$ sont des graphes xch et qu'il y a exectement quatre croisements de chemins de $\mathcal{P}$ dans $M(i, j)$ et dans $M(i+1, j)$. Alors il n'existe pas de non-chemin entre un sommet de la ligne $i-1$ et un sommet de la ligne $i+2$.*

Preuve. Soit $i \in [\![2, p-2]\!]$ tel que dans l'énoncé, et supposons qu'il existe un non-chemin Q entre les lignes $i-1$ et $i+2$. Alors Q doit passer par u_5u_7 ou u_6u_8 dans un certain $M(j, i)$, disons u_6u_8 par symétrie, et ensuite pare$_3^{j,i}$, $e_4^{j,i}$, $e_1^{j+1,i}$ ou $e_2^{j+1,i}$ (puisque toutes les arêtes incidentes aux a, b, c et d sont utilisés par des chemins).
Soient P_1, P_2, P_3, P_4 les cheminspassant par $c^{j,i}u_8^{j,i}$, $d^{j,i}u_{11}^{j,i}$, $a^{j,i+1}u_3^{j,i+1}$, $c^{j,i+1}u_6^{j,i+1}$ respectivement. Considérons la coupe $C = \delta(U)$, avec $U = \{u_8^{j,i}, u_{11}^{j,i}, u_{12}^{j,i}, u_3^{j,i+1}, u_4^{j,i+1}, u_6^{j,i+1}\}$, intersectant ces quatre chemins et Q. Puisque $|C| = 8$, aux plus quatre chemins et non-chemins intersectent C. Par décroisement, P_1 et P_2 sont distinct et parallèle, de même que P_3 et P_4. Il n'existe pas de sommet de croisement dans U donc $P_2 = P_3$. Par le Lemme 4.2.1, P_1, P_2 et P_4 sont croisés par des chemins qui leurs sont orthogonaux, dans le même ordre, donc $P_1 = P_4$. Remarquons qu'alors $j \neq n$. Soit $U' = \{u_7^{j+1,i}, u_9^{j+1,i}, u_{10}^{j+1,i}, u_1^{j+1,i+1}, u_2^{j+1,i+1}, u_5^{j+1,i+1}\}$. Au moins l'un de P_1, P_2 et Q doit passer par U'. Par des arguments similaires, $\mathcal{P}$ contient des chemins P'_1 passant par $b^{j+1,i}u_7^{j+1,i}$ et $u_5^{j+1,i+1}b^{j+1,i+1}$, et P'_2 passant par $d^{j+1,i}u_{10}^{j+1,i}$ et $u_2^{j+1,i+1}a^{j+1,i+1}$. Exactement un des quatre chemins considérés doit utiliser deux arêtes parmi $u_6^{j,i}u_8^{j,i}$, $u_5^{j+1,i}u_7^{j+1,i}$, $u_6^{j,i+1}u_8^{j,i+1}$ et

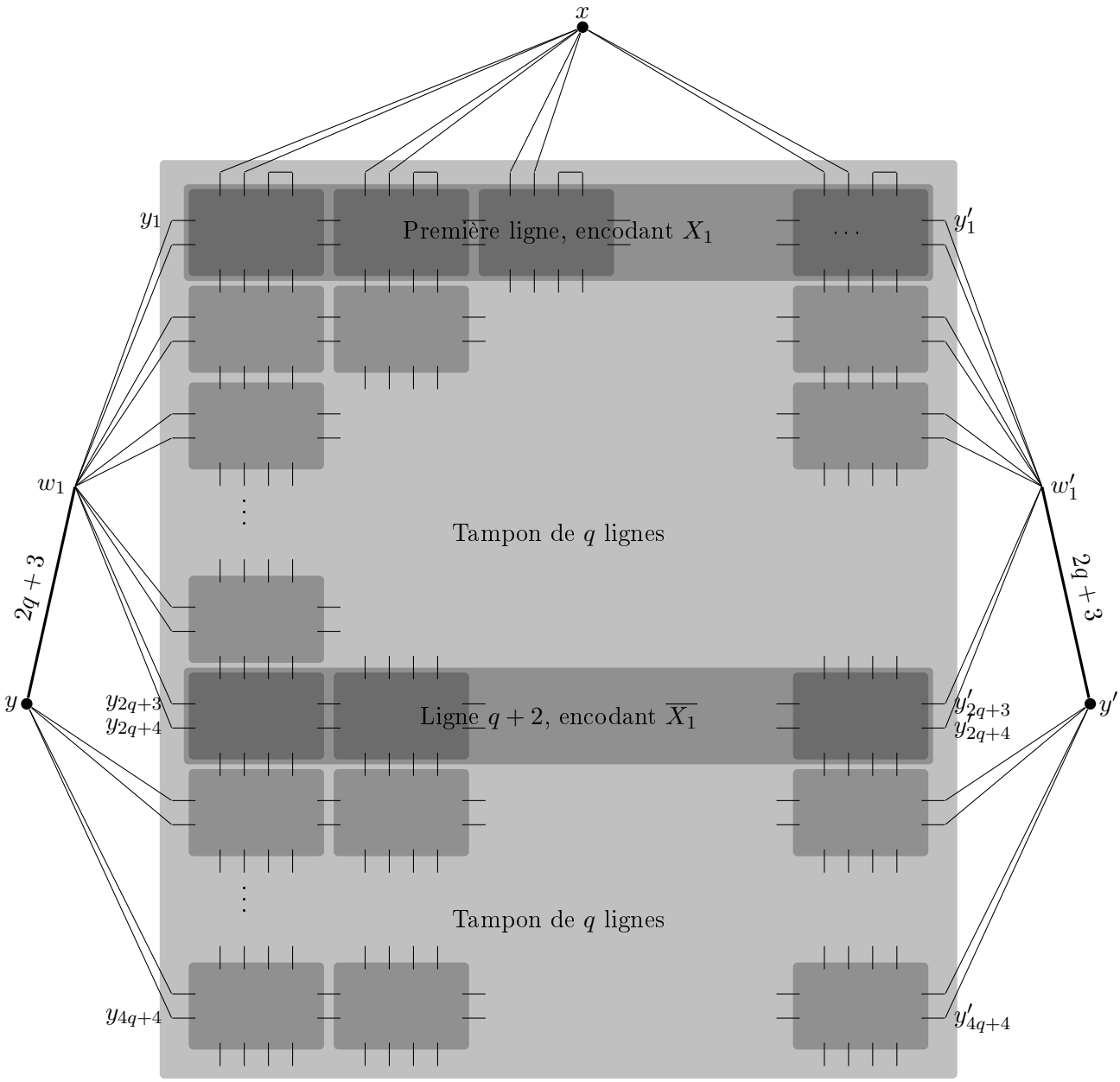


FIGURE 4.11 – *Schéma pour la construction de la grille, représentant les premières lignes, encodant la première variable. Le grand rectangle grisé se répète verticalement autant de fois que le nombre de variables.*

$u_5^{j+1,i+1}u_7^{j+1,i+1}$, disons P_1 (les autres cas sont isomorphes). Alors, considérons le cycle formé par le sous-chemin de P_1 entre $c^{j,i}$ et $c^{j,i+1}$, les arêtes $c^{j,i+1}a^{j,i+1}$ et $c^{j,i}d^{j,i}$ et le sous-chemin de P_2 entre $d^{j,i}$ et $a^{j,i+1}$. Par le Lemme 4.2.1, aucun chemin ne peut croiser ce cycle mais son intérieur contient au moins un sommet de croisement des lignes i et $i+1$, contradiction. □

Lemme 4.6.2. *Le complementaire Q de $\mathcal{P}$ peut être décomposé en cycles et en un (w_i, w'_i)-chemin pour tout $i \in [\![1, p']\!]$.*

Preuve. Il existe exactement np gadgets dans le graphe, parmi lesquels exactement $3n$ sont des LIC. Le nombre de croisement est $4np + 6n$. De plus, il y a $2n$ chemins verticaux, croisant chacun les $2p - p'$ chemins horizontaux. Au plus $2n(p'+3)$ sommets de croisement ne sont pas utiliser par des croisements. Puisque le nombre de ligne d'un tampon est $q = 4(p'+3)n + 2$, il y a au moins deux lignes consécutives dans lesquelles tous les sommets de croisements sont utilisés. Nous pouvons alors appliqué le Lemme 4.6.1 : il n'existe pas de non-chemins traversant un tapon de haut en bas.
En raison de la parité des degrés des sommets, $G \setminus \mathcal{P}$ est un $W \cup W'$-joint, et peut de cce fait se décomposer en cycles et chemins d'extrémités couvrant $W \cup W'$, mais les seuls chemins possibles sont des (w_i, w'_i)-chemins, puisque tous les autres chemins traverseraient au moins un tampon. □

Ceci a pour principale conséquence :

Lemme 4.6.3. *Tout chemin vertical est contenu dans une colonne.*

Preuve. $2p - p'$ chemins horizontaux etp' non-chemins sont routés à travers chaque coupe verticale, qui contiennent exactement $2p$ arêtes chacune. Donc les chemins verticaux ne peuvent utiliser ces arêtes. □

4.7 Preuve de NP-complétude

Théorème 4.7.1. *Le problème de chemins arête-disjoints avec graphe d'offre planaire et seulement deux demandes sur le bord de la face extérieure est NP-complet.*

Preuve. Nous utilisons le graphe construit en Section 4.5, qui est de taille polynomiale. Commençons par montrer le sens facile : s'il existe une assignation validant φ, il existe une solution au problème de chemins arête-disjoints. Soit une telle assignation pour les variables $X_1, \ldots, X_{p'}$.
Nous faisons passer dans chaque ligne deux chemins horizontaux, sauf dans les lignes suivantes :

- Si à X_i est assignée la valeur *vrai*, un seul chemin horizontal traverse la ligne $2(q+1)(i-1)+1$.
- Si à X_i est assignée la valeur *faux*, un seul chemin horizontal traverse la ligne $2(q+1)(i-1)+q+2$.

Les deux chemins verticaux de chaque colonne sont systématiquement décalés sauf dans le cas suivant. Soit i le plus petit indice telle que l'assignation de X_i permet de satisfaire la clause k, et soit j la ligne correspondante ($i = 2(q+1)(j-1)+1$ ou $2(q+1)(j-1)+q+2$). Puisqu'il ne passe qu'un seul chemin horizontal par cette ligne et que le gadget de coordonnée (k, j) est un LIC, le Lemme 4.4.4 garantit qu'il est possible de garder les deux chemins verticaux à cet endroit, c'est ce que nous choisissons de faire, pour chaque clause. Les Lemmes 4.4.1, 4.4.2 et 4.4.3 assurent la posibilité de routage dans chaque gadget. Notons qu'il faut nous assurer que le chemin horizontal passant par $M(k, j)$ ne doit pas entrer par les deux arêtes du bas, pour appliquer le Lemme 4.4.4, mais comme $M(k-1, j)$ et $M(k+1, j)$ (s'ils existent) sont gérées par le Lemme 4.4.2, il est effectivement possible de s'en assurer. Au final, nous avons bien construit une solution puisque les chemins verticaux sont tous gardés exactement une fois.

Nous montrons à présent la réciproque, soit une solution $\mathcal{P}$, décroisée, au problème de chemins arête-disjoints défini par (G_φ, H_φ), prouvons l'existence d'une assignation satisfaisant φ. Par le Lemme 4.6.2, le complémentaire de la solution contient des chemins $Q_1, \ldots, Q_{p'}$. Par conséquent, les coupes verticales de la grille sont serrées, puisqu'elle comprennent $2p$ arêtes par lesquelles passent $2p - p'$ chemins. Donc, les chemins verticaux ne peuvent pas utiliser ces arêtes.

Cependant, les chemins horizontaux peuvent utiliser des arêtes des coupes horizontales, mais de manières limitées. Pour montrer cela, nous prouvons que les non-chemins se décalent d'au plus 2 lignes vers le haut ou vers le bas entre chaque colonne. La démonstration se fait itérativement sur chacune des colonnes depuis celle de gauche. Ainsi, nous savons que les non-chemins entre par la gauche dans la colonne k espacés entre eux par $q - 2 > 3$ lignes au minimum. Puisque toutes les autres arêtes des coupes verticales sont utilisés, mais aussi que les chemins horizontaux ne se croisent pas, cela revient juste à montrer que le problème défini dans le Lemme 4.4.5 est sans solution, ce qui est effectivement l'énoncé de ce lemme.

Puisque les non-chemins se déportent d'au plus deux lignes par colonne, chacun d'eux ne peut approcher à moins de 3 lignes qu'une seule des lignes possédant des gadgets LIC (séparés par les tampons de $q > 2n + 5$ lignes). Puisque le passage d'un non-chemin à plus de 3 lignes de distance d'un gadget LIC ne modifie pas le comportement attendu des chemins verticaux

(à savoir être décalé dans chaque ligne) par le Lemme 4.4.6, les chemins verticaux ne peuvent avoir été décalé que lorsque un non-chemin passe à proximité d'un gadget LIC. Le non-chemin permet donc bien d'assigner à la variable X_i une valeur *vrai* ou *faux*, selon qu'il soit passé à proximité de la ligne $2(q+1)(i-1)+1$ ou $2(q+1)(i-1)+2q+2$ respectivement, et l'assignation produite ainsi valide bien φ, puisque dans chaque colonne les chemins verticaux sont décalés une fois. □

Troisième partie

Tours graphiques et ensembles éclatants

Introduction.

Cette partie est issue d'un travail avec Vincent Jost, sur un problème original mais possèdant de nombreux liens avec des problèmes classiques. Nous cherchons à déterminer quand une condition naturelle suffit pour caractériser la longueur minimale d'un tour d'un graphe, passant par tous les sommets.

L'idée originale s'inspire d'un travail de Fonlupt et Naddef [13], étudiant les graphes pour lesquels la condition de coupe suffit pour caractériser les tours de voyageur de commerce. Un ensemble de sommets est dit *éclatant* si sa cardinalité est strictement inférieure au nombre de composantes du graphe obtenu en retirant ces sommets. Un graphe possédant un tel ensemble ne peut posséder de cycle hamiltonien. Une classe de graphes est dite *cycle-tough* si la réciproque est vrai. Nous cherchons à étendre ces concepts, en proposant de déterminer quand la longueur d'une plus courte marche fermée passant par tous les sommets d'un graphe est caractérisée par les packings d'ensembles éclatants.

Malgré les résultats présentés dans la suite de ce travail, nous avons toujours plus de questions que de réponses. Nous souhaitons que ce sujet puisse encore nourrir de nombreux travaux, et pas seulement les nôtres.

Chapitre 5

Généralités

5.1 Contexte et définitions

Soit G un graphe non-orienté connexe, tous les graphes seront supposés connexes par la suite. Un *tour* de G est une marche fermée de G passant par chaque sommet. On appelle *cycle Hamiltonien* ou *tour Hamiltonien* tout tour de G passant par chaque sommet une et une seule fois. Un graphe possèdant un cycle hamiltonien est dit *Hamiltonien.*

Problème 5.1.1. *Soit G un graphe non-orienté. Décider si G est Hamiltonien.*

Il est bien connu que ce problème est NP-complet en général [28]. Des algorithmes polynomiaux de reconnaissance des graphes Hamiltoniens existent néanmoins pour certaines classes de graphes. L'existence d'un algorithme polynomial implique l'existence d'une bonne caractérisation, donc d'un certificat d'inexistence. Puisque le problème est NP-complet, à moins que NP = coNP il n'existe pas de tel certificat en général, mais on peut espérer en trouver pour des classes de graphes particuliers. Parmi les certificats les plus simples, nous nous proposons d'étudier les ensembles éclatants. Nous notons $c(U)$ pour un ensemble U de sommets d'un graphe G le nombre de composantes connexes de $G - U$

Définition 5.1.2. *Un* ensemble éclatant *(*scattering set*) d'un graphe G est un ensemble $U \subseteq V(G)$ de sommets tels que le nombre de composantes de $G - U$ est strictement supérieur à la cardinalité de U. Autrement dit, $c(U) - |U| > 0$. Pour tout $U \subseteq V$, on appelle* déficience *(deficiency) de U et on note def(U) la valeur* $\max\{0, c(U) - |U|\}$.

Supposons que G possède un ensemble éclatant U. Un cycle Hamiltonien doit passer par chaque composante de $G - U$, et pour passer d'une composante à l'autre, est forcé d'utiliser au moins un sommet de U. Il traverse donc en tout au moins $c(U)$ sommets de U, ce qui contredit le fait que le cycle passe une seule fois par chaque sommet. Nous venons donc de montrer :

Proposition 5.1.3. *Si G est Hamiltonien, G ne possède pas d'ensemble éclatant.*

Notons que la recherche d'ensembles éclatants est un problème aussi difficile que celle d'un tour Hamiltonien, le problème suivant étant NP-complet [32] :

Problème 5.1.4. *Soit G un graphe non-orienté. Décider si G possède un ensemble éclatant.*

Il est intéressant de trouver des classes de graphes tels que la Proposition 5.1.3 admette une réciproque. Celle-ci ne peut être vraie en général (à moins que NP = co-NP), puisque le problème est NP-complet. La Figure 5.1 présente deux graphes non-Hamiltoniens et sans ensemble éclatant. Notons que ceci a amené Chvátal [7] à définir la *solidité* d'un graphe (*toughness*) : pour un graphe G, il s'agit du minimum du rapport $|U|/c(U)$ pris sur tous les sous-ensembles $U \subset V$ non-triviaux. Un graphe est dit *k-solide* (*k-tough*) si ce minimum est supérieur ou égal à k. Il est particulièrement intéresssant de connaître des conditions suffisantes pour que les graphes 1-solides soient Hamiltoniens, les graphes 1-solides étant exactement les graphes sans ensemble éclatant. Chvátal [6] conjecturait en 1973 qu'il existe une constante t telle que tous les graphes t-solides sont Hamiltoniens. La conjecture est toujours ouverte, et des exemples montrent que si t existe, t vaut au moins $\frac{9}{4}$ (voir par exemple [1]).

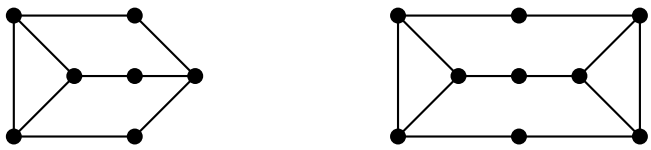


FIGURE 5.1 – *Deux graphes 1-solides mais non-Hamiltoniens.*

Un des résultats les plus marquants dans cette direction est que la réciproque de la Proposition 5.1.3 est vraie pour les graphes de co-comparabilité.

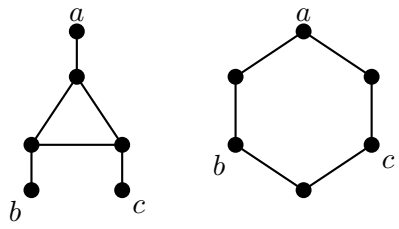


FIGURE 5.2 – *Deux graphes contenant des triplets astéroïdaux a, b, c.*

Les *graphes de comparabilité* sont les graphes obtenus en négligeant l'orientation d'un digraphe transitif (c'est-à-dire du graphe d'ordre d'un ordre partiel). Les *graphes de co-comparabilité* sont leurs complémentaires.

Théorème 5.1.5 (Deogun, Kratsch, Steiner [9]). *Soit G un graphe de co-comparabilité 1-solide, alors G est Hamiltonien.*

Un k-arbre est un graphe obtenu à partir de K_k en appliquant un nombre fini de fois la procédure suivante. On sélectionne une clique C de taille k, et on ajoute un nouveau sommet connecté à chaque sommet de C. Les 1-arbres sont les arbres (obtenus en ajoutant successivement des feuilles).

Théorème 5.1.6 (Broersma, Xiong, Yoshimoto [4]). *Soit G un 2-arbre 1-solide avec au moins 3 sommets. Alors G est Hamiltonien.*

Les mêmes auteurs ont montré que pour $k > 2$, il existe des k-arbres non-Hamiltoniens mais sans ensemble éclatant.

Un triplet astéroïdal est un ensemble indépendant de trois sommets tel que chaque paire est connectée par un chemin évitant le voisinage du troisième sommet. Deux exemples sont donnés en Figure 5.2. Les graphes de la Figure 5.1 possèdent tous deux des triplets astéroïdaux.

La classe des graphes sans triplet astéroïdal contient strictement celle des graphes de co-comparabilité.

Conjecture 5.1.7 (Jost, 2008). *Tout graphe sans triplet astéroïdal 1-solide est Hamiltonien.*

La version dite graphique de problème du voyageur de commerce consiste à trouver un plus petit tour dans un graphe quelconque. Les arêtes possèdent des longueurs données par une fonction $l : E(G) \to \mathbb{R}^+$. Une solution peut passer plusieurs fois par le même sommet, contrairement au Problème 5.1.1. Nous appelons *tour graphique* un tel tour. De nouveau se pose la question

de savoir si les ensembles éclatants peuvent certifier l'optimalité d'un tour. Pour tout ensemble éclatant U, nous savons que n'importe quel tour doit passer au minimum $c(U)$ fois par U. On pose donc le programme linéaire en nombres entiers suivant :

$$\begin{aligned} \min \quad & x(V) \qquad \text{s.t.} \\ & x(U) \geq \mathrm{c}(U) \qquad (U \subseteq V) \end{aligned} \tag{P'}$$

Soit $x_v \in \mathbb{N}$ le nombre de passage par v d'un tour de G, nous appelons x le *vecteur de transit* de ce tour. Alors $x \in \mathbb{N}^{V(G)}$ appartient au polyèdre défini dans (P'), et la fonction objectif atteint pour x la longueur du tour, donc l'optimum est nécessairement inférieur. (P') fournit donc une borne inférieure sur la longueur du plus court tour. Il n'y a pas nécessairement égalité, par exemple les graphes présentés en Figure 5.1 ne sont pas Hamiltoniens, et acceptent pourtant le vecteur constamment égal à 1 comme solution de (P') (puisqu'ils n'ont pas d'ensembles éclatants).

La version duale de (P') s'écrit :

$$\begin{aligned} \max \quad & \sum_{U \subseteq V} y_U \mathrm{c}(U) \qquad \text{s.t.} \\ & \sum_{U \ni v} y_U = 1 \qquad (v \in V) \\ & y \geq 0 \end{aligned} \tag{D'}$$

Il s'agit d'une partition maximisant le nombre de composantes créées, ce qui revient à maximiser la somme des déficiences, à condition que $c(U) > |U|$ pour toute variable y_U active :

$$\begin{aligned} \sum_{U \subseteq V} y_U \mathrm{c}(U) &= \sum_{U \subseteq V} y_U \mathrm{def}(U) + \sum_{U \subseteq V} y_U . |U| \\ &= \sum_{U \subseteq V} y_U \mathrm{def}(U) + \sum_{v \in V} \sum_{U \subseteq V, U \ni v} y_U \\ &= \sum_{U \subseteq V} y_U \mathrm{def}(U) + |V| \end{aligned}$$

Sans perte de généralité, une solution optimale consistera donc seulement en des ensembles de déficience strictement positive et des singletons, puisque un ensemble de déficience négative (ou nulle) sera remplacé par les singletons correspondants, qui ont une déficience positive ou nulle. Plus généralement :

Lemme 5.1.8. *Il existe une solution optimale du programme* (D') *telle que pour toute variable active y_U, il n'existe pas de partition propre $U_1, \ldots, U_k$ de U telle que :*

$$def(U) \leq \sum_{i=1}^{k} def(U_i)$$

□

Nous en déduisons immédiatement :

Corollaire 5.1.9. *Dans une telle solution, si y_U est une variable active avec $|U| > 1$, tout sommet $u \in U$ est adjacent à au moins trois sommets dans trois composantes connexes distinctes de $G - U$.* □

Appelons $\lambda(G)$ la longueur minimale d'un tour graphique, $\sigma(G)$ l'optimum entier de (P') et $\pi(G)$ l'optimum entier de (D'), $\sigma^*(G)$ et $\pi^*(G)$ les optimums fractionnaires de ces deux programmes. Les inégalités suivantes sont alors vraies pour tout graphe G :

$$\lambda(G) \geq \sigma(G) \geq \sigma^*(G) = \pi^*(G) \geq \pi(G) \tag{5.1}$$

Dans notre encodage en programme linéaire, nous avons choisi de prendre pour variables les sommets du graphe. Usuellement, pour un problème de tour de voyageur de commerce, c'est un encodage sur les arêtes du graphe qui est utilisé, puisque ce sont les arêtes qui déterminent le tour. En choisissant les sommets, nous perdons une information importante : une solution optimale de (P') est insuffisante pour construire un tour optimal, même lorsque $\lambda(G) = \sigma(G)$. Nous obtenons seulement la séquence des degrés d'un tour optimal lorsque la précédente égalité est vraie. Le problème de déterminer une solution optimale entière de (P') est un problème moins difficile (ou autant) que de chercher directement le tour.

Nous pouvons ensuite ajouter des poids, qui ne seront pas disposés sur les arêtes comme dans le cas classique, mais sur les sommets, $w : V(G) \to \mathbb{N}$. Le problème de tour graphique associé est donc un tour avec coût de passage sur les sommets. Nous définissons donc les deux programmes linéaires en nombres entiers suivant :

$$\begin{aligned} \min \quad & wx \qquad \text{s.t.} \\ & x(U) \geq \mathrm{c}(U) \qquad (U \subseteq V) \end{aligned} \tag{P}$$

et son dual :

$$\begin{aligned} \max \quad & \sum_{U \subseteq V} y_U \mathrm{c}(U) \qquad \text{s.t.} \\ & \sum_{U \ni v} y_U = w_v \qquad (v \in V) \\ & y \geq 0 \end{aligned} \tag{D}$$

Nous notons $\lambda_w(G)$ la longueur du plus court tour graphique, la longueur d'un tour étant définie par : $\sum_{v \in V} w_v s(v)$ où $s(v) \geq 1$ désigne le nombre de fois où le tour passe par le sommet v. $\sigma_w(v)$ dénote l'optimum entier de (P), et $\pi_w(G)$ celui de (D). Avec ces notations :

$$\lambda_w(G) \geq \sigma_w(G) \geq \pi_w(G) \tag{5.2}$$

5.2 Étude préliminaire

Le système d'inégalités définissant le programme (P) n'est pas entier, comme le démontre l'exemple de la Figure 5.3. Les optimums du primal et du dual y ont pour valeur $\frac{21}{2}$. La solution primale consiste à prendre $x_u = \frac{3}{2}$ pour les sommets extérieurs, $x_u = 1$ pour le sommet du centre. La solution duale utilise les ensembles éclatants construits en prenant deux sommets non-consécutifs du cycle extérieur et le sommet central, ainsi pour un tel ensemble U, $c(U) = 4$. Elle s'obtient en prenant pour chacun de ces ensembles U, $y_U = \frac{1}{2}$, et en complétant en prenant le singleton contenant le sommet central. L'exemple se généralise en prenant un cycle extérieur de longueur $2n + 1$, donnant des solutions de la forme $p - \frac{1}{n}$ pour un entier p, montrant qu'il n'existe pas nécessairement de solution $\frac{1}{k}$-entière, quel que soit k.

La fonction c telle que nous l'avons définie n'est pas une fonction inconnue. Par exemple [17] utilise l'inégalité suivante (rappellons que $d(A, B)$ est le nombre d'arêtes entre $A \setminus B$ et $B \setminus A$) :

Lemme 5.2.1. *Pour tout $A, B \subseteq V$:*

$$c(A) + c(B) \leq c(A \cup B) + c(A \cap B) + d(A, B)$$

Nous en déduisons le théorème suivant :

Théorème 5.2.2. *Pour tout graphe connexe ne contenant pas de chaise induite (*cf. *Figure 5.4), le système d'inégalités de* (P) *est LSTU.*

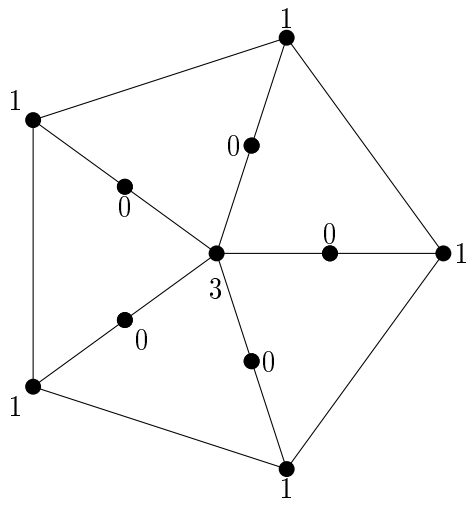


FIGURE 5.3 – *Notre système n'est pas entier sur ce graphe. Le poids d'un sommet est indiqué lorsqu'il n'est pas nul.*

Preuve. Soient $G = (V, E)$ un tel graphe, w une fonction de poids entière, et y un vecteur solution du dual (P). Soit $\mathcal{F}$ la famille des ensembles définissant les variables actives du dual (*i.e.* les U tels que $y_U \neq 0$). Supposons que $\mathcal{F}$ ne soit pas laminaire. Par définition, il existe deux ensembles A et B croisés dans $\mathcal{F}$, $A \cap B, A \setminus B, B \setminus A \neq \emptyset$. D'après le Lemme 5.2.1, s'il n'existe pas d'arête entre $A \setminus B$ et $B \setminus A$, A et B se décroisent en $A \cap B$ et $A \cup B$. Nous avons donc $d(A, B) > 0$.
Pour tout $v \in A \setminus B$, v est adjacent à trois composantes distinctes de $G - A$ par le Corollaire 5.1.9 (symétriquement pour $v \in B \setminus A$). Distinguons les deux cas suivants :

- Cas 1.
  S'il existe $a \in A \setminus B$, $b \in B \setminus A$ et $u \notin A \cup B$ tel que $au, ab \in E$ et $ub \notin E$, alors b est adjacent à au moins trois composantes de $G - B$, l'une d'elles contient a et u, les deux autres fournissent deux sommets v et w adjacents à b, mais ni à a ni à u, alors l'ensemble $\{a, b, u, v, w\}$ induit une chaise.
- Cas 2.
  S'il existe $a \in A \setminus B$, $b \in B \setminus A$ et $u \notin A \cup B$ tel que $au, ab \in E$ et $ub \in E$, alors a est adjacent à au moins trois composantes de $G - A$, l'une contenant u et b. Si l'une d'elles contient un sommet b' dans $B \setminus A$,

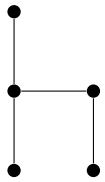


FIGURE 5.4 – *La chaise.*

a, b', u nous replace dans le cas 1. Sinon, n'importe laquelle des deux composantes fournit un sommet u' hors de $A \cup B$ qui n'est pas adjacent à b, et donc a, u', b nous ramène au cas 1 aussi.

Tout ce qu'il reste à montrer, c'est donc qu'il existe $a \in A \setminus B$, $b \in B \setminus A$ et $u \notin A \cup B$ avec ua, $ab \in E$ (ou symétriquement en inversant A et B). Supposons que ce ne soit pas le cas. Nommons A' les sommets de $A \setminus B$ adjacents à $B \setminus A$, et symétriquement B', alors A' et B' sont non-vides puisque $d(A, B) > 0$. Alors, le nombre de composantes de $G - A$ adjacentes à A' est au moins $|A'| + 2$ par le Lemme 5.1.8 (car A' est strictement inclu dans A puisque $A \cap B \neq \emptyset$), et au plus $|B'|$ (puisque A' n'est adjacent qu'aux sommets de B en dehors de A), donc $|A'| + 2 \leq |B'|$. De même par symétrie, $|B'| + 2 \leq |A'|$. Contradiction.
Donc $\mathcal{F}$ peut-être choisie laminaire, le système est LSTU. □

Comme la chaise contient une griffe (le biparti complet $K_{1,3}$) et un P_4, nous avons immédiatement le corollaire suivant :

Corollaire 5.2.3. *Pour les graphes sans griffes et pour les graphes sans P_4, le système est TLSU.*

5.3 Mineurs de sommets

Une direction possible pour améliorer notre compréhension du problème consiste à tenter de caractériser par mineurs exclus les graphes pour lesquels l'optimum de (D) est entier et correspond à la longueur du plus court tour graphique. Nous définissons deux opérations de mineurs.
Nous disons qu'un graphe H est obtenu par *délétion d'un sommet* ou *suppression de sommet* de G s'il existe $u \in V(G)$ tel que $V(H) = V(G) \setminus \{u\}$ et $E(H) = E(G) \setminus \delta(u)$. Nous notons $H = G \setminus u$ ou $H = G - u$. H est donc un sous-graphe induit de G s'il s'obtient par une séquence de suppressions de sommets depuis G.
H est obtenu par *contraction d'un sommet* de G s'il existe $u \in V(G)$ tel

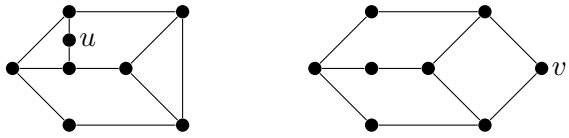


FIGURE 5.5 – *Les notions de mineurs de sommets sont insuffisantes pour notre problème. Sur le graphe de gauche, le graphe est hamiltonien, donc possède les bonnes propriétés, mais la suppression de u le transforme en graphe non-hamiltonien, sans ensemble éclatant. Le graphe de droite possède un ensemble éclatant, tel que dupliquer n'importe quel sommet de cet ensemble rend le graphe hamiltonien. Pourtant, la contraction de v donne le même mauvais graphe.*

que $V(H) = V(G) \setminus \{u\}$ et $E(H) = (E(G) \setminus \delta(u)) \cup \{vw \ : \ vw \in N_G(u)\}$, autrement dit en remplaçant un sommet de G par une clique sur son voisinage. Nous notons $H = G/u$. H est un *sous-graphe contracté* de G s'il s'obtient depuis G par un séquence de contractions de sommets. H est un *mineur par sommet* de G s'il s'obtient par un séquence de délétions et de contractions de sommets de G.

Lemme 5.3.1. *Les opérations de suppressions et contractions de sommets commutent.*

Preuve. Il suffit de montrer (par récurrence sur le nombre d'opérations appliquées) que la relation d'adjacence du graphe obtenu en contractant S et supprimant T est : u et v sont adjacents s'il existe un chemin entre u et v dont tous les sommets internes sont dans S. □

Idéalement, nous voudrions montrer que la classe des graphes pour lesquels la longueur du plus court tour est égale à l'optimum de (D) pour toute fonction de poids, est fermée par mineurs de sommets. Malheureusement, cela n'est pas vrai (voir Figure 5.5). Nous pouvons montrer la fermeture par mineurs de sommets de la bonne classe de graphes pour une généralisation "façon Steiner" de notre problème. Nous définissons donc le problème suivant :

Problème 5.3.2. *Soit G un graphe non-orienté connexe, $w : V(G) \to \mathbb{R}^+$ et $\widetilde{V} \subset V(G)$ un ensemble de sommets* obligatoires. *Trouver la longueur minimum d'un tour de G passant par tous les sommets de $\widetilde{V}$.*

Nous définissons $V \setminus \widetilde{V}$ l'ensemble des sommets *optionnels*. Soit $U \subset V$ un sous-ensemble quelconque de sommets. Nous notons $\widetilde{c}(U)$ le nombre de

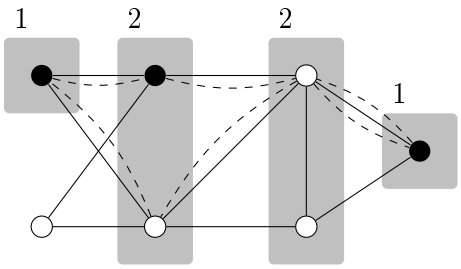


FIGURE 5.6 – *Un graphe avec ensemble de sommets obligatoires en noir, optionnels en blanc. La longueur d'un plus court tour passant par chaque sommet obligatoire est ici bien caractérisée par la partition donnée, créant 6 composantes avec au moins un sommet obligatoire.*

composantes connexes de $G - U$ intersectant $\widetilde{V}$ si ce nombre est au moins 2, sinon $\widetilde{c}(u) = 1$ si $u \in \widetilde{V}$, et 0 dans les autres cas. $\widetilde{c}(U)$ donne donc une borne inférieure simple sur le nombre de passages de toute marche fermée solution du Problème 5.3.2 (définir $\widetilde{c}$ en terme de composantes intersectant $\widetilde{V}$ n'aurait pas suffi, puisque nous pouvons ne pas passer par un ensemble U sans obligatoire, ayant une seule composante contenant des obligatoires). Similairement à (P), nous pouvons définir le programme suivant, donnant une borne inférieure à toute solution du problème :

$$\begin{aligned} \min \quad & wx \qquad \text{s.t.} \\ & x(U) \geq \widetilde{c}(U) \qquad (U \subseteq V) \end{aligned} \tag{PS}$$

et son dual :

$$\begin{aligned} \max \quad & \sum_{U \subseteq V} y_U \widetilde{c}(U) \qquad \text{s.t.} \\ & \sum_{U \ni v} y_U = w_v \qquad (x \in V) \\ & y \geq 0 \end{aligned} \tag{DS}$$

(5.3)

Pour bien comprendre les propositions suivantes, rappelons que nous travaillons sur trois objets : le tour graphique, la solution primale (dont nous espérons qu'elle corresponde au vecteur de transit d'un tour optimal) et la solution duale, qui est un empilement d'ensembles éclatants. Prouver que le système sous-jacent à (P) est TDI ne résout pas notre problème, puisque

cela n'implique rien sur les tours optimaux. Sur n'importe quel graphe non-hamiltonien et sans ensemble éclatant, le système est trivialement entier. Nous devons donc distinguer la classe $\mathcal{C}_1$ des graphes pour lesquels notre système d'inéquations est entier, de la classe plus petite $\mathcal{C}_2$ de ceux dont le tour optimal est égal à l'empilement maximum d'ensembles éclatants (pour toute fonction de poids entière et tout ensemble de sommets obligatoires dans les deux cas).

Proposition 5.3.3. *La classe $\mathcal{C}_1$ des graphes pour lesquels le système induit par* (PS) *est TDI pour tout ensemble $\widetilde{V}$ de sommets obligatoires est fermée par mineurs de sommet.*

Preuve. Soit G un graphe connexe avec (PS) TDI pour tout $\widetilde{V}$ obligatoire, et soit $H = G \setminus v$ connexe. Soit $\widetilde{V}$ un sous-ensemble de sommets de H et $w : V(H) \to \mathbb{N}^+$. Nous étendons w à G en définissant w_G avec $w_G(u) = w(u)$ si $u \neq v$ et $w_G(v) = +\infty$. Puisque (PS) est TDI pour $(G, \widetilde{V}, w_G)$, il existe une solution duale entière y. Posons $y'_{V'} = y_{V'} + y_{V' \cup \{v\}}$ pour tout $V' \in V(H)$, y' est trivialement une solution entière pour le dual (DS) pour H, et est de même valeur que y. Soit x une solution de (PS) pour $(G, \widetilde{V}, w_G)$, puisque le sommet v est optionnel et de poids infini, et que $G \setminus \{v\}$ est connexe, $x_v = 0$, et $x_{V(H)}$ est donc une solution pour le primal de H, de même valeur que y', donc y' est optimal.
Soit $H = G/v$. Soit à nouveau $\widetilde{V}$ un sous-ensemble de sommets de H et $w : V(H) \to \mathbb{N}^+$. Nous posons $w_G(u) = w(u)$ si $u \neq v$, et $w_G(v) = 0$. Il existe une solution duale entière y pour $(G, \widetilde{V}, w_G)$, qui est aussi réalisable et de même valeur pour $(H, \widetilde{V}, w)$ en ignorant les ensembles contenant v, puisque $w_G(v) = 0$, et que la contraction de v ne peut pas créer de nouvelles composantes connexes, donc modifier $\widetilde{c}$. Soit x une solution optimale du primal pour $(G, \widetilde{V}, w_G)$. $\widetilde{c}$ n'étant pas modifiée, $x_{V(H)}$ est aussi une solution pour (PS) dans $(H, \widetilde{V}, w)$, de même valeur que x puisque $w_v = 0$, ce qui prouve l'optimalité de y dans $(H, \widetilde{V}, w)$. □

Avec une preuve similaire, nous obtenons aussi :

Proposition 5.3.4. *La classe $\mathcal{C}_2$ des graphes pour lesquels l'optimum entier de* (DS) *est égal à la longueur du plus court tour pour tout ensemble $\widetilde{V}$ de sommets obligatoires et toute fonction de poids entière w, est fermée par mineurs de sommet.*

Voici une des raisons qui nous pousse à étudier la classe des graphes AT-free et certaines de ses sous-classes dans le cadre de notre étude :

Proposition 5.3.5. *Les classes de graphes suivantes sont fermées par mineurs de sommet :*

(i) les graphes AT-free,
(ii) les graphes de co-comparabilité,
(iii) les graphes d'intervalles.

Preuve. La fermeture par induction est évidente dans tous les cas.

(i) Soit H un graphe obtenu par contraction du sommet v d'un graphe AT-free G. Soient a, b, c trois sommets indépendants de H. Puisque (a, b, c) n'est pas un triplet astéroidal de G, sans perte de généralité il n'existe pas de (b, c)-chemin dans $G \setminus (N_G(a) \cup \{a\})$, donc tout (b, c)-chemin P de H doit passer par une arête e créée par la contraction de v, ou par $N_H(a)$. Si $e \in P$, le chemin obtenu en remplaçant $e = u_1u_2$ par u_1v, vu_2 dans P, est un (b, c)-chemins de G, donc passe par $N(a) \cup \{a\}$. Alors, soit $v \in N_G(a)$, dans ce cas $u_1, u_2 \in N_H(a)$, sinon il existe un sommet de P dans $N_G(a)$ donc $N_H(a)$. Donc il n'existe pas de (b, c)-chemins dans $H \setminus N_H(a)$.

(ii) Soit $\geq$ un ordre partiel sur un ensemble V et $u \in V$. Contracter u dans le graphe de co-comparabilité induit par $\leq$ revient à supprimer u et rendre tous les éléments incomparables avec u incomparables entre eux. Il suffit donc de montrer que la relation $\preccurlyeq$ est un ordre, avec :

$$v \preccurlyeq w \Leftrightarrow v = w \vee (v < w \wedge \ v \text{ ou } w \text{ est comparable à } u.) \qquad (5.4)$$

Cette relation est d'évidence réflexive. Soit v et w tel que $v \preccurlyeq w$ et $w \preccurlyeq v$. Alors soit $v \leq w$ et $w \leq v$, soit $v = w$. Dans les deux cas, $v = w$, donc elle est antisymétrique.
Enfin pour la transitivité, soit $v \preccurlyeq w$ et $w \preccurlyeq x$, v, w et x distincts. Alors soit $w \leq u$, donc $v \leq w$ et $w \leq u$ implique par transitivité de $\leq$ que $v \leq u$ et $v \leq x$. Donc $v \preccurlyeq x$. Soit symétriquement $u \leq w$ et donc $v \preccurlyeq x$. Soit enfin u et w sont incomparables et donc u est comparable à v et x, et $v \leq w \leq x$ donc $v \preccurlyeq x$ par transitivité de $\leq$.

(iii) Dans un modèle d'intervalle, contracter un intervalle I revient à remplacer chacun des intervalles adjacents par son union avec I (ou de manière équivalente, identifier les éléments de I), les unions d'intervalles non-disjoints étant eux-mêmes des intervalles. □

Pour finir cette section, nous donnons en Figure 5.7 une liste de graphes minimaux par mineurs de sommets qui sont 1-tough mais pas Hamiltoniens. Cette liste n'est probablement pas exhaustive. Nous l'avons générée par ordinateur, en testant tous les graphes de 10 sommets ou moins.

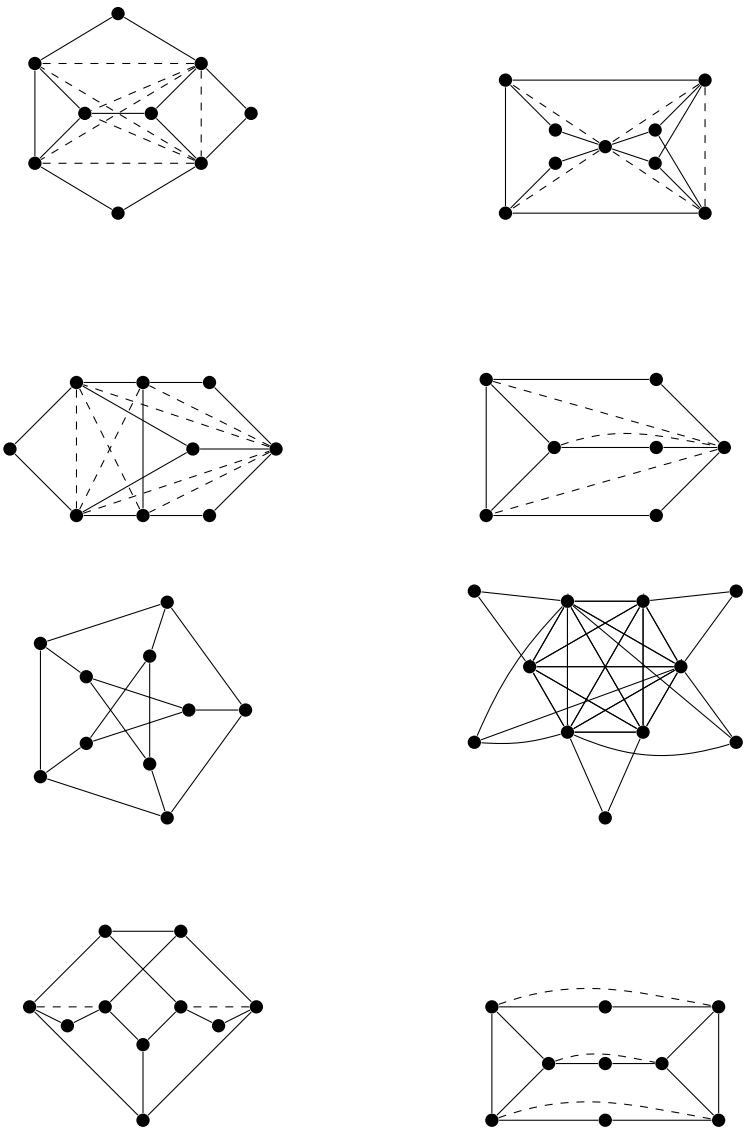


FIGURE 5.7 – *Liste de graphes minimaux par mineurs de sommets, pour lesquels la longueur d'un plus court tour n'est pas donné par une partition en ensemble éclatant. Les arêtes en pointillées sont optionnelles.*

5.4 Séparation

Réécrivons notre système (P) :

$$\begin{aligned} \min \quad & wx \qquad \text{s.t.} \\ & x(U) \geq \mathrm{c}(U) \qquad (U \subseteq V) \end{aligned} \tag{P}$$

Pour pouvoir trouver un optimum de la relaxation de ce programme linéaire (ou du programme entier si le polyèdre sous-jacent est entier), il serait suffisant de savoir séparer ses inégalités (puis d'appliquer la méthode des ellipsoïdes). C'est-à-dire, étant donné un vecteur x, décider s'il existe $U \subseteq V$ tel que $x(U) < \mathrm{c}(U)$. Pour cela, il suffit de trouver une solution au problème suivant :

$$\max_{U} \quad \mathrm{c}(U) - x(U) \tag{5.5}$$

avec le maximum pris sur U décrivant les sous-ensembles de $V(G)$. Plus généralement, intéressons-nous au problème suivant :

$$\max_{S,U} \quad y(S) - x(U) \tag{5.6}$$

avec S décrivant les ensembles indépendants de G et U bloqueur des S-chemins (chemins dont les deux extrémités sont des sommets distincts de S). En prenant pour y le vecteur constamment égal à 1, une solution du problème (5.6) implique une solution de (5.5) : en effet, S peut être vu comme un échantillon de sommets, au plus un par composante de $G - U$. Puisque nous cherchons un maximum, il y en aura bien exactement un par composante, donc $|S| = c(U)$. Enfin, en prenant $y(v) = 1$ si v est obligatoire, $y(v) = 0$ sinon, nous pouvons séparer le problème de Steiner (PS).
Pourquoi étudier ainsi un problème plus général que ce que donne directement la séparation ? En fait, (5.6) généralise plusieurs problèmes classiques et difficiles. En fixant $x = 0$, c'est exactement le problème de trouver un stable maximum dans un graphe. Si S est un stable de G, prendre pour fonction y définie par $y(s) = \infty$ pour $s \in S$, $y(v) = 0$ si $v \notin S$, le problème revient à minimiser $x(U)$, pour U un ensemble de sommets bloquant des S-chemins, problème connu sous le nom de *coupe multi-terminale* (*multiway cut*). En choisissant pour $y = k$ et $x = 1$, déterminer si le maximum est strictement positif revient à déterminer si le graphe est k-solide, car dans ce cas $k.c(U) > |U|$ pour le maximum U, et réciproquement. En particulier, pour $k = 1$ il s'agit du problème de trouver un ensemble éclatant.
Il se trouve que le problème du stable est polynomial sur les graphes sans triplet astéroïdal [5], et que nous allons montrer comment résoudre la coupe

multiterminale sur cette même classe de graphes. Nous conjecturons qu'il est en fait possible de résoudre sur les graphes sans AT, non seulement le problème de la solidité, mais aussi le problème plus général que nous venons de définir : (5.6). Nous en donnerons une solution dans le cas particulier des graphes d'intervalles.

Soit $G = (V, E)$ un graphe, et $S \subset V$ un sous-ensemble indépendant de sommets de G. Considérons le système suivant :

$$x(P) \geq 1 \quad \text{pour tout } S\text{-chemin } P \tag{5.7}$$

Un vecteur entier de ce système minimisant un objectif wx avec w positif est donc un vecteur 0-1, caractéristique d'un bloqueur des S-chemins. Un vecteur entier du dual de (5.7) pour une fonction de poids positive est un vecteur caractéristique d'un (multi-)ensemble de S-chemins sommets disjoints. Nous sommes intéressés par décider quand les optimums sont entiers, donc lorsque le nombre maximum de S-chemins sous capacité w est égal au poids minimum d'un bloqueur des S-chemins. Une condition suffisante est de montrer que le système (5.7) est TDI.
Rappelons que trouver un packing maximum de S-chemin dans un graphe quelconque n'est pas un problème difficile. Mader [38] l'a résolu en donnant la caractérisation suivante. Notons $B(U) = N(U) \setminus U$.

Théorème 5.4.1 (Mader, 1978). *Soit $G = (V, E)$ un graphe et soit $S \subseteq V$ un stable de G. Le maximum de S-chemins internement sommet-disjoints est égal au minimum de*

$$|U_0| + \sum_{i=1}^{n} \left\lfloor \frac{|B(U_i) \setminus U_0|}{2} \right\rfloor$$

pris sur toutes les partitions $U_0, \ldots, U_n$ de $V \setminus S$ telles que tout S-chemin intersecte U_0 ou $E(U_i)$ pour un certain i.

Notre problème revient donc à tenter de trouver une condition faisant seulement intervenir U_0 dans le cas de graphes particuliers. Notons que nous pouvons trivialement supposer que les sommets de S n'ont pas de voisins communs, puisqu'un tel sommet serait forcément pris une fois dans le bloqueur, et routerait seulement des chemins de longueur 2 sans perte de généralité. De même, pour tout $s \in S$, $G \setminus (\{s\} \cup N(s))$ n'a pas de composante connexe disjointe de S, puisque nous pouvons supposer que tout chemin passe au plus une fois par $N(s)$, et pour la même raison, une solution duale peut

attribuer une valeur nulle à chaque sommet de cette composante. Enfin, par un raisonnement analogue, nous pouvons ignorer les arêtes de $E(N(s))$ pour tout $s \in S$.
Pour tout $G = (V, E)$ et S stable de G, nous définissons un graphe auxiliaire G_S, par délétion de tous les sommets communs aux voisinages de deux sommets de S, puis en contractant tous les sommets de $V \setminus (S \cup N(S))$, et enfin en supprimant tous les sommets de S et toutes les arêtes appartenant au voisinage d'un sommet de S. Il reste donc les voisins des sommets de S.
Le théorème suivant est une généralisation du théorème de Menger (pour lequel $|S| = 2$) :

Théorème 5.4.2. *Soit G un graphe et $S \subset V$ un ensemble indépendant de G. Alors le système* (5.7) *est TDI ssi le graphe G_S est biparti.*

Preuve. Si G_S n'est pas biparti, soit C un cycle impair dans G_S, et $w \in \{0, 1, +\infty\}^{V(G)}$ la fonction définie par :

$$w(u) = \begin{cases} 1 & \text{si } u \in V(C) \\ 0 & \text{si } u \in N(S) \setminus V(C) \\ +\infty & \text{sinon.} \end{cases} \tag{5.8}$$

Alors, pour toute arête $e \in E(C)$ il existe un chemin de G de même extrémité, n'intersectant $N(S)$ qu'en ses extrémités. Un packing fractionnaire maximum de chemins peut donc être pris en prenant un tel chemin pour chaque arête $e \in E(C)$, et en l'étendant vers les sommets de S adjacents à ces extrémités. Nous obtenons ainsi $|E(C)|$ chemins, que nous prenons chacun avec un poids $\frac{1}{2}$, ce qui donne une solution valant $\frac{|E(C)|}{2}$.
Puisque tous les S-chemins doivent passer ou bien par un sommet de $N(S) \setminus V(C)$, ou bien par deux sommets de $V(C)$, un bloqueur fractionnaire est donné en prenant $\frac{1}{2}$ sur les sommets de $V(C)$, 0 ailleurs, donnant une solution de valeur $\frac{|E(C)|}{2}$ aussi, prouvant l'optimalité de ces solutions non-entières, donc le système n'est pas TDI.
Supposons maintenant que G_S est biparti, de classes A et B, et soit w une fonction de poids sur les sommets. Considérons une composante connexe C de $G \setminus (S \cup N(S))$. Si cette composante connexe est adjacente aux voisinages de trois sommets distincts de S, alors G_S ne pourrait pas être biparti car il comprendrait un triangle. Donc C est adjacente à deux sommet u et v de S, et sans perte de généralité, $N(C) \cap N(u) \subseteq A$ et $N(C) \cap N(v) = B$. Donc tout S-chemin dans G intersecte nécessairement A et B.
Soit G' le graphe obtenu depuis G en supprimant les sommets de S et les arêtes de $E(N(s))$ pour tout sommet s (puisque sans perte de généralité,

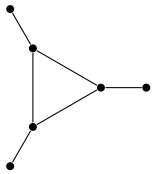


FIGURE 5.8 – *La longue griffe.*

aucun chemin n'est routé par ces arêtes), puis en ajoutant deux nouveaux sommets s et t adjacents respectivement aux sommets de A et B. Alors, en appliquant le théorème de Menger dans G', w entre s et t, nous obtenons un ensemble de chemins disjoints qui se transforme naturellement en packing de S-chemins (car chacun intersecte A et B), et un bloqueur des (s,t)-chemins de même valeur, qui bloque donc aussi les S-chemins. Ces solutions étant entières, et ceci étant vrai pour toute fonction de poids w, le système est TDI. □

Appelons *longue griffe* le graphe à 6 sommets présenté en Figure 5.8. Nous aimerions donner une caractérisation par mineur exclu de la classe $\mathcal{C}$ des graphes pour lesquels le système (5.7) est TDI. Plaçons-nous dans un cas pour lequel ce système n'est pas TDI, c'est-à-dire que le graphe G_S n'est pas biparti. Il contient donc un cycle simple impair C. Considérons le graphe G' obtenu par suppression des sommets ayant au moins deux voisins dans S, par contraction des sommets de $V(G) \setminus (S \cup N(S))$, puis par suppression des sommets de $N(S) \setminus C$. Nous avons donc $V(G') = S \cup V(C)$, et G' contient C. Colorions de couleurs différentes les sommets de S, et attribuons à chaque sommet de C la couleur de son voisin dans S. Tout sommet du cycle est voisin d'exactement un sommet de S. Le cycle est proprement colorié, et ne peut avoir des cordes uniquement entre deux sommets de même couleur.

Ceci nous suggère la définition suivante. Soit $\mathcal{A}_k$ la classe des graphes possédant un stable dominant T (c'est-à-dire $T \cup N(T) = V(G)$), et une coloration des sommets tels que les sommets de T sont de couleurs distinctes, et les arêtes incidentes à deux sommets de couleurs distinctes forment un cycle couvrant de $G \setminus T$. G' appartient donc à $\mathcal{A}_{|C|}$. La longue griffe appartient à $\mathcal{A}_3$.

Théorème 5.4.3. *Soit G un graphe, et T un stable de G. Le système $x(P) \geq 1$ pour tout T-chemin P est TDI ssi G ne contient pas un graphe de $\mathcal{A}_k$ pour k impair par mineur de sommet.*

Preuve. Par la construction de G, nous avons montré une implication. Pour

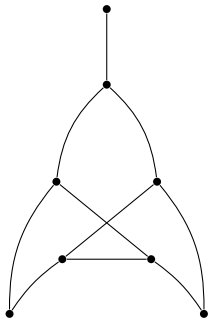


FIGURE 5.9 – *Le graphe appelé* fusée

l'autre, il suffit de montrer que pour tout graphe G de $\mathcal{A}_k$ avec k impair, le système n'est pas TDI. Pour cela, choisissons le vecteur de poids constamment égal à 1 sur les sommets de G. La solution x attribuant $\frac{1}{2}$ à chaque sommet du cycle impair C a une valeur de $\frac{|C|}{2}$. La solution duale consistant à choisir $y_P = \frac{1}{2}$ pour tout T-chemin P de longueur 3 dont l'arête centrale est dans C, a la même valeur, qui n'est pas entière car $|C|$ est impair. □

Nous ne présentons pas les preuves des résultats suivants, qui seront disponibles dans un article prochainement. Nous pouvons aussi montrer que $\mathcal{A}_3$ possède une infinité de graphes minimaux par mineurs de sommets.

Théorème 5.4.4. *Soit G un graphe, et T un stable de G. Le système $x(P) \geq 1$ pour tout T-chemin P est TDI ssi G ne contient pas un graphe de $\mathcal{A}_3$ par mineur de sommet.*

Corollaire 5.4.5. *Pour tout graphe AT-free G et pour tout ensemble indépendant $S \subset V(G)$, le système* (5.7) *est TDI.*

Preuve. Les graphes AT-free sont fermés par contraction d'arête par définition, et par mineurs de sommet par la Proposition 5.3.5, or les sous-graphes minimaux exclus contiennent tous un triplet astéroïdal formé des représentants de chaque couleur. □

Pour conclure cette partie, mentionnons qu'en ajoutant la contraction d'arêtes, il suffit d'exclure deux graphes pour avoir une classe sur laquelle le système (5.7) est TDI. Introduisons le graphe *fusée* décrit en Figure 5.9, qui appartient à $\mathcal{A}_5$.

Théorème 5.4.6. *Soit G un graphe ne contenant pas de longue griffe ni de fusée par mineur de sommet et contraction d'arête, et S un ensemble indépendant de G, alors le système* (5.7) *est TDI.*

Chapitre 6

Tours et Cographes

Un *cographe* G est un graphe qui ne contient pas de P_4 induit. Les cographes admettent une décomposition qui va nous être utile : un cographe avec au moins deux sommets est non-connexe ou bien son complémentaire est non-connexe. De plus, les cographes étant par définition une classe fermée par sous-graphe induit, chacune des composantes connexes (de G ou de son complémentaire) est elle-même un cographe. Nous introduisons alors la notion de *composition de graphes* : soit G un graphe avec $V(G) = \{u_1, \ldots, u_n\}$, et $H_1, \ldots, H_n$ des graphes quelconques. La *composition de G par $H_1, \ldots, H_n$*, notée $G[H_1, \ldots, H_n]$, est le graphe dont l'ensemble des sommets est (u_i, v) tel que $v \in V(H_i)$, avec (u_i, v) adjacent à (u_j, w) si $u_iu_j \in E(G)$ ou $(i = j$ et $vw \in E(H_i))$. Ainsi, les cographes sont les graphes obtenus par la construction suivante :

- Le graphe à un sommet est un cographe.
- Si $H_1, \ldots, H_n$ sont des cographes, $S_n[H_1, \ldots, H_n]$ et $K_n[H_1, \ldots, H_n]$ sont des cographes.

On peut donc associer à tout cographe un *arbre de décomposition*, dont les nœuds internes représentent des cliques ou des stables de taille égale à l'arité du nœud, et les feuilles sont les sommets du graphe. Cette décomposition est connue sous le nom de *décomposition modulaire* de G (cette décomposition se généralise à tout graphe, les cographes ayant la particularité de ne faire appel qu'à des cliques et des stables). À tout nœud interne N est associé l'ensemble des sommets $V(N)$ correspondant aux feuilles dont ce nœud est un ancêtre.

Le Corollaire 5.2.3 montre que notre système d'inéquation est TDI sur les cographes. Nous allons montrer qu'il est possible de résoudre aussi bien

le problème de trouver un tour optimal et une partition en ensembles éclatants de même taille dans le cas pondéré, que de résoudre le problème de séparation (5.6). Dans les deux cas, on utilise un algorithme sur l'arbre de décomposition modulaire du cographe.

6.1 Plus court tour pondéré

Rappelons la condition de Dirac pour l'Hamiltonicité d'un graphe :

Lemme 6.1.1 (Dirac). *Tout graphe à n sommets de degré minimum supérieur ou égal à $\frac{n}{2}$ est Hamiltonien.*

En particulier, il est possible de trouver un cycle Hamiltonien dans ce cas en temps polynomial.

6.1.1 Ensemble éclatant de déficience maximale

Montrons d'abord comment trouver un ensemble éclatant de déficience maximale. Soit G un cographe connexe avec aux moins deux sommets. G se décompose en $K_n[H_1, \ldots, H_n]$, avec $H_1, \ldots, H_n$ cographes soit réduits à un point, soit non-connexes. Sans perte de généralités, on suppose $|V(H_1)| \geq \ldots \geq |V(H_n)|$. Si $V(H_1) \leq \frac{|V(G)|}{2}$, alors tous les sommets ont degré au moins $\frac{|V(G)|}{2}$. Par le Lemme 6.1.1, il existe un chemin Hamiltonien, donc il n'existe pas d'ensemble éclatant dans G.
Sinon, $V(H_1)$ contient plus de la moitié des sommets. Tout ensemble éclatant doit contenir les sommets de $n-1$ composantes du complémentaire de G. Pour être éclatant, un ensemble doit donc contenir $\bigcup_{i=2}^{n} V(H_i)$. Nous décomposons $H_1 = S_p[G_1, \ldots, G_p]$. Soit $S_1, \ldots, S_p$ les ensembles éclatants de $G, \ldots, G_p$ respectivement, de déficience maximale, obtenu inductivement (si G_i n'a pas d'ensemble éclatant, on prend $S_i = \emptyset$).
Soit $S = \bigcup_{i=1}^{p} S_i \cup (V(G) \setminus V(H_1))$. Si S est éclatant, alors c'est un ensemble éclatant de déficience maximale. En effet, s'il n'est pas de déficience maximale, il existe un ensemble S' meilleur, contenant $V(G) \setminus V(H_1)$. S' induit alors sur chacun des S_i un ensemble S'_i. Le nombre de composantes créées par S (resp. S') est égal à la somme des composantes des graphes induits par $G_i - S_i$ (resp. $G_i - S'_i$). Il existe donc une valeur de $i \in [\![1, p]\!]$ tel que la déficience de S'_i soit strictement supérieure à celle de S_i dans G_i, ce qui contredit le choix de S_i. Enfin, si S n'est pas éclatant, il n'existe pas d'ensemble éclatant pour la même raison : un ensemble éclatant induirait un meilleur ensemble éclatant dans l'un des G_i. De plus, si tous les S_i sont minimaux

par inclusion, S aussi est minimal par inclusion (par rapport à sa déficience). Plus généralement :

Lemme 6.1.2. *Les ensembles éclatants du graphe $K_n[H_1, \ldots, H_n]$ avec $H_1 = S_p[G_1, \ldots, G_p]$ et $|V(H_1)| \geq \ldots |V(H_n)|$, sont de la forme :*

$$S = \bigcup_{i=2}^{n} V(H_i) \cup \bigcup_{i=1}^{p} S_i \tag{6.1}$$

où S_i est soit un ensemble vide, soit un ensemble éclatant de G_i. La déficience de S est alors la somme des déficiences des S_i, moins $|\bigcup_{i=2}^{n} V(H_i)|$.

Ce qui permet de décrire tous les ensembles éclatants, en utilisant la décomposition modulaire :

Lemme 6.1.3. *Tout ensemble éclatant de G est l'ensemble des feuilles d'un sous-arbre enraciné T' de l'arbre T de décomposition modulaire, construit selon la règle suivante : si N est un nœud de clique à n fils qui appartient à T', T' contient au moins $n-1$ sous-arbres fils de N entièrement. La déficience d'un tel ensemble est alors le nombre de branches coupées (entre T et T') moins le nombre de feuilles de T'.*

De plus, un ensemble éclatant de déficience maximale peut alors être trouvé en temps polynomial dans tout cographe connexe, minimal par inclusion, s'il en existe un.

Nous montrons que la longueur (en nombre de sommet) d'un tour minimum des sommets de G est égale à la cardinalité de $V(G)$ plus la déficience de cet ensemble éclatant maximal. En effet, par induction, il existe un tour T_i sur chacun des G_i, de longueur $|V(G_i)| + \mathrm{def}_{G_i}(S_i)$. Chaque tour T_i peut être décomposé en $c_{G_i}(S_i) - |S_i|$ chemins sommet-disjoints (en supprimant toutes les occurences multiples d'un sommet sauf une). Nous obtenons donc $\sum_{i=1}^{p} c_{G_i}(S_i) - |V(H_1) \cap S|$ chemins disjoints couvrant $V(H_1)$. Nous utilisons autant de sommets de $V(H_2) \cup \ldots \cup V(H_n)$ pour créer un tour passant par tous les sommets de G, qui aura donc pour longueur :

$$\begin{aligned} V(H_1) + \max\{|V(H_2) \cup \ldots \cup V(H_n)|, \sum_{i=1}^{p} c_{G_i}(S_i) - |V(H_1) \cap S|\} \\ = \max\{|V(G)|, \sum_{i=1}^{p} c_{G_i}(S_i) - |S| + |V(G)|\} \end{aligned}$$

Nous venons donc de démontrer :

Lemme 6.1.4. *Sur les cographes connexes, $\lambda(G) = \pi(G)$. On peut trouver en temps polynomial un tour minimum et une partition maximum.*

6.1.2 Cas général

Soit T l'arbre de décomposition modulaire d'un cographe G. Pour tout nœud de clique A, définissons $\mu(A)$ comme le nombre minimum de chemins sommet-disjoints sur le sous-graphe de G induit par $V(A)$, couvrant tous les sommets, plus un si A est la racine. Pour la racine, cette valeur correspond à un plus la déficience maximum d'un ensemble, pour les autres nœuds, il s'agit de la déficience maximum d'un ensemble sur le sous-graphe induit par $V(A)$ (il est facile de transformer un tour en ensemble de chemins couvrants, en supprimant les occurences multiples des sommets, c'est donc une conséquence du Lemme 6.1.4). Nous appelons *déficience* d'un graphe la somme maximum des déficiences des parties d'une partition des sommets de ce graphe.
Informellement et sans démonstration, un sommet descendant de A ne peut pas être utilisé plus de $\mu(A) - 1$ fois par un tour minimum, car $\mu(A)$ suffit pour obtenir un chemin parcourant tous les sommets de A. Ainsi, pour tout sommet v , la valeur minimale de $\mu(A)$ pour A contenant v borne le nombre de passages d'un tour minimum par ce sommet. Nous allons montrer que l'algorithme consistant à répliquer le sommet de poids minimum de l'ensemble éclatant de déficience maximum, autant de fois que cette valeur, permet de trouver le tour minimum (et un packing d'ensembles éclatants maximum).
Plus rigoureusement, notons la relation de récurrence suivante : si A est un nœud de clique, $B_1, \ldots, B_n$ les racines de ses fils avec $|V(B_1)| \geq \ldots \geq |V(B_n)|$, et $C_1, \ldots, C_p$ les racines des fils de B_1, nous avons, si R est la racine :

$$\mu(\text{feuille}) = 1 \tag{6.2}$$

$$\mu(A) = \max\{1, (\sum_{i=1}^{p} \mu(C_i) - \sum_{i=2}^{n} |V(B_i)|\} \qquad (A \neq R) \tag{6.3}$$

$$\mu(R) = \max\{1, (\sum_{i=1}^{p} \mu(C_i) - \sum_{i=2}^{n} |V(B_i)| + 1\} \tag{6.4}$$

Soit S l'ensemble de déficience maximum, minimal par inclusion. Soit u le sommet de plus petit poids dans S. Soit A le nœud de T, ancêtre de u, minimisant $\mu(A)$ (en cas d'égalité, on prend A au plus près de la racine). Définissons G' le graphe obtenu en répliquant u en $\mu(A)$ copies adjacentes

(on remplace la feuille u par une clique C_u de taille $\mu(A)$ dans l'arbre de décomposition modulaire), toutes de poids $w(u)$. Notons que par minimalité de S, $\mu(A) \geq 2$, car sinon $S \setminus V(A)$ a même déficience que S. Soit T' l'arbre de décomposition modulaire de G'. Soit A' le nœud de T' correspondant à A ($V(A) \setminus \{u\} = V(A') \setminus V(C_u)$), $\mu(A') = 1$, donc A' n'est pas contenu dans l'ensemble éclatant de déficience maximal de G'. De même, les nœuds ancêtres de A ont tous perdu $\mu(A) - 1$ en valeur pour μ (par les équations (6.3) et (6.4)).
Démontrons par induction sur la déficience maximale de G que la solution duale est de la forme suivante : il existe $S_1 \supset \ldots \supset S_k$ tel que S_1 est l'ensemble de déficience maximal et :

$$\begin{aligned} y_{S_1} &= \min_{v \in S_1} w(v) \\ y_{S_i} &= \min_{v \in S_i} w(v) - \min_{v \in S_{i-1}} w(v) \\ y_{\{v\}} &= w(v) - \sum_{S_j \ni v} y_{S_j} \\ y_U &= 0 \quad \text{sinon.} \end{aligned}$$

Si A est la racine de T, alors $k = 1$ et $S_1 = S$ convient. En effet, il existe un tour de poids $w(V) + \text{def}(S)w(u) = w(V) + (\mu(A) - 1)w(u)$, correspondant au tour Hamiltonien obtenu sur G' (puisque G' n'a pas d'ensemble éclatant). La solution duale a pour valeur :

$$\begin{aligned} w(u).c(S) + \sum_{v \in V} w(v) - \sum_{v \in S} w(u) &= w(V(G)) + w(u)(c(S) - |S|) \\ &= w(V(G)) + w(u)\text{def}(S) \end{aligned}$$

Il y a donc égalité, ce qui prouve l'optimalité des deux solutions.
Si A n'est pas racine de T, alors soit $S_1, \ldots, S_k$ définissant une solution y' pour G' (par hypothèse d'induction, car G' a une déficience inférieur de $\mu(A) - 1$ à celle de G). $\sum_{i=1}^{k} y'_{S_i}.c_{G'}(S_i) + \sum_{u \in V(G')} y'_u$ est égal au poids du plus court tour de G', qui est supérieur ou égal au poids d'un plus court tour de G. Notons que $S_1 = S \setminus V(A)$ (A ayant été choisi au plus proche de la

racine, cela suit de l'équation 6.3). Définissons $S_0 = S$ et :

$$\begin{aligned}
y_{S_0} &= w(u) \\
y_{S_1} &= \min_{v \in S_1} w(v) - y_{S_0} = y'_{S_1} - w(u) \\
y_{S_i} &= \min_{v \in S_i} w(v) - \min_{v \in S_{i-1}} w(v) = y'_{S_i} \\
y_{\{v\}} &= w(v) - \sum_{S_j \ni v} y_{S_j} \\
y_U &= 0 \quad \text{sinon.}
\end{aligned}$$

Notons que $\mathrm{def}_G(S_i) = \mathrm{def}_{G'}(S_i)$, car $u \notin C_i \subseteq C_1$. De plus, $S_0 \supset S_1 \supset \ldots \supset S_k$ avec S_0 ensemble éclatant de déficience maximum, minimal par inclusion. Il suffit donc de montrer que cette solution y dans G atteint la même valeur que y' dans G'. Appelons Obj la valeur atteinte par y, et Obj$'$ l'optimum pour G' atteint par y'. Soit u' le sommet de poids minimum dans $S_1 \setminus S_0$.

$$\begin{aligned}
\mathrm{Obj} &= \sum_{i=0}^{k} y_{S_i} c_G(S_i) + \sum_{v \in V(G)} y_{\{v\}} && (6.5) \\
&= \sum_{i=2}^{k} y'_{S_i} c_{G'}(S_i) + w(u) c_G(S_0) + (w(u') - w(u)) c_{G'}(S_1) \\
&\quad + \sum_{v \notin S_0 \setminus S_1} y'_v + \sum_{v \in S_0 \setminus S_1} (y'_v - w(u)) && (6.6) \\
&= \sum_{i=2}^{k} y'_{S_i} c_{G'}(S_i) + y'_{S_1} c_{G'}(S_1) + w(u)[c_G(S_0) - c_{G'}(S_1)] \\
&\quad + \sum_{v \in V(G')} y'_v - w(u).|S_0 \setminus S_1| && (6.7) \\
&= \mathrm{Obj}' + w(u)[c_G(S_0) - c_{G'}(S_1) - |S_0 \setminus S_1| - \mu(A)] && (6.8) \\
&= \mathrm{Obj}' && (6.9)
\end{aligned}$$

Le passage de (6.7) à (6.8) tient compte des sommets répliqués de u. Pour obtenir (6.9), on remarque que $c_G(S_0) = c_G(S_1) - 1 + d$ où d est le nombre de composantes créées par le retrait de S_0 dans A, et donc $d = \mu(A) + 1 + |A \cap S_0|$ donc $c_G(S_0) = c_G(S_1) + \mu(A) + |S_0 \setminus S_1|$. Nous obtenons donc une solution duale, de la forme désirée, d'une valeur atteignant la longueur d'un tour, démontrant l'optimalité des deux. Ceci termine l'induction.

Théorème 6.1.5. *Pour tout cograph G connexe, $\lambda_w(G) = \sigma_w(G) = \pi_w(G)$.*

Nous pourrions facilement déduire des raisonnements ci-dessus un algorithme efficace pour trouver le tour minimum et le packing d'ensembles éclatants maximum.

6.2 Séparation

Le problème de séparation tel que nous l'avons généralisé est facile à résoudre sur les cographes. En effet, pour un cograph connexe, tout stable est inclu dans une co-composante, et tout bloqueur d'un tel stable contient toutes les autres co-composantes, plus un bloqueur de la co-composante contenant le stable. Si $G = K[G_1, \ldots, G_k]$, et Opt_i est l'optimum pour G_i, l'optimum pour G est donc donné par $\max_{i \in [\![1,k]\!]} Opt_i + x(V(G)) - x(V(G_i))$. Nous en déduisons un algorithme récursif.

Chapitre 7

Tours et graphes d'intervalles

7.1 Modification de l'algorithme de Keil

Rappelons la définition des graphes d'intervalles. Un graphe G est un *graphe d'intersection* s'il existe une bijection des sommets de G vers une famille F d'ensembles telle que deux sommets sont adjacents dans G ssi leurs images dans F s'intersectent. Un *graphe d'intervalle* est un graphe d'intersection d'intervalles de $\mathbb{R}$ (F est une famille d'intervalles réels).
Pour des raisons pratiques, nous allons représenter les graphes d'intervalles par des arbres de cliques. Un *graphe triangulé* est un graphe sans C_k induit pour $k \geq 4$. L'*arbre de cliques* d'un graphe triangulé est un arbre dont les sommets sont les cliques maximales de G, deux cliques étant adjacentes si leur intersection est un séparateur. Il s'agit effectivement d'un arbre pour les graphes triangulés connexes, avec la propriété que l'ensemble des cliques contenant un sommet donné est un sous-arbre.
Les graphes d'intervalles sont des graphes triangulés. L'arbre de cliques des graphes d'intervalles est une chaîne [21], et s'obtient comme produit de l'algorithme de reconnaissance des graphes triangulés en temps linéaire de Booth et Lueker [3]. Dorénavant, nous utiliserons donc la représentation en arbre de cliques pour les graphes d'intervalles.
Nous commençons par expliquer une modification d'un algorithme de Keil [29] (qui fut généralisé par Damashke [8]), pour décider si G est Hamiltonien. L'algorithme est modifié pour construire un ensemble critique de sommets, qui est éclatant lorsque le graphe n'est pas Hamiltonien.

Soit G un graphe d'intervalle connexe, donné sous la forme d'un arbre de clique $C_1, \ldots, C_p$ (dans l'ordre de la chaîne). Notons $S_i = C_i \cap C_{i+1}$ le $i^{\text{ième}}$

séparateur. Tout sommet de G appartient à des cliques successives, pour $u \in V$, nous notons donc $l(u)$ le plus petit indice tel que C_i contienne u, et $r(u)$ le plus grand de ces indices. Nous dirons que $u \preceq v$ (u est *plus court* que v, ou v est *plus long* que u) si $r(u) \leq r(v)$, $\preceq$ est donc un préordre sur V.
Nous présentons l'Algorithme 1 pour trouver un cycle Hamiltonien ou un ensembe éclatant dans tout graphe d'intervalle connexe. Pour un chemin P d'extrémité s et t, avec $s \preceq t$, nous entendons par l'expression *allonger P vers u* (adjacent à s) : former un nouveau chemin d'extrémité u et t, et contenant P comme sous-chemin. Par la suite, nous noterons $P \cdot u$ le chemin obtenu par l'allongement de P vers u

Le principe est de construire un chemin à partir de la gauche (disons) de l'arbre de cliques, et d'ajouter itérativement les sommets d'un côté ou de l'autre du chemin. Les choix sont effectués de manière gloutonne : nous ajoutons toujours les sommets sur l'extrémité la plus courte du chemin, et nous choisissons le plus court sommet possible à ajouter, qui doit donc être adjacent à l'extrémité courte du chemin.

Prouvons d'abord des invariants simples, permettant de montrer que toutes les opérations de l'algorithme sont valides, notamment les accès à $A(u)$ pour certains u. En particulier, il existe toujours un sommet dans $C_1 \setminus C_2$ (car G n'est complet), justifiant l'initialisation de P.

Proposition 7.1.1. *L'Algorithme 1 possède les invariants suivants :*

(i) *P est un chemin.*

(ii) *les deux extrémités de P sont dans C_i.*

(iii) *Tout sommet u tel que $r(u) < i$ est dans $V(P)$.*

(iv) *Pour tout sommet u dans P, à la ligne 5, si $u \in S_i$, $A(u)$ est défini.*

(v) $\max r(u)$ *pour u extrémité de P croît durant l'exécution de l'algorithme.*

Preuve. (i) et (ii) C'est vrai à l'initialisation de P. Lors de l'ajout d'un sommet u à l'étape i, $u \in C_i$ donc les invariants sont conservés. Lors de l'incrémentation de i, par la condition de la ligne 5, les deux extrémités sont bien dans C_{i+1}, ce qui permet l'incrémentation.
(iii) Par la condition de la ligne 3, tout sommet dans $C_i \setminus C_{i+1}$ est bien dans P lors de l'incrémentation de i vers $i+1$
(iv) Si $u \in S_i$ est ajouté à P lors de l'étape i, alors il est ajouté à la ligne 8, et donc $A(u)$ est défini immédiatement après, donc sera bien défini lors des prochains passages à la ligne 5.

Entrée : Un graphe d'intervalle G connexe non complet, donné par son arbre de cliques $C_1, C_2, \ldots, C_p$

Sortie : Un chemin Hamiltonien s'il en existe, un ensemble éclatant sinon.

Algorithme :

1: Soit P un chemin sur les sommets de $C_1 \setminus C_2$.
2: **pour** i **de** 1 **à** $p-1$ **faire**
3: **tant que** il existe un sommet $u \in C_i \setminus (C_{i+1} \cup V(P))$ **faire**
4: Allonger P vers u
5: **tant que** une des extrémités de P n'est pas dans C_{i+1} **faire**
6: **si** $S_i \setminus V(P) \neq \emptyset$ **alors**
7: Soit u le plus court sommet de $S_i \setminus V(P)$
8: Allonger P vers u
9: **si** $S_i \cap V(P) \setminus \{u\} \neq \emptyset$ **alors**
10: soit v le dernier sommet ajouté à P parmi $S_i \cap P \setminus \{u\}$
11: **si** aucun sommet n'a été ajouté à P après v et avant u **alors**
12: $A(u) \leftarrow \{u, v\}$
13: **sinon**
14: $A(u) \leftarrow A(v) \cup \{u\}$
15: **sinon**
16: $A(u) \leftarrow \{u\}$
17: **sinon**
18: Soit v le dernier sommet de S_i ajouté à P
19: **retourner** l'ensemble éclatant $A(v)$
20: Ajouter les sommets manquant de $C_p \setminus P$ à P, fermer P en un tour Hamiltonien, le **retourner**

ALG 1: *Un algorithme déterminant un cycle Hamiltonien ou un ensemble éclatant dans un graphe d'intervalle.*

(v) P est toujours étendu par l'extrémité la plus courte. □

La terminaison de l'algorithme est assuré par le fait qu'on agrandit P à chaque itération d'une boucle ***tant que***. Une conséquence immédiate des invariants et de la terminaison est que si l'algorithme dit retourner un cycle Hamiltonien, il s'agit effectivement d'un cycle Hamiltonien. Il faut maintenant montrer que si l'algorithme retourne un ensemble à la ligne 19, il s'agit effectivement d'un ensemble éclatant. Ceci nécessite le lemme suivant :

Lemme 7.1.2. *Soit u un sommet ajouté à l'itération i avec $u \in C_{i+1}$. Alors le nombre de composantes connexes de $V(P \cdot u) - A(u)$ dans le sous-graphe de G induit par $P \cdot u$ est au moins $|A(u)| - 1$, et $V(P \cdot u) \cap S_i \subseteq A(u)$.*

Preuve. Par induction sur l'ordre d'insertion des sommets dans P. Soit u tel que $A(u)$ est défini. Nous distinguons trois cas :

- $A(u)$ est défini à la ligne 16. La condition sur le nombre de composantes est évidente puisque $|A(u)| = 1$, et $V(P) \cap S_i = \{u\} \subseteq A(u)$.
- $A(u)$ est défini à la ligne 14 par $A(u) \leftarrow A(v) \cup \{u\}$, lors de l'itération i. Dans ce cas, v est le dernier sommet autre que u ajouté à P qui soit dans $S_i \cup V(P)$. Soit U l'ensemble des sommets ajoutés après v et avant u à P, $U \neq \emptyset$ par la condition de la ligne 11. Soit j l'itération pendant laquelle v a été ajouté à P. Pour tout $w \in U$, $w \notin S_i$ car v est le dernier ajouté dans $S_i \cup V(P)$. En particulier, w est plus court que v, donc $w \notin S_j$ par choix de v lors de l'itération j.
  Soit P' l'état du chemin après l'ajout de v. Par hypothèse d'induction, $S_i \cap V(P \cdot u) = S_i \cap (V(P \cdot u) - U) \subseteq S_i \cap (V(P') \cup \{u\})$, donc $V(P \cdot u) \cap S_i \subseteq A(u)$. De plus, pour tout $w \in U$, $j < l(u) \leq r(u) \leq i$, donc les sommets de U ne sont donc pas adjacents aux composantes créés par $A(v)$ dans le sous-graphe induit par P'. Le nombre de composantes connexes de $V(P \cdot u) - A(u)$ dans le sous-graphe de G induit par $P \cdot u$ est égal au nombre de composantes créés par $A(v)$ dans le sous-graphe induit par P', plus un pour la composante induite par U, ce qui est donc supérieur ou égal à $|A(u)| - 1$.
- $A(u)$ est défini à la ligne 12 par $A(u) \leftarrow \{u, v\}$, lors de l'itération i. Il suffit donc de montrer que $S_i \cap P = \{u, v\}$. L'ajout de u à P a été provoqué par l'existence d'une extrémité w de P juste avant l'ajout de u, telle que $w \in C_i \setminus C_{i+1}$. De plus w a été ajouté à P avant v, par la condition de la ligne 11. D'après le (v) de la Proposition 7.1.1, tous les sommets ajoutés avant v sont plus courts que w, ce qui prouve $S_i \cap P = \{u, v\}$.

□

Théorème 7.1.3. *L'Algorithme 1 est correct.*

Preuve. Comme remarqué précédemment, il suffit de montrer que si l'algorithme retourne un ensemble $S = A(v)$ à la ligne 19 lors de l'itération i, S est un ensemble éclatant. Comme $C_i \setminus C_{i-1}$ est non-vide (C_1 si $i = 1$), il y a deux cas :

- v a été ajouté à P pendant l'itération i. Par (v) de la Proposition 7.1.1, aucun sommet de $P \setminus \{v\}$ n'appartient à S_i, v est un point d'articulation, et $A(v) = \{v\}$.
- l'ensemble U des sommets ajoutés après v à P est non-vide. Comme dans la preuve du Lemme 7.1.2 aucun sommet de U n'appartient à S_i par choix de v à l'itération i, donc tous les sommets de U sont strictement plus courts que v, et donc aucun d'eux n'appartient à S_j, avec j l'itération pendant laquelle v a été ajouté au chemin. De plus, il existe un sommet $w \in C_p \setminus C_{p-1}$. Les composantes connexes induites par $V(G) - S$ sont donc celles induites par S dans le sous-graphe de G induit par P_v, plus U, plus au moins une composantes contenant $C_p \setminus C_{p-1}$, soit au moins $S + 1$ composantes par le Lemme 7.1.2.

Dans les deux cas, S est un ensemble éclatant. □

Corollaire 7.1.4 (Manacher, Mankus, Smith, 1996)**.** *Soit G un graphe d'intervalles connexe. G est non-Hamiltonien ssi G possède un ensemble éclatant.*
□

Manacher, Mankus et Smith [39] utilisent un algorithme assez proche, qui lui aussi construit un chemin en forme de U, et cherche un ensemble éclatant avec une technique similaire. Cependant, les deux algorithmes sont différents dans l'ordre d'insertion des éléments dans les chemins. De plus, ils utilisent une procédure annexe pour calculer l'ensemble éclatant. On pourrait aussi étendre leur algorithme que nous allons le faire pour le notre afin de trouver une partition d'ensembles éclatants optimale. Notre formalisme parait néanmoins plus simple à manipuler. D'autre part, Habib, Möhring et Steiner [23] donnent un algorithme sur les graphes de co-comparabilité, dont celui que nous donnons semble être une instanciation au cas particulier de la recherche d'un cycle Hamiltonien sur un graphe d'intervalle.

Les Figures 7.1 à 7.12 donnent un exemple de l'exécution de l'algorithme sur un graphe d'intervalle à 23 sommets. Les cliques maximales sont représentées par des rectangles gris, les sommets par des intervalles parcourant ces cliques. Les sommets noirs sont ceux qui n'appartiennent pas encore à P, les

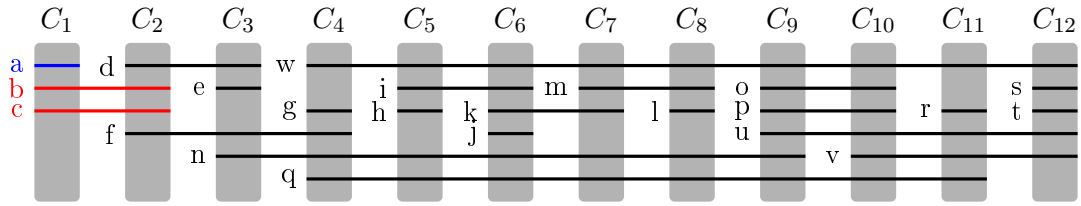


FIGURE 7.1 – *On initialise le chemin avec a, puis on ajoute b et c, avec* $A(b) = \{b\}$, $A(c) = \{b, c\}$ *créant une composante avec a. État du chemin : bac*

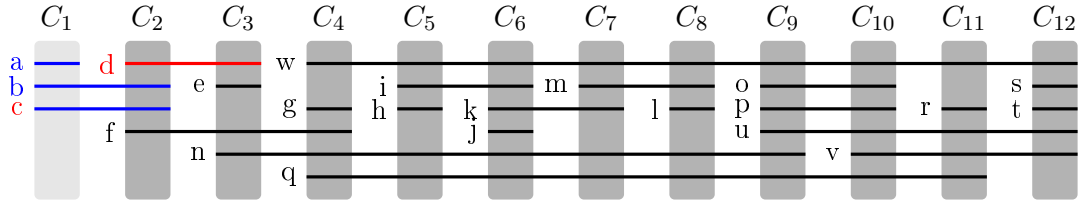


FIGURE 7.2 – *Il faut remplacer b comme extrémité de chemin, puisqu'il n'apparait pas dans* C_3, *on ajoute d,* $A(d) = \{d\}$ *(ligne 16). État du chemin : dbac*

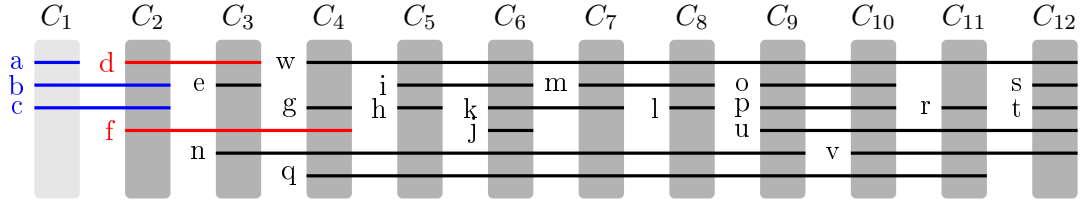


FIGURE 7.3 – *Il faut remplacer c qui n'apparait pas dans* C_3, *on ajoute f,* $A(f) = \{d, f\}$ *(ligne 12). État du chemin : dbacf*

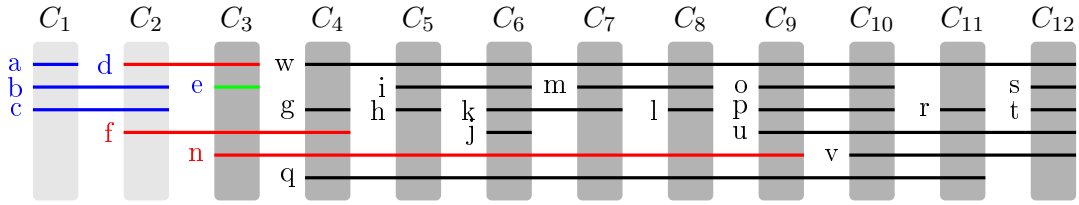


FIGURE 7.4 – *Dans C_3, on doit placer e après d, puis n. $A(n) = A(f) \cup \{n\} = \{d, f, n\}$. e forme une nouvelle composante. État du chemin : nedbacf*

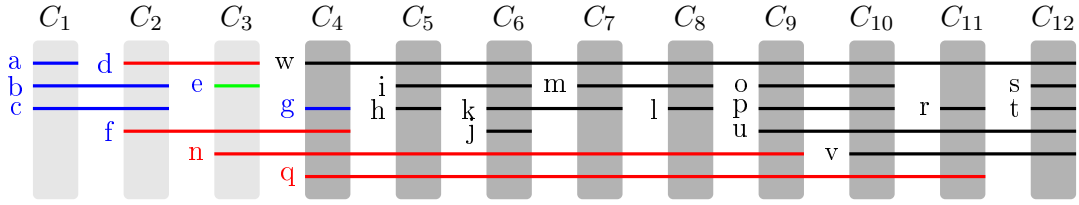


FIGURE 7.5 – *Dans C_4, on doit placer g, puis q. $A(q) = A(n) \cup \{q\} = \{d, f, n, q\}$. g forme une nouvelle composante. État du chemin : nedbacfgq*

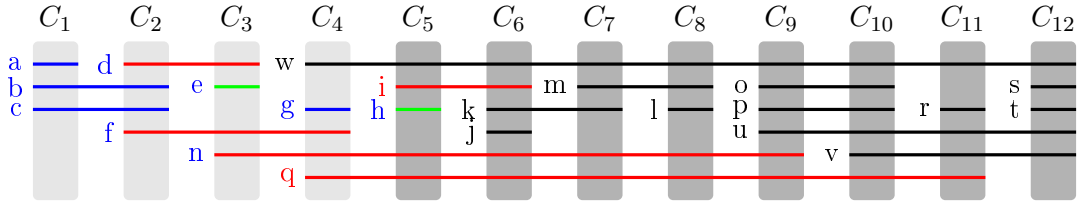


FIGURE 7.6 – *Dans C_5, on doit placer h, puis i. $A(i) = A(q) \cup \{i\} = \{d, f, n, q, i\}$. h forme une nouvelle composante. État du chemin : ihnedbacfgq*

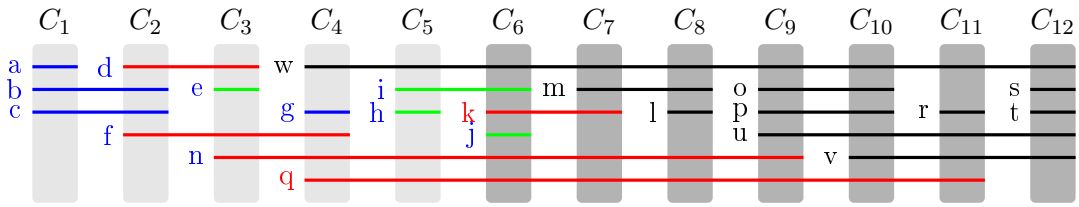


FIGURE 7.7 – *Dans C_6, on doit remplacer i par j, puis k. $A(k) = A(q) \cup \{k\} = \{d, f, n, q, k\}$. h, i, j forment une nouvelle composante. État du chemin : kjihnedbacfgq*

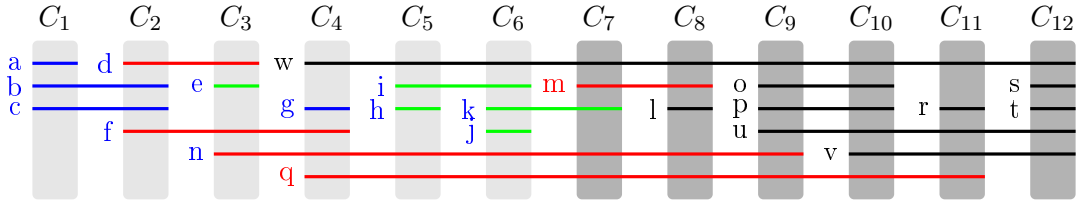


FIGURE 7.8 – *Dans C_7, on doit remplacer k par m. $A(m) = A(q) \cup \{k\} = \{d, f, n, q, m\}$. h, i, j, k forment une nouvelle composante. État du chemin : mkjihnedbacfgq*

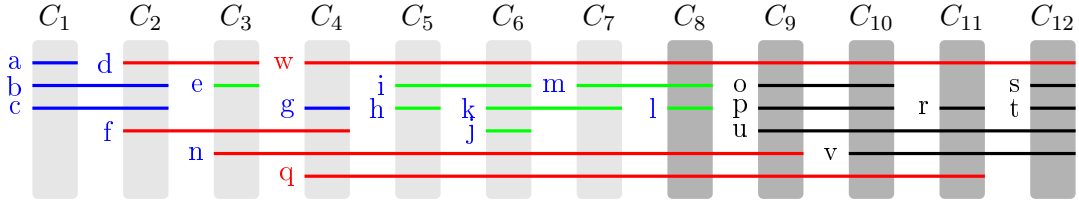


FIGURE 7.9 – *Dans C_8, on doit remplacer m par l puis w. $A(w) = A(q) \cup \{w\} = \{d, f, n, q, w\}$. h, i, j, k, m, l forment une nouvelle composante. État du chemin : wlmkjihnedbacfgq*

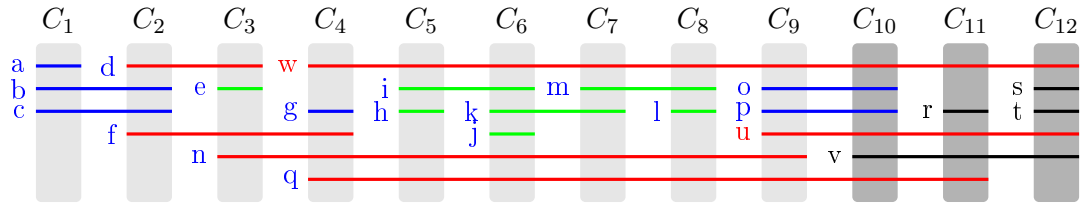


FIGURE 7.10 – *Il ne se passe rien dans C_9. Dans C_{10}, on place o, p, u dans cet ordre. $A(u) = A(w) \cup \{u\} = \{d, f, n, q, w, u\}$. o, p forment une nouvelle composante. État du chemin : wlmkjihnedbacfgqopu*

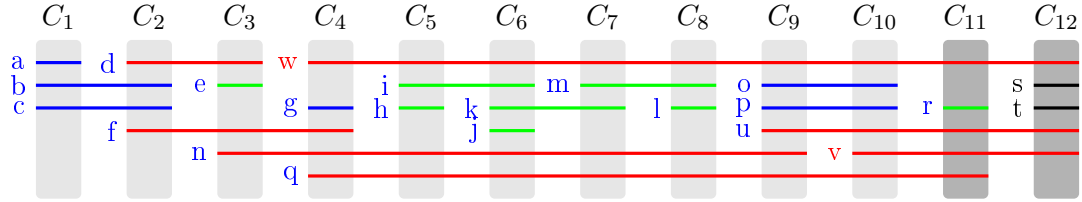


FIGURE 7.11 – *Dans C_{11}, on place r puis v. $A(v) = A(u) \cup \{v\} = \{d, f, n, q, w, u, v\}$. r forme une nouvelle composante. État du chemin : wlmkjihnedbacfnqopurv. Dans cette situation, on atteint la dernière clique, le cycle Hamiltonien trouvé est wlmkjihnedbacfgqopurvstw.*

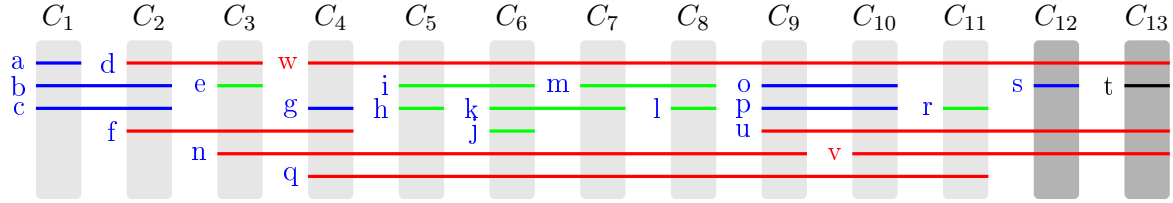


FIGURE 7.12 – *Si s et t sont disjoints, il faudrait ajouter s après v, puis l'algorithme ne trouve pas de candidat et retourne l'ensemble éclatant $A(v) = \{d, f, n, q, u, v, w\}$. L'une des composantes est formée par s, et une autre par t.*

sommets rouges correspondent à $A(u)$ avec u le dernier sommet ajouté à P. Les lettres identifiants chaque sommet sont en bleu lorsque le sommet est à l'intérieur de P, et en rouge s'il s'agit d'une extrémité de P. La coloration en bleu et vert des intervalles permet de bien distinguer les composantes connexes du graphe induit par $V(P_u) - A(u)$.

7.2 Partition en ensembles éclatants

Nous aimerions maintenant montrer que $\lambda(G) = \pi(G)$ lorsque G est un graphe d'intervalle. Pour cela nous allons donner un algorithme, conçu à partir de l'Algorithme 1, qui trouve un tour de longueur minimale, et une partition de $V(G)$ en ensembles maximisant la somme des déficiences, qui sera alors égale à la longueur du plus court tour, ce qui sera suffisant d'après les inéquations (5.1). Remarquons que l'Algorithme 1 lorsqu'il trouve un ensemble éclatant, contruit aussi un cycle sur une partie des sommets de G (le chemin P, qui peut être fermé en un cycle).

Entrée : Un graphe d'intervalle G connexe.

Sortie : Un tour minimal de G et une partition de $V(G)$ maximisant la somme des déficiences.

Algorithme :

1: **si** G est Hamiltonien **alors**
2: **Retourner** un cycle Hamiltonien, et $\{\{v\} \ : \ v \in V\}$
3: **sinon**
4: Soit S l'ensemble éclatant retourné par l'Algorithme 1, et C le cycle sur une partie $U \subseteq V(G)$ obtenu.
5: Soit u un plus long sommet de S
6: Trouver récursivement un tour et une partition $\mathcal{P}'$ optimals pour $G[(V(G) \setminus U) + u]$
7: Fusionner le tour avec le cycle C par le sommet u
8: **Retourner** ce tour et la partition $(\mathcal{P}' - \{P_u\}) \cup \{P_u \cup S\} \cup \{\{v\} \ : \ v \in U \setminus S\}$, où P_u est la partie de $\mathcal{P}'$ contenant u.

ALG 2: *Un algorithme permettant de trouver un tour de longueur minimale et une partition de déficience maximale dans un graphe d'intervalle.*

L'algorithme commence par déterminer si le graphe est Hamiltonien. Dans ce cas, la solution est évidente. Sinon, l'Algorithme 1 détermine un ensemble éclatant S et un tour des sommets de $C_1 \cup \ldots \cup C_i = U$ (le i étant l'indice de la dernière itération de l'Algorithme 1). L'idée est d'utiliser deux fois dans le tour de G le sommet le plus grand de S, puisque de toute façon un des sommets de S devra être utilisé plus d'une fois, et que celui-ci est susceptible d'être le plus utile pour connecter le reste du graphe. Nous appelons donc l'algorithme récursivement sur $G' = G[(V(G) \setminus U) + u]$. Le tour ainsi trouvé peut être raccroché au tour de U par le sommet commun u, et la partition peut être fusionné avec la partition $\{S\} \cup \{\{u\} \ : \ u \in U \setminus S\}$ de U, formant alors une partition de $V(G)$ notée $\mathcal{P}$.

Pour montrer la correction de cette méthode, il suffit de montrer que la somme des déficiences pour $\mathcal{P}$ est supérieure de 1 à celle de $\mathcal{P}'$ (puisqu'on a utilisé u une fois de plus).

Proposition 7.2.1. *Soit $P \in \mathcal{P}'$ ne contenant pas u, alors $c_G(P) = c_{G'}(P)$.*

Preuve. Dans G', les composantes de $G' - P$ ne contenant pas u ne peuvent pas contenir de sommet dans $N(u) \setminus P$. Puisque u est le plus long sommet de U, tout sommet de $V(G')$ adjacent à un sommet de S_i est adjacent à U, ces composantes ne peuvent donc pas intersecter U. □

Lemme 7.2.2. *$def_G(P_u \cup S) = def_{G'}(P_u) + 1$*

Preuve. Nous le démontrons par récurrence sur le nombre d'appel récursif, en même temps que la propriété suivante : pour toute partie $P \in \mathcal{P}$, $c(P) - |P| \geq 0$. Cette dernière est vraie pour les singletons, et reste vrai pour les parties ne contenant pas u par la Proposition 7.2.1. Nous l'utilisons pour montrer que la déficience de la partie contenant u augmente strictement, donc entre autres est strictement positive.

$$\begin{aligned}
\mathrm{def}_G(P_u \cup S) &= c_G(P_u \cup S) - |P_u \cup S| \\
&= c_{G'}(P_u) + (c_G(S) - c_{G'}(\{u\})) - (|P_u| + |S| - 1) \\
&= c_{G'}(P_u) - |P_u| + c_G(S) - |S| - c_{G'}(\{u\}) + 1 \\
&= \mathrm{def}_{G'}(P_u) + (\mathrm{def}_G(S) - c_{G'}(\{u\})) + 1 \\
&= \mathrm{def}_{G'}(P_u) + 1
\end{aligned}$$

car $\mathrm{def}_G(S) = c_{G'}(\{u\})$, puisque nous avions exhibé $|S|$ composantes de $G - S$ dans $U \setminus S$. □

Théorème 7.2.3. *L'Algorithme 2 est correct : le tour et la partition retournés sont optimaux, et* $\lambda(G) = \pi(G)$.

Preuve. Si G est Hamiltonien, la déficience de la partition est de 0, donc $\pi(G) = |V| = \lambda(G)$.
Si G n'est pas Hamiltonien, par récurrence sur le nombre d'appel récursif, la longueur l du tour trouvé est égal à $|V|$ plus la somme des déficiences de la partition. □

7.3 Graphes d'intervalles avec sommets optionnels

Nous montrons qu'une légère modification dans les Algorithmes 1 et 2 permet de résoudre le problème avec sommets optionnels, tel que défini en Section 5.3. Dans le cas simple, sans optionnalité, étudié précédemment, la longueur du chemin le plus court était donné par la partition maximisant la somme de la fonction c. Avec les sommets optionnels, c'est la partition maximisant la somme de $\widetilde{c}$ qui donne la longueur d'un plus court tour. La preuve se fait par induction sur le nombre de sommets optionnels et sur la taille de G. Nous prouvons qu'il existe un tour des sommets obligatoires et une partition tel que le nombre de passages du tour par chaque partie de la partition est exactement la valeur de $\widetilde{c_G}$ pour cette partie.
Nous exécutons l'Algorithme 1 en ignorant les sommets optionnels. Si l'algorithme trouve un tour Hamiltonien, la partition optimale consiste à prendre les singletons. Sinon, l'algorithme échoue à l'itération i, avec un cycle C et un ensemble $S' = A(u)$ s'il existe un sommet dans $S_i \cap \widetilde{V}$, $S' = \emptyset$ sinon. Définissons $S = S' \cup (\bigcup_{j \in [\![1,i]\!]} C_j \setminus \widetilde{V})$, obtenu en ajoutant des sommets optionnel à S'. Ainsi, $\widetilde{c_G}(S)$ vaut au moins $|S'| + 1$, puisque S est un ensemble éclatant dans $G \setminus U$ (et si $S' = \emptyset$, $\widetilde{c_G}(S)$ vaut au moins 2).
La même intuition que dans le cas sans sommet optionnel se révèle correcte : nous choisissons un sommet u parmi les plus longs de S, si possible un obligatoire.

Si effectivement u est un sommet obligatoire, par l'hypothèse d'induction (l'algorithme est appliqué récursivement) sur $G' = G[(V(G) \setminus (S \cup V(C))) + u]$, nous trouvons un tour T' de G' et une partition $\mathcal{P}'$. T' et C possède le sommet u en commun, et peuvent être donc liés en un tour T de G. Comme dans le cas où tous les sommets sont obligatoires, nous contruisons la partition $\mathcal{P}$ en reprenant les parties de $\mathcal{P}'$ ne contenant pas u et en ajoutant S à la partie contenant u, les autres sommets étant ajoutés comme singletons. Toute partie P de $\mathcal{P}'$ ne contenant pas u vérifie $\widetilde{c_G}(P) \geq \widetilde{c_{G'}}(P)$, puisque u

est le plus long des sommets de C. Pour la partie P_u contenant u, à laquelle S' est ajouté, les composantes connexes de P_u dans G' sont conservées, et les composantes de S dans G aussi, sauf celles contenant $(C_{i+1} \cup \ldots \cup C_n) \setminus S_i$ dont le nombre est égal à $\widetilde{c_G}(S') - |S'| \geq 1$. Ainsi :

$$\widetilde{c_G}(P_u \cup S) = \widetilde{c_{G'}}(P_u) + \widetilde{c_G}(S) - (\widetilde{c_G}(S) - |S'|) \tag{7.1}$$
$$= \widetilde{c_{G'}}(P_u) + |S'| \tag{7.2}$$

Le premier terme de (7.2) correspond au nombre de passages de T' parmi les sommets de P_u, le second terme au nombre de passages de C parmi les sommets de S. Au final, ceci prouve que le nombre de passages du tour par n'importe quelle partie de $\mathcal{P}$ est bien la valeur de $\widetilde{c_G}$ sur cette partie, ce qui termine le premier cas.

Si u est un sommet optionnel, nous relançons l'algorithme sur $(G, \widetilde{V} \cup \{u\})$. Par hypothèse d'induction, puisque le nombre de sommets optionnels a diminué, il existe un tour T de G et une partition $\mathcal{P}'$ de même valeur. Si la partie P_u contenant u contient un autre sommet, $\widetilde{c_G}(P_u)$ a la même valeur pour les ensembles d'obligatoires $\widetilde{V}$ ou $\widetilde{V} + \cup\{u\}$, il n'y a donc rien à prouver. Sinon, $P_u = \{u\}$. Supposons alors qu'il existe un sommet $s \in S$ tel que s n'appartiennent pas à un singleton dans $\mathcal{P}'$, notons T cette partie. En tant qu'ensemble déconnectant, T est l'union de séparateur minimaux de G, autrement dit il existe un sous-ensemble I de $[\![1, n-1]\!]$ tel que $T = \bigcup_{i \in I} S_i$. Puisque u n'est pas dans T (car $P_u = \{u\}$), I et $[\![l(u), r(u)-1]\!]$ sont disjoints. Notons $T^- := \bigcup_{i \in I, i < l(u)} S_i$ et $T^+ := \bigcup_{i \in I, i \geq r(u)} S_i$, T^+ et T^- forment une partition de T (par le choix de u comme étant le plus long sommet de S_i). De plus, si l'ensemble des obligatoires contient u, on a $\widetilde{c_G}(T^+) + \widetilde{c_G}(T^-) \geq \widetilde{c_G}(T) + 1$ car la composante contenant u est compté deux fois. Ceci contredit la maximalité de la partition $\mathcal{P}'$, partition maximale par l'existence de T. Ainsi, tous les éléments de S sont des singletons dans $\mathcal{P}'$, $\mathcal{P}$est alors défini en retirant ces singletons et en ajoutant la partie S. La valeur de cette partition pour $\widetilde{c_G}$ est alors la même que la valeur de la partition $\mathcal{P}'$ avec u obligatoire, ce qui prouve l'optimalité de cette partition et du tour T.

Ceci démontre :

Théorème 7.3.1. *Soit G un graphe d'intervalle et $\widetilde{V}$ un sous-ensemble de sommets de G. La longueur minimum d'un tour passant par tous les sommets obligatoires est égal au maximum de la somme de $\widetilde{c_G}$ pris sur toutes les partitions de $V(G)$.*

De plus, on peut trouver le tour minimum et la partition maximum en temps polynomial.

7.4 Séparation

Nous donnons maintenant un algorithme pour résoudre le problème (5.6) :

$$\max_{S,U} y(S) - x(U)$$

avec S décrivant les stables de G, et U bloqueur des S-chemins, avec x et y deux vecteurs positifs. Rappelons que ceci permet de calculer aussi la solidité du graphe d'intervalle, en particulier déterminer s'il existe un ensemble éclatant, ainsi que trouver un stable maximum.
Notre algorithme est dynamique et procède sur l'arbre de clique. Remarquons que pour une solution S, U, U est l'union de plusieurs séparateurs minimaux car x est positif, et que les séparateurs minimaux d'un graphe d'intervalle correspondent aux arêtes de l'arbre de cliques. S est alors constitué, d'au plus un sommet entre deux séparateurs minimaux inclus dans U, consécutifs dans l'arbre de cliques. Soit $C_1, C_2 \ldots, C_n$ l'arbre de clique de G. Rappelons que S_i, pour $i \in [\![1, n-1]\!]$, est le séparateur minimal $V(C_i) \cap V(C_{i+1})$, et G_i le graphe induit par $V(C_1) \cup V(C_2) \ldots \cup V(C_i)$ pour $i \in [\![1, n]\!]$. Par commodité, on pose $S_0 = S_n = \emptyset$. Pour tout $i \in [\![1, n]\!]$, nous calculons le maximum $M(i)$ de $y(S) - x(U)$ avec S un stable de G_i et U bloqueur des S-chemins contenant S_i, mais pas $S_{i+1}, \ldots, S_{n-1}$ (en particulier, U et S sont disjoints).
Pour $0 \leq i < j$, notons $y(i, j)$ le maximum de $\{y(u) : u \in C_k, k \in [\![i+1, j]\!], u \notin S_i \cup S_j\}$, autrement dit la maximum atteint par y sur les sommets strictement entre le $i^{\text{ième}}$ et le $j^{\text{ième}}$ séparateur, et 0 s'il n'existe pas de tel sommet. Alors pour tout $i < j$:

$$M(i) = \max\{y(0, i) - x(S_i), \max_{j \in [\![1, i-1]\!]} M(j) - x(S_i \setminus S_j) + y(j, i)\} \qquad (7.3)$$

Autrement dit, le problème se décompose selon le dernier séparateur S_j pris. Il y a $i - 1$ séparateurs possibles, plus la possibilité de ne pas prendre de séparateur. Si j atteint le maximum pour le calcul de $M(i)$, on obtient le bloqueur correspondant à la solution de $M(i)$ en ajoutant S_i au bloqueur de $M(j)$ et le stable s'obtient en ajoutant le sommet définissant $y(j, i)$.

Théorème 7.4.1. *Pour les graphes d'intervalles, $M(n)$ est égal au maximum de (5.6), et peut être calculé en temps polynomial.*

Preuve. Plus généralement, nous montrons que $M(i)$ est le maximum de $y(S) - x(U)$ avec S un stable de G_i et U bloqueur des S-chemins contenant

S_i comme nous l'avons prétendu. Par récurrence sur i. Soit Opt le maximum pour i, atteint pas S_{opt}, U_{Opt}. Notons que $M(i)$ correspond à une solution réalisable par construction. Si S_{Opt} ne contient qu'un seul sommet, Opt est trivialement égal à $y(0,i) - x(S_i)$, donc puisque $Opt \geq M(i) \geq Opt$, $M(i)$ est lui-même optimal.
Sinon, soit $j < i$ maximum tel que $S_j \subseteq U_{Opt}$, alors $S_{Opt} \setminus (C_{j+1} \cup \ldots \cup C_i), U_{opt} \setminus (S_i \setminus S_j))$ définit une solution pour le rang j, donc de valeur égale à $M(j)$. Alors :

$$\begin{aligned} Opt &= M(j) + y(S_{Opt} \cap (C_{j+1} \cup \ldots \cup C_i)) - x(S_i \setminus S_j) && (7.4)\\ &\leq M(j) + y(j,i) + x(S_i \setminus S_j) && (7.5)\\ &\leq M(i) \leq Opt && (7.6) \end{aligned}$$

Donc $M(i)$ est bien maximum. Le caractère polynomial de cet algorithme est évident. □

En supposant que les calculs de $y(j,i)$ et $x(S_i \setminus S_j)$ sont instantanées, la complexité pour calculer $M(n)$, optimum recherché, est en $O(n^2)$. Nous pouvons l'obtenir en précalculant $y(j,i)$ et $x(S_i \setminus S_j)$. Pour cela, il faut détailler la représentation des graphes d'intervalles que nous utilisons. Nous codons chaque sommet u par les indices minimum ($l(u)$) et maximum ($r(u)$) des cliques le contenant. De plus, pour chaque clique C_i nous est donné la liste des sommets d'indice minimum i ($C_i \setminus S_{i-1}$), et la liste de ceux d'indice maximum i ($C_i \setminus S_i$).
Un sommet appartient à $S_i \setminus S_j$ ssi $l(u) \leq i < r(u) \leq j$. Vérifier qu'un sommet appartient à un séparateur est donc instantané. De même, vérifier l'appartenance à une clique l'est aussi. Pour chaque $i \in [\![1,n]\!]$, nous calculons pour tout $j \in [\![1, i-1]\!]$ la valeur de $x(S_i \setminus S_j)$ en temps $O(|V|)$ de la façon suivante. Nous parcourons l'ensemble des cliques de C_1 à C_n dans l'ordre, en maintenant un compteur X initialement nul. Lors de la $k^{\text{ième}}$ itération, pour chaque sommet u de $(C_k \setminus S_{k-1}) \cap S_{i+1}$, X est augmenté de $x(u)$. Pour chaque sommet u de $(C_k \setminus S_k) \cap S_i$, nous retranchons $x(u)$ de X. Puis la valeur courante de X est affecté à $x(S_i \setminus S_j)$. Chaque sommet voit sa valeur pour x ajouté et retiré au plus une fois, donc pour chaque valeur possible de i le temps de calcul est $O(|V|)$, pour un temps total de $O(n|V|)$. Le calcul des $y(j,i)$ est similaire : à i fixé, les cliques $C_1, \ldots C_i$ sont parcourues dans l'ordre inverse, en gardant un compteur Y initialement égal à $-\infty$. Lors de l'itération sur la clique C_k, Y est remplacé par le maximum de Y et $\max y(u) \ : \ u \in C_k \setminus (S_{k-1} \cup S_i)$, puis la valeur courante de Y est affecté à $y(k,i)$. Le temps total est à nouveau $O(n|V|)$. La complexité finale de notre

problème est donc bien $O(n|V|)$.

Quatrième partie

Conclusion

Chapitre 8

La fin du début

Ce travail ouvre bien plus de questions qu'il n'en ferme. Sur les problèmes de multiflots, le tableau proposé devrait permettre de satisfaire l'appétit des amateurs pendant encore quelques années ; l'objectif ultime sera de résoudre le problème de l'aller-retour de Schrijver. La frontière entre polynomialité et NP-complétude reste assez floue dans plusieurs autres cas. Trouver des chemins disjoints dans les graphes planaires semble relativement simple dès lors que les chemins ne peuvent se croiser. Nous avons tenté d'étendre les résultats de Schrijver, qui cherche des chemins dans des classes d'homotopie fixées, pour prendre en compte les croisements. En faisant abstraction de la localisation des croisements, nous avons obtenu un résultat de pseudopolynomialité dans un cas assez contraint. Ce résultat pourrait naturellement s'étendre vers un algorithme polynomial. Une direction possible pour cela est d'analyser plus finement la structure du graphe planaire, en prenant en compte le fait qu'il ne possède qu'un nombre fixe de sources et de puits (les terminaux des demandes). Il semble raisonnable de croire que des techniques de réduction de certaines parties du graphe en petits graphes équivalents du point de vue du routage pourraient donner des algorithmes paramétrés efficaces. Par ailleurs, nous avons renforcé les résultats de NP-complétude, en montrant que plusieurs variantes du problème sont difficiles avec seulement deux demandes.

Une des clés pour bien comprendre les cas encore ouverts semble être l'étude des croisements entre les chemins. Par exemple, dans le cas du problème de l'aller-retour, est-il possible de trouver des descriptions simples de comment les deux chemins peuvent se croiser, et à partir de là en déduire un algorithme ? Nous avons aussi montré à quel point les croisements jouent un rôle dans nos résultats de NP-complétude. L'idée de contraindre certains

sommets à ne pas pouvoir croiser de chemins permet de définir plusieurs nouveaux problèmes. En particulier, permettons-nous de poser le problème suivant : quelle est la complexité du routage de chemins arête-disjoints, où tous les terminaux sont sur le bord d'une grille rectangulaire complète, mais où les croisements ne sont autorisés que sur un sous-ensemble précis de sommets ? Nous pensons que notre preuve de NP-complétude pour deux arêtes de demandes dans les graphes non-orientés planaires peut s'adapter pour traiter ce cas, mais plusieurs difficultés techniques nous bloquent encore.
Pour finir sur les chemins disjoints, rappelons que les problèmes que nous avons étudiés sont, en un sens, les plus simples : nous n'avons gardé que des paramètres naturels, et nous les avons étudié dans un contexte de décidabilité. Notre étude est donc (volontairement) très ciblée, et peut s'étendre dans diverses directions. Nous n'avons par exemple pas du tout abordé les problèmes de maximisation, et donc d'approximation. Nous n'imposons rien sur la nature des flots, si ce n'est l'intégralité, mais plusieurs applications exigent toutefois des conditions supplémentaires, en restreignant notamment les chemins envisageables.
Le problème de tours graphiques et d'inégalités d'ensembles éclatants nous semble très prometteur. Non seulement parce que nous avons, avec Vincent Jost, plusieurs conjectures élégantes, mais aussi parce que notre problème paraît être à la croisée de nombreux problèmes classiques de l'optimisation combinatoire, en leur donnant un éclairage particulier. Ceci nous laisse espérer une profondeur que nous ne soupçonnions pas au début de notre étude. Prouver l'intégralité du polyèdre correspondant aux empilements d'ensembles éclatants dans les graphes d'intervalle ne pourra être qu'une étape. Il est probablement assez simple d'établir notre bonne caractérisation dans les graphes de cocomparabilité, comme nous l'avons fait dans les graphes d'intervalle. Le vrai défi est plutôt posé par les graphes sans triplet astéroïdaux, puisque dans cette classe, le problème de l'hamiltonicité est ouvert. Il y a pourtant de belles raisons d'espérer ; l'activité autour de ces graphes s'est intensifié ces dernières années.
L'idée de choisir une bonne caractérisation puis chercher quels graphes possèdent cette caractérisation n'est pas originale en soit. Nous avons cité comme source d'inspiration le travail de Fonlupt et Naddef [13], nous pourrions ajouter par exemple toute la théorie des graphes parfaits, qui sont ceux pour lesquels l'indice chromatique est caractérisée par les cliques. Mais cette idée est encore sous-exploitée. Nous avons vu par le Théorème 5.4.2 comment dans certains graphes le Théorème 5.4.1 de chemins-disjoints de Mader admet une caractérisation plus simple. Il ne s'agit pas ici de trouver un nouveau cas de polynomialité, mais seulement de manipuler des objets moins

complexes. Une conséquence immédiate est que sur cette classe de graphes, trouver les chemins disjoints se réduit à résoudre un problème de flot, et non-plus un problème de couplage pondéré. La complexité algorithmique est donc réduite. Rechercher des caractérisations simples de problèmes même polynomiaux nous semble donc être une ligne de recherche viable et pleine d'intérêt, et nous réservant probablement encore de jolies surprises.

Bibliographie

[1] D. Bauer, H. Broersma et E. Schmeichel : Toughness in graphs–a survey. *Graphs and Combinatorics*, 22(1):1–35, 2006.

[2] C. Bentz : *Résolution exacte et approchée de problèmes de multiflot entier et de multicoupe : algorithmes et complexité*. Thèse de doctorat, Conservatoire National des Arts et Métiers, 2006.

[3] K.S. Booth et G.S. Lueker : Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *Journal of Computer and System Sciences*, 13(3):335–379, 1976.

[4] H. Broersma, L. Xiong et K. Yoshimoto : Toughness and hamiltonicity in k-trees. *Discrete Mathematics*, 307(7-8):832–838, 2007.

[5] Hajo Broersma, Ton Kloks, Dieter Kratsch et Haiko Müller : Independent sets in asteroidal triple-free graphs. *SIAM J. Discrete Math.*, 12(2):276–287, 1999.

[6] V. Chvatal : Tough graphs and hamiltonian cycles. *Discrete Math*, 5:215–228, 1973.

[7] V. Chvátal : Tough graphs and hamiltonian circuits. *Discrete Mathematics*, 306(10-11):910–917, 2006.

[8] P. Damaschke : The Hamiltonian circuit problem for circle graphs is NP-complete. *Information Processing Letters*, 32(1):1–2, 1989.

[9] J.S. Deogun, Kratsch D. et Steiner G. : 1-tough cocomparability graphs are hamiltonian. *Discrete Mathematics*, 170(1-3):99–106, 1997.

[10] J. Edmonds et R. Giles : A min-max relation for submodular functions on graphs. *In Studies in Integer Programming : Proceedings of the Institute of Operations Research Workshop, Sponsored by IBM, University of Bonn, Germany, Sept. 8-12, 1975*, volume 1, pages 185–204. North Holland, 1977.

[11] S. Even, A. Itai et A. Shamir : On the complexity of timetable and multicommodity flow problems. *SIAM Journal on Computing*, 5(4):691–703, décembre 1976.

[12] G. Farkas : On the applications of the mechanical principle of Fourier. *Mathematikai es termes-zettudomayi Ertesito*, 12:457–472, 1894.

[13] J. Fonlupt et D. Naddef : The traveling salesman problem in graphs with some excluded minors. *Mathematical Programming*, 53(1):147–172, 1992.

[14] S. Fortune, J. Hopcroft et J. Wyllie : The directed subgraph homeomorphism problem. *TCS : Theoretical Computer Science*, 10, 1980.

[15] A. Frank : Edge-disjoint paths in planar graphs. *Journal of combinatorial theory. Series B*, 39(2):164–178, 1985.

[16] A. Frank : *Paths, Flows and VLSI Layouts*, chapitre Packing paths, circuits and flows, pages 47–100. Springer, 1990.

[17] A. Frank, E. Tardos et A. Sebő : Covering directed and odd cuts. *Mathematical Programming study 22*, pages 99–112, 1984.

[18] D. Gale, H.W. Kuhn et A.W. Tucker : Linear programming and the theory of games. *Activity Analysis of Production and Allocation*, pages 317–329, 1951.

[19] J.F. Geelen et B. Guenin : Packing odd circuits in Eulerian graphs. *Journal of Combinatorial Theory, Series B*, 86(2):280–295, 2002.

[20] A.M.H. Gerards et A. Sebő : Total dual integrality implies local strong unimodularity. *Mathematical Programming*, 38(1):69–73, 1987.

[21] P.C. Gilmore et A.J. Hoffman : A characterization of comparability graphs and of interval graphs. *Canad. J. Math*, 16(99):539–548, 1964.

[22] M. Grötschel, L. Lovasz et A. Schrijver : The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

[23] M. Habib, R.H. Möhring et G. Steiner : Computing the bump number is easy. *Order*, 5(2):107–129, 1988.

[24] F.L. Hitchcock : The distribution of a product from several sources to numerous localities. *J. Math. Phys. Mass. Inst. Tech*, 20:224–230, 1941.

[25] AJ Hoffman : A generalization of max flow—min cut. *Mathematical Programming*, 6(1):352–359, 1974.

[26] T. Ibaraki et S. Poljak : Weak Three-Linking in Eulerian Dgraphs. *SIAM journal on Discrete Mathematics*, 4:84–98, 1991.

[27] L. Kantorovich : Mathematical methods of organising and planning production (translated from a report in russian, dated 1939). *Management Science*, 6:366–422, 1960.

[28] R.M. Karp : On the computational complexity of combinatorial problems. *Networks*, 5(1):45–68, 1975.

[29] J.M. Keil : Finding Hamiltonian circuits in interval graphs. *Information Processing Letters*, 20(4):201–206, 1985.

[30] B. Korte et J. Vygen : *Combinatorial optimization : theory and algorithms*. Springer, 2002.

[31] M.R. Kramer et J. van Leeuwen : The complexity of wire-routing and finding minimum area layouts for arbitrary VLSI circuits. *Advances in computing research*, 2:129–146, 1984.

[32] D. Kratsch, J. Lehel et H. Muller : Toughness, hamiltonicity and split graphs. *Discrete Mathematics*, 150(1):231–245, 1996.

[33] D. Lichtenstein : Planar formulae and their uses. *SIAM journal on computing*, 11:329, 1982.

[34] M.V. Lomonosov : Combinatorial approaches to multiflow problems. *Discrete Applied Mathematics*, 11(1):1–93, 1985.

[35] L. Lovász : *Combinatorial problems and exercises*. Chelsea Pub Co, 2007.

[36] C.L. Lucchesi et D.H. Younger : A minimax theorem for directed graphs. *J. London Math. Soc*, 17(2):369–374, 1978.

[37] J.F. Lynch : The equivalence of theorem proving and the interconnection problem. *ACM SIGDA Newsletter*, 5(3):31–36, 1975.

[38] W. Mader :
"Uber die Maximalzahl kreuzungsfreier H-Wege. *Archiv der Mathematik*, 31(1):387–402, 1978.

[39] G.K. Manacher, T.A. Mankus et C.J. Smith : An optimum theta(n log n) algorithm for finding a canonical hamiltonian path and a canonical hamiltonian circuit in a set of intervals. *Information Processing Letters*, 35, 1990.

[40] D. Marx : Eulerian disjoint paths problem in grid graphs is NP-complete. *Discrete Applied Mathematics*, 143(1-3):336–341, 2004.

[41] M. Middendorf et F. Pfeiffer : On the complexity of the disjoint paths problem. *Combinatorica*, 13(1):97–107, 1993.

[42] D. Müller : On the complexity of the planar directed edge-disjoint paths problem. *Mathematical Programming*, 105(2):275–288, 2006.

[43] H. Nagamochi et T. Ibaraki : Multicommodity flows in certain planar directed networks. *Discrete Appl. Math.*, 27(1):125–145, 1990.

[44] C.St.J.A. Nash-Williams : Exercice 6.56 in [35], 71.1a in [57].

[45] G. Naves : The hardness of routing two pairs on one face. *Mathematical Programming*, 2010.

[46] G. Naves et A. Sebő : Multiflow Feasibility : an Annotated Tableau. *Research Trends in Combinatorial Optimization*, pages 261–283, 2008.

[47] H. Okamura et P.D. Seymour : Multicommodity flows in planar graphs. *J. Combinat. Theory, Ser. B*, 31(1):75–81, 1981.

[48] K. Onaga et O. Kakusho : On feasibility conditions of multicommodity flows in networks. *IEEE Transactions on Circuit Theory*, 18(4):425–429, 1971.

[49] F. Pfeiffer : Zur Komplexitat des Disjunkte-Wege-Problems. *PhD., Universitat Bonn*, 1990.

[50] A. Recski : *Matroid theory and its applications in electric network theory and in statics*. Springer Verlag, 1989.

[51] N. Robertson et P.D. Seymour : Graph minors. XIII. The disjoint paths problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995.

[52] B. Rothschild et A. Whinston : Feasibility of two commodity network flows. *Operations Research*, pages 1121–1129, 1966.

[53] A. Schrijver : *Theory of linear and integer programming*. Wiley, 1986.

[54] A. Schrijver : The Klein bottle and multicommodity flows. *Combinatorica*, 9(4):375–384, 1989.

[55] A. Schrijver : Homotopic routing methods. *Paths, Flows, and VLSI-Layout*, pages 329–371, 1990.

[56] A. Schrijver : Finding k Disjoint Paths in a Directed Planar Graph. *SIAM Journal on Computing*, 23:780, 1994.

[57] A. Schrijver : *Combinatorial optimization : polyhedra and efficiency*. Springer, 2003.

[58] W. Schwärzler : On the complexity of the planar edge-disjoint paths problem with terminals on the outer boundary. 2009.

[59] A. Sebő : Integer plane multiflows with a fixed number of demands. *Journal of Combinatorial Theory Series B*, 59(2):163–171, 1993.

[60] P.D. Seymour : Matroids and multicommodity flows. *European Journal of Combinatorics*, 2(3):257–290, 1981.

[61] P.D. Seymour : On odd cuts and plane multicommodity flows. *Proceedings of the London Mathematical Society*, 3(1):178, 1981.

[62] J. von Neumann : Discussion of a maximum problem. *John von Neumann, Collected Works*, 6:89–95.

[63] J. Vygen : NP-completeness of some edge-disjoint paths problems. *Discrete Applied Mathematics*, 61(1):83–90, 1995.

Table des figures

Résumé. L'étude des cycles, flots et chemins des graphes est intimement liée au développement de l'optimisation combinatoire. Dans l'introduction nous mettons en parallèle ces concepts à partir de résultats classiques, et les deux autres parties de la thèse développent les nouveaux résultats dans deux directions différentes.
La première porte sur les problèmes d'existence de multiflots entiers. Plusieurs paramètres naturels s'appliquent à ces problèmes, générant plus d'une centaine de cas. Après un rappel des résultats de la littérature sous une forme synthétique, nous résolvons plusieurs problèmes ouverts. En particulier, nous montrons que trouver deux flots disjoints dans les graphes planaires est un problème NP-complet. Nous donnons aussi un algorithme polynomial pour router les digraphes planaires acycliques eulériens, lorsque le nombre de classes d'arcs de demande est fixé.
Ensuite, nous nous intéressons au problème consistant à trouver une plus courte marche fermée passant par tous les sommets d'un graphe. Spécifiquement, nous cherchons à caractériser les graphes pour lesquels une bonne caractérisation est donnée par des empilements d'ensembles éclatants. Nous présentons quelques résultats de nature polyédrale, puis étudions le cas des cographes et des graphes d'intervalles.
Mots-clés. flot, multiflot entier, chemin disjoint, graphe planaire, graphe eulérien, cycle hamiltonien, ensemble éclatant, solidité.

Optimal Routings : Tours, Flows and Paths.

Abstract. The study of cycles, flows and paths in graphs is closely related to the development of combinatorial optimization. In the introduction, we explicit this relationship by using classical results, and the two next parts develops new results in two distinct directions.
First, we look at problems of existence of integer multiflow. Many parameters can naturally be applied to them. By combination they yield a hundred distinct cases. We present the state of the art in a synthetic way, and then prove some new results. We prove the NP-completeness of finding two disjoint flows in planar graphs. We also give an algorithm for solving multiflow problems in acyclic digraphs with Euler condition, when the number of classes of demand is bounded.
Then, we consider the problem of finding a shortest spanning closed walk. More precisely, when does this problem admit a good caracterisation based on packing of scattering sets ? We give some polyhedral results, and investigate the case of cographs and interval graphs.
Keywords. flow, integer multiflow, planar graph, Eulerian graph, Hamiltonian cycle, scattering set, toughness.