Conception de Réseaux Dynamiques Tolérants aux Pannes

Florian Huc

▶ **To cite this version:**

Florian Huc. Conception de Réseaux Dynamiques Tolérants aux Pannes. Réseaux et télécommunications [cs.NI]. Université Nice Sophia Antipolis, 2008. Français. NNT : . tel-00472781

HAL Id: tel-00472781
https://theses.hal.science/tel-00472781

Submitted on 13 Apr 2010

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

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

N° d'ordre: 00000

THESE

Présentée devant

devant l'Université de Nice Sophia Antipolis

pour obtenir

le grade de : Docteur de l'Université de Nice Sophia Antipolis
Mention Informatique

par

Florian Huc

Equipe d'accueil : Projet MASCOTTE
Ecole Doctorale : ED-STIC
Composante universitaire : I3S (CNRS - UNSA) INRIA

Titre de la thèse :
Conception de Réseaux Dynamiques Tolérants aux Pannes

soutenue le 14 novembre 2008 devant la commission d'examen

M. :	Jean-Claude	König	Président
MM. :	Pierre	Fraigniaud	Rapporteurs
	Arie	Koster	
	Andrew	Thomason	
MM. :	David	Coudert	Examinateurs
	Jean-Claude	König	
	Yann	Vaxès	
	Jean-Claude	Bermond	

Remerciements

Je tiens à remercier tout particulièrement Jean-Claude Bermond qui a été mon directeur de thèse ainsi que David mon (presque, à une signature près) co-directeur de thèse. Toutes les discussions que nous avons eues m'ont beaucoup appris, et vos conseils m'ont été et me seront encore très utiles. Merci aussi pour toutes les opportunités que vous m'avez ouvertes et amenées à saisir au cours de ces trois ans.

Merci également à Pierre Fraigniaud, Arie Koster et Andrew Thomasson pour avoir été rapporteurs de ma thèse, ainsi qu'a Jean-Claude König et Yann Vaxès pour avoir participé à mon jury.

Merci Patricia Lachaume, Ephie Dériche et Sandra Devauchelle pour votre gentillesse et votre aide si précieuse pour venir à bout de tous les papiers administratifs.

Un grand merci à tous les membres du projet MASCOTTE, notamment à Jean-Claude sans qui MASCOTTE serait tout autre et qui aura réussi à me faire courir un marathon relais avec une équipe formidable. Merci à David qui m'a permis d'agrandir ma collection de guides du routard et de lonely planet, à Frédéric & Frédéric pour les franches rigolades et les barbecues, à Joanna pour ses conseils et sa bonne humeur, à Stéphane pour ses idées, à Bruce pour le meilleur whisky que j'ai jamais bu et les bons moments passés, à Hervé pour ses nombreux conseils et sa disponibilité, à Michel et à l'IUT pour toutes leurs histoires. Merci également à tous les sud-américains Cristiana, Juan-Carlos, Julian, Napoleao et Patricio pour leur aide précieuse pour faire de la caipirinha et leur gentillesse. Merci aussi à Marie avec qui, à mon grand regret, je n'aurais pas eu assez de temps pour faire de l'escalade, à Nathann et sa curiosité insatiable qui m'auront fait découvrir de nombreuses choses, à Christelle et à sa tortue Cacahuète dont j'aurais tant entendu parlé, à Dorian avec qui ça a été un plaisir de travailler et qui, j'en suis désolé, a hérité d'un surnom par ma faute (mais aussi celle d'Hervé tout de même), à Judicael et aussi à Ignasi avec qui ce fut un plaisir de travailler et de voyager. Merci aussi à Marie-Emilie, Jean-Sébastien et Omid, qui ont été dans MASCOTTE mais qui en sont parti avant que je ne finisse ma thèse. Merci à tous pour la bonne ambiance qui règne au sein de l'équipe, rendant agréable le fait de venir travailler.

Merci à ma famille bien sur et à toutes ses "pièces rapportées", cela va de soi, merci d'être là. Merci à Héloïse pour les bons moments passés ensemble et tous ceux à venir.

Merci à Omid, Pooran, Alexandre et Guillaume pour avoir passé tant de temps en colloc ensemble et être devenu ma famille Antiboise.

Merci à Nicolas et Yoann pour leur amitié exceptionnelle, les fou-rires, les batailles acharnées, leur complicité et leur présence tout simplement. Merci à Xavier pour son amitié.

Merci à Yvan, Marie, Annelise (sans y) et Aurélie que la distance n'a pas éloignés.

Merci à tous mes amis de Cahors, Guillaume, Bertrand, Delphine, Virginie, Pierre et Adeline qui sont tout aussi indispensables.

Merci à Pierre, Sylvain, Sapna, Vincent, Benjamin, Nicolas, Nathalia, Damien, Demian, Nadia, Senem, Ozge, Pinar ... et tous les autres sans lesquels la vie ici n'aurait pas été pareille.

Merci à tous ceux que j'ai rencontrés pendant mes déplacements, avant ou pendant ma thèse, Simon, Nicolas et Olga tout particulièrement mais aussi Frederico, Luis, Louis, Yusuf, Brigitte et tous les autres.

Table des matières

Chapitre 1

Introduction

Le déploiement des réseaux de télécommunication assure l'interconnexion des utilisateurs où qu'ils soient dans le monde. Ce développement a lieu au travers de plusieurs types de réseaux. Les réseaux satellites permettent de relayer les communications partout dans le monde mais leur coût est élevé. Les réseaux WiFi type GSM (Groupe Spécial Mobile) sont utilisés dans les pays occidentaux dans lesquels l'utilisation des téléphones mobiles s'est banalisée et dans les pays en voie de développement car le coût des infrastructures est faible et que ces réseaux sont facile et rapide à installer. Les réseaux optiques sont utilisés dans les zones ayant un fort trafic ou pour assurer les liaisons intercontinentales, elles permettent entre autre une liaison plus rapide que par satellite. J'ai débuté ma thèse dans un contexte d'essor de ces réseaux de télécommunication suite à un ralentissement aux alentours de 2002. Les sommes engagées dans le développement de ces réseaux sont très importantes (elles se comptent en centaines de milliards d'euros rien qu'en Europe) et sont en forte croissante. Ainsi, il est crucial de construire des réseaux adaptés.

Les réseaux que j'ai cités précédemment -réseaux embarqués pour les satellites, réseaux d'antennes captant les signaux des satellites, réseaux WiFi et réseaux optiques- ont chacun leurs propres spécificités techniques, mais un grand nombre de problématiques fondamentales sont transverses, comme le dimensionnement, la tolérance aux pannes, la nécessité de pouvoir s'adapter à un environnement ou à une demande dynamique et la minimisation des ressources nécessaires à une qualité de services suffisante. La maîtrise de ces différentes problématiques est essentielle à tout opérateur qui souhaite être compétitif et la maîtrise de l'une d'elles n'est utile que conjointement à la maîtrise des autres problématiques.

Au cours de ma thèse, je me suis intéressé à ces problématiques en utilisant plusieurs approches, notamment de nature combinatoire et algorithmique. Mon objectif a été d'étudier différentes étapes du cycle de vie d'un réseau, de sa conception à son exploitation quotidienne. Cela me permet de mettre en évidence les points communs entre les différents problèmes, d'utiliser des techniques variées pour les résoudre et d'apporter une expertise originale. Les approches que j'utilise dans chacun des chapitres sont pour la plupart génériques et peuvent être adaptées à d'autres problèmes. Ceci dit, afin de

les illustrer, je les applique à un type donné de problème et de réseau (WiFi, optique ou embarqué selon les chapitres).

1.1 Principaux résultats obtenus

La majorité des résultats que j'ai obtenu au cours de ma thèse ont fait l'objet de publications. Elles figurent dans les annexes A à F aux côtés d'articles en préparation. La liste de ces travaux se trouve à la fin de ce chapitre.

Je présente ci-dessous les résultats principaux de ma thèse. Dans les chapitres 3 à 7, je présente en détail les problèmes étudiés et les résultats obtenus. Les démonstrations de ces résultats sont présentées en annexes.

Conception de réseaux ; graphes d'expansion et robustesse : Chapitre 3. Le Chapitre 3 est dédié à l'étude de la conception des réseaux. Plus précisément, je présente des constructions de réseaux destinés à être embarqués dans des satellites. Originellement, ce problème a été proposé par Alcatel Space Industrie. Le rôle des satellites en question est de relayer des signaux télévisés, ils reçoivent un certain nombre de signaux et doivent les réémettre. Les signaux sont reçus via des entrées, seulement celles-ci ne sont pas toujours fonctionnelles. Certaines peuvent être en panne ou mal orientées, il en faut donc plus que de signaux à recevoir. Les signaux sont ensuite transmis, via des guides d'onde interconnectés par des commutateurs, à des amplificateurs avant d'être réémis en direction de la terre. Comme pour les entrées, seuls certains amplificateurs sont fonctionnels, il en faut donc plus que de signaux à réémettre. Le problème est de concevoir des réseaux, capables de router les signaux arrivant par les entrées fonctionnelles vers un nombre suffisant d'amplificateurs fonctionnels. On cherche des réseaux ayant le plus petit nombre de commutateurs car ceux-ci coûtent cher à construire et à envoyer dans l'espace. Ce problème a déjà été étudié, entre autre dans la thèse de E. Darrot [], ainsi que dans de nombreux autres articles dont []. Cependant, jusqu'à présent, seul les réseaux tolérants un petit nombre de pannes ont été étudiés. Dans le Chapitre 3, je présente plus en détail ce problème ainsi que les résultats antérieurs à ma thèse. J'y présente également des constructions et des résultats théoriques permettant de calculer des bornes inférieures sur la taille des réseaux, et ce, entre autres, quand le nombre d'entrées et de sorties non fonctionnelles est logarithmique par rapport au nombre de signaux. Je présente plusieurs constructions optimales, basées sur des graphes d'expansion. Pour construire des réseaux dont le nombre d'entrées et de sorties non fonctionnelles peut être plus grand, une notion est présentée : l'α-*robustesse*. L'α-robustesse d'un graphe G traduit le fait que les petits sous-ensembles de sommets ont besoin d'être fortement connectés (proportionnellement à leur taille) au reste du graphe au cas où aucune des entrées ou sorties à l'intérieur ne soient fonctionnelles, alors que pour les plus grands sous-ensembles, le nombre d'entrées ou de sorties non fonctionnelles est borné (supérieur à une constante) par les hypothèses du problème. Je présente des résultats sur l'α-robustesse des graphes aléatoires 4-réguliers permettant de construire des réseaux quand le nombre d'entrées et de sorties non fonctionnelles est

linéaire en le nombre de signaux. Ces résultats ont donnés lieu à deux publications : [ABG+06] et [AGHP]. Cette dernière est donnée dans l'Annexe A.1.

Un autre aspect important de la conception des réseaux est le dimensionnement des liens. Cela nécessite de prévoir le trafic qui va transiter par chacun d'eux. Dans les trois chapitres suivants de ma thèse, je m'intéresse au problème de routage, qui est un problème d'allocation de ressources. En plus de permettre le dimensionnement d'un réseau, si le problème de routage est résolu efficacement, la solution proposée peut également être utilisée lors de l'exploitation du réseau.

Dimensionnement de réseau ; algorithme de routage : Chapitre 4. Dans le Chapitre 4, j'étudie trois problèmes différents : le routage de paquets dans des réseaux en forme de grille, l'établissement de connexions dans des réseaux d'antennes recevant des signaux d'un satellite et dans des réseaux radios sans fils maillés.

Concernant le routage de paquets, beaucoup de résultats sont connus. Entre autre, un résultat de T. Leighton, B. Maggs et S. Rao, publié dans [LMR88, LMR94], dit que pour tout réseau, tout ensemble de paquets, il existe un ordonnancement asymptotiquement optimal à un facteur multiplicatif près. Dans ce chapitre, je présente le problème de routage de paquets pour le modèle Δ-*port* (toutes les sorties d'un nœud peuvent être utilisées simultanément), tant dans le cas *half-duplex* (les liens ne peuvent être utilisés que dans un sens à un instant donné) que *full-duplex* (les liens peuvent être utilisés dans les deux sens à la fois). Dans l'article [AHSZ08], inclus dans l'Annexe B.1, sont présentés des algorithmes de routage optimaux en plus courts chemins pour plusieurs situations initiales (permutation, routage r-central et (ℓ,ℓ)-routage).

Concernant l'établissement de connexions dans les réseaux d'antennes recevant des signaux d'un satellite, le problème consiste à allouer des fréquences aux antennes de telle sorte que les antennes puissent capter les signaux qui leurs sont destinés, et cela tout en utilisant le moins possible de fréquences au total. Le problème de coloration classique peut être utilisé pour modéliser l'allocation de fréquence, cependant, les contraintes imposées (deux antennes adjacentes ne peuvent pas recevoir sur la même fréquence) sont plus fortes que nécessaire. En effet, pour une antenne, tant que le bruit présent sur sa fréquence de réception n'est pas trop grand, elle peut correctement interpréter le signal reçu. Ainsi, le problème d'allocation de fréquences est mieux modélisé par le problème de *coloration k-impropre* -une coloration dans laquelle un sommet a la même couleur qu'au plus k de ses voisins- qui est *NP*-dur, et ce même dans le cas qui nous intéresse : les grilles triangulaires, [HKS05, Ser06]. Il existe des algorithmes d'approximation pour la coloration propre des grilles triangulaire , [MR00], je présente des algorithmes d'approximation pour la coloration k-impropre d'un graphe. Dans les grilles triangulaires, les coefficients sont, pour $1 \leq k \leq 5$: $\frac{25}{13}$, $\frac{12}{7}$, $\frac{18}{13}$, $\frac{80}{63}$ et $\frac{49}{43}$. Ces résultats ont été publiés dans l'article [BHHL07]. Une version étendue de l'article est donnée dans l'Annexe B.2.

Dans le cas des réseaux radios sans fils maillés, j'étudie la complexité du problème d'ordonnancement des connexions radios. Nous modélisons ce nouveau problème par

un problème de coloration des arêtes d'un graphe pondéré (G, w) appelé *coloration proportionnelle*. Dans cette coloration, chaque arête e veut recevoir un nombre de couleurs dépendant (proportionnellement à son poids, $w(e)$) du nombre total de couleurs. Je présente des résultats concernant la complexité de cette coloration et des classes de graphes pour lesquelles ce problème est polynomial. Ces résultats ont été publiés dans l'article [HLR08], et une version étendue de cet article est présentée dans l'Annexe B.3.

L'optimisation des routages de requêtes *unicasts* n'est pas toujours suffisante pour exploiter au mieux un réseau. Le développement des applications multi-utilisateurs tels les jeux en réseaux ou la télévision numérique entraîne l'envoi des mêmes données à plusieurs utilisateurs. Dans ce cas, il est possible d'économiser des ressources du réseau en dupliquant les données à l'intérieur du réseau au lieu de les envoyer plusieurs fois.

Requêtes multicast ; coloration de graphes orientés et programmation linéaire : Chapitre 5. Dans le Chapitre 5, j'étudie les requêtes *multicasts*. Je commence ce chapitre en présentant les composants dont sont équipés les réseaux optiques que j'étudie. Je présente ensuite un problème de coloration appelé *directed star colouring*, permettant d'évaluer d'un point de vue théorique, le gain induit par l'usage de requêtes *multicasts* au lieu de requêtes *unicasts*. Ce problème de coloration est une version du problème de *star arboricity* introduit par I. Algor et N. Alon dans [AA89], adaptée aux graphes orientés. La *directed star arboricity* d'un graphe orienté D est le nombre minimum d'ensembles d'étoiles disjointes nécessaires pour partitionner les arcs de D, elle est notée $dst(D)$. Ces travaux poursuivent ceux de R. Brandt et T.F. Gonzalez [BG05]. , il est entre autre prouvé que $dst(D) \leq 2\Delta^-(D)+1$ et que $dst(D) \leq 2\max(\Delta^+(D), \Delta^-(D))$. L'ensemble des résultats obtenus ont été rédigés dans un article présenté dans l'Annexe C.1. Je propose ensuite différentes formulations mathématiques permettant de calculer des routages de requêtes *multicasts* dans les réseaux de type réseaux optiques de cœurs et réseaux d'accès. Un article présentant les formulations concernant les réseaux optiques d'accès est en cours, il est donné dans l'Annexe C.2.

L'optimisation de l'utilisation des ressources d'un réseau n'est pas le seul critère à prendre en compte lors de l'exploitation d'un réseau. En effet, un client n'est pas intéressé par le profit réalisé par l'exploitant mais par la qualité du service qui lui est fourni.

Qualité de service, groupes de risques et protection ; programmation linéaire : Chapitre 6 Le problème de pannes multiple, ou groupes de risques, a récemment été étudié, par exemple dans la thèse de M-E. Voge. Dans le Chapitre 6, je présente ce problème dans les réseaux optiques et je propose une formulation mathématique permettant de calculer des routages de requêtes *unicasts* tout en prenant en compte ces pannes. Je présente également des résultats expérimentaux sur le temps

de calcul de cette formulation. Ces travaux ont donné lieu à l'article [CHPV08] qui est donné en Annexe D.1. Je présente ensuite deux modes de protection par segments dans les réseaux optiques WDM (*Wavelength Division Multiplexing*) utilisant le groupage de trafic. Leur principe est de protéger un ensembles d'arcs (appelés segments) simultanément. Le but des modes de protection présentés est de se baser sur le routage existant des requêtes pour minimiser le coût de la protection. Un article sur ce problème est en cours de préparation, il est présenté dans l'Annexe D.2.

Reroutage ; décomposition de graphes : Chapitre 7. Dans le Chapitre 7, je m'intéresse à la gestion dynamique des réseaux. Étant donné un ensemble de demandes et une attribution des ressources qui répond à ces demandes, lors de l'évolution des demandes -ajout et suppression- certaines peuvent ne pas être satisfaites alors que si les ressources étaient attribuées différemment, cela serait possible. Une solution consiste à modifier l'attribution des ressources pour arriver à cette situation sans perturber trop de demandes. Le *process number* est un paramètre qui a été introduit pour modéliser ce type de reconfiguration de réseau. Dans [**?**], les auteurs présentent ce paramètres ainsi que des algorithmes pour le calculer s'il est inférieur à trois. Dans ce chapitre, je rappelle les liens avec différents paramètres dont la *pathwidth* et je donne un aperçu des résultats connus. Ensuite, je présente un algorithme dynamique permettant de calculer le *process number* d'un arbre en temps optimal. J'expose également de nouveaux résultats théoriques sur les liens entre la *pathwidth* des graphes planaires extérieurs et celle de leur dual, permettant de répondre à plusieurs conjectures de H. Bodlaender et F. Fomin. Entre autres, pour tout graphe planaire extérieur G, $pw(G^*) \leq pw(G) \leq 2pw(G^*) - 1$. Ce résultat donne un algorithme de 2-approximation pour le calcul de la *pathwidth* des graphes planaires extérieurs. Je présente ensuite des avancées sur des conjectures sur les liens entre la *pathwidth* des graphes planaires et celle de leurs duaux, en particulier, pour tout graphe planaire 3-connexe G, $pw(G^*)/3 - 2 \leq pw(G) \leq 3pw(G^*) + 2$. Je finis ce chapitre avec des résultats concernant l'indice d'échappement arête connexe des arbres. Les résultats présentés dans ce chapitre ont donnés lieu à quatre publications : [CHS07, AHP, CHM08a, CHM08c]. Dans l'Annexe E.1, figure une version étendue des articles [CHM08a, CHM08c]. Les publications [CHS07, AHP] sont données dans les Annexes E.2 et E.3.

Structure des graphe orientés : Annexe F. Dans l'Annexe F, j'introduis un nouveau paramètre concernant les graphe orientés, appelé *mixing*. Un graphe orienté D est dit f-mixing si pour toute paire d'ensembles A et B avec plus de f arcs de A vers B, il y a au moins $f/2$ arcs de B vers A. Certaines conditions, sur ce paramètre dans un graphe orienté D, assurent l'existence de schémas, décrit par un autre (petit) graphe orienté H. Ce paramètre a un intérêt théorique car il permet de mieux comprendre la structure des graphe orientés. Le résultat principal de cet article est que la plus grande valeur de f telle que tout graphe orienté f-mixing contient un circuit de longueur quatre est e^2/n^2. Je présente également des bornes inférieures et supérieures pour des cycles orientés plus longs. Le problème est complètement résolu dans le cas où le graphe orienté D est dense.

Enfin, je présente des résultats sur la valeur minimale de ce paramètre dans les graphes orientés réguliers. Une partie de ces résultats ont fait l'objet de l'article [AGH08].

1.2 Réseaux de télécommunication : problématiques et techniques

Les problématiques que j'ai dégagées, concernant le cycle de vie d'un réseau, m'ont permis de mettre en œuvre des techniques variées dont certaines sont transverses. Dans cette section, je décris leurs utilités.

Graphes d'expansion et structure des graphes. Le rôle des réseaux étant d'interconnecter des utilisateurs entre eux, ceux-ci doivent avoir de bonnes propriétés de connectivité. Dans les cas où la topologie du réseau peut être choisie, l'utilisation de graphes d'expansion (des graphes dont tout ensemble de moins de la moitié des sommets a un nombre de voisins proportionnel à sa taille) comme modèle peut s'avérer intéressante, tout comme celle de graphes ayant une bonne robustesse si le besoin en connectivité est moindre. Des techniques de décomposition de graphes peuvent également être utiles pour mettre en exergue de petites structures interdites et ainsi obtenir des garanties d'optimalité pour les réseaux conçus ; la q-*quasi partition* (une répartition des sommets en ensembles connexes de taille à peu près q et tels que peu de sommets soient dans plusieurs ensembles) en est un exemple. D'autres types de décompositions sont utilisés pour la conception de réseaux, telle la *vertex separation* (une énumération des sommets telle que tous segments initiaux aient peu de voisins, un paramètre équivalent à la *pathwidth*) qui a été introduite pour la conception de circuits. Enfin, l'utilisation de méthodes probabilistes permet d'obtenir des garanties théoriques sur l'existence de "bons" réseaux.

Problèmes de coloration de graphes, méthode probabiliste et programmation linéaire. Le problème de dimensionnement de lien est complémentaire au problème de choix de la structure. Il s'agit d'assurer que les capacités du réseau permettent de satisfaire les demandes prévues. Il s'agit donc de garantir la répartition de ressources disponibles entre différentes requêtes (ce sont typiquement des problèmes de routage et/ou d'ordonnancement) de telle sorte qu'une ressource ne soit pas simultanément utilisée par plusieurs requêtes. De tels problèmes sont facilement modélisables par des problèmes de coloration de graphes. Les couleurs représentent alors les ressources, les sommets du graphe représentent les requêtes et une arête représente une interaction entre deux requêtes. Les problèmes de coloration que j'étudie sont la coloration pondérée k-impropre, la coloration proportionnelle, la *coloration fractionnaire* -une relaxation de la coloration normale- et le *directed star colouring*. Le problème est alors de trouver le nombre de couleurs minimum pour colorier les graphes d'une classe de graphes bien précise. Lors du dimensionnement des réseaux, l'important est l'existence de solution, il est donc pertinent d'utiliser des techniques probabilistes telle que le Lemme Local de Lovász, duquel il existe une version déterministe. Une approche complémentaire pour

résoudre des problèmes d'allocation de ressources est l'usage de programmes linéaires. Quand un problème est trop complexe pour être modélisé de manière suffisamment simple pour pouvoir le résoudre de manière théorique, l'utilisation de programmes linéaires permet, à l'aide de simulateurs et d'un solveur, d'obtenir des résultats empiriques sur le comportement du réseau que l'on conçoit. Au cours de ma thèse, j'ai conçu des programmes linéaires pour des problèmes de routage et de protection. Les objectifs à considérer sont variés, il est nécessaire de prendre en compte le profit réalisé ainsi que la qualité du service offert aux clients. J'ai utilisé la technique dite de génération de colonnes afin de pouvoir résoudre des problèmes de grande taille.

Décomposition de graphes et algorithmes d'approximation. Lors de la conception d'un réseau, le temps de calcul n'est pas un facteur limitant. Cependant, lors de son exploitation, des algorithmes d'optimisation efficaces sont indispensables afin de pouvoir faire face aux évolutions dynamiques des demandes et des caractéristiques. Dans des cas où le problème peut être modélisé par un graphe de petite *pathwidth* ou *treewidth* (deux types de description de séparateurs de petite tailles), il est parfois possible de calculer la solution optimale. Sinon cela prend trop de temps car les problèmes sous-jacents sont souvent NP-durs. Ainsi, il est nécessaire de développer des algorithmes d'approximation et si possible distribués pour faciliter l'exploitation des réseaux. Les problèmes de coloration se prêtent souvent bien à la conception de tels algorithmes. Si les programmes linéaires sont suffisamment vite résolus, ils peuvent également être utilisés lors du routage. Une autre solution pour exploiter un réseau dont les caractéristiques ou les demandes évoluent dynamiquement est de se baser sur une solution courante et de la faire évoluer. Ce problème a été modélisé par un paramètre appelé *process number*. Il se trouve que ce paramètre est étroitement lié à la *pathwidth*, l'étude de ce paramètre est donc intéressante tant d'un point de vue pratique que théorique.

Ces différents problèmes permettent d'illustrer la diversité des applications des outils combinatoires et algorithmiques que j'ai utilisés au cours de ma thèse. Leurs utilisations sont complémentaires, ce qui en renforce l'intérêt.

1.3 Publications

Journaux :

[AHP] O. Amini, , F. Huc, and S. Pérennes. On the pathwidth of planar graphs. Research repport : https ://hal.inria.fr/inria-00082035, to appear in SIAM, journal of discrete maths.

[AGHP] O. Amini, F. Giroire, F. Huc, and S. Pérennes. Minimal selectors and fault tolerant networks. Research report : http ://hal.inria.fr/inria-00082015, to appear in Networks.

[CHS07] D. Coudert, F. Huc, and J.-S. Sereni. Pathwidth of outerplanar graphs. *Journal of Graph Theory*, 55(1) :27–41, May 2007.

Conférences internationales :

[AGH08] O. Amini, S. Griffiths, and F. Huc. 4-cycles in mixing digraphs. In *The IV Latin-American Algorithms, Graphs, and Optimization Symposium (LAGOS 07)*, Puerto Varas, Chile, November 2007. Electronic Notes in Discrete Mathematics, Volume 30, 20 February 2008, pages 63-68.

[CHM08c] D. Coudert, F. Huc, D. Mazauric. A distributed algorithm for computing and updating the process number of a forest. In *DISC 2008*, *brief announcement*, Arcachon, France, September 2008.

[CHPV08] D. Coudert, F. Huc, F. Peix, and M.E. Voge. Reliability of Connections in Multilayer Networks under Shared Risk Groups and Costs Constraints. In *ICC 2008*, pages 5170-5174, Beijing, China, May 2008.

[HLR08] F. Huc, C. Linhares, and H. Rivano. The proportional colouring problem : Optimizing buffers in radio mesh networks. In *The IV Latin-American Algorithms, Graphs, and Optimization Symposium (LAGOS 07)*, Puerto Varas, Chile, November 2007. Electronic Notes in Discrete Mathematics, Volume 30, 20 February 2008, pages 141-146.

Conférences nationales :

[ABG+06] O. Amini, J.-C. Bermond, F. Giroire, F. Huc, and S. Pérennes. Design of minimal fault tolerant networks : Asymptotic bounds. In *Huitièmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel'06)*, pages 37–40, Trégastel, France, May 2006.

[BHHL07] J-C Bermond, F. Havet, F. Huc, and C. Linhares. Allocation de fréquence et coloration impropre des graphes hexagonaux pondérés. In *9èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel'07)*, pages 53-56, Ile d'Oléron, May 2007.

[CHM08a] D. Coudert, F. Huc, and D. Mazauric. Algorithme générique pour les jeux de capture dans les arbres. In *10èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel'08)*, pages 37–40, Saint Malo, France, May 2008.

Soumis :

[AHHT] O. Amini, F. Havet, F. Huc, and S. Thomassé. WDM and Directed Star Arboricity. Submitted to Combinatorics, Probability and Computing.

[AHSZ08] O. Amini, F. Huc, I. Sau Valls, and J. Zerovnik. Optimal (l,k)-routing on plane grids. Technical Report : http ://hal.inria.fr/inria-00265297, submitted to Journal of Interconnection Networks.

[HLR] F. Huc, C. Linhares, and H. Rivano. The proportional colouring problem : Optimizing buffers in radio mesh networks. Submited to a special issue of Discrete Applied Mathematics.

Chapitre 2

Notions de base

Dans ce chapitre, je présente les définitions dont je vais avoir besoin au cours de ma thèse. Ensuite, je présente le fonctionnement d'un réseau et le problème de routage.

2.1 Définitions et notations

La seule prétention de cette section est d'introduire les notations que je vais utiliser dans ma thèse et quelques définitions qui pourraient ne pas être standard. Il est préférable de la consulter quand cela est nécessaire plutôt que de la lire en détail.

2.1.1 Graphes

Un **graphe** G est un couple d'ensembles (V,E) où V est un ensemble d'éléments appelés **sommets** de G et E un ensemble de paires de sommets distincts de V. Les éléments de E sont appelés **arêtes** de G. De manière générale $|V| = n$ et $|E| = e$. On utilisera $V = \{v_1, \ldots, v_n\}$ comme ensemble de sommets. Dans certains cas des graphes avec des boucles sont considérés, une **boucle** étant une arête d'un sommet vers lui même, c.à.d. une paire contenant deux fois le même sommet. Un graphe avec des arêtes multiples (plusieurs fois une même paire de sommets dans E) est un **multigraphe**.

Un **graphe orienté** D est un graphe dans lequel chaque arête (appelée **arc** dans ce contexte) est un *couple de sommets* au lieu d'être une paire de sommets. La différence entre un couple et une paire est que dans un couple les éléments sont ordonnées. Ainsi un arc d'un graphe orienté a une source et une destination. Un **graphe orienté asymétrique** est un graphe orienté dans lequel, s'il y a un arc uv, il ne peut pas y avoir l'arc vu. Un graphe **graphe orienté symétrique** est un graphe orienté dans lequel, s'il y a un arc uv, il y a aussi l'arc vu. Un **multigraphe orienté** est un graphe orienté dans lequel un arc peut être répété.

Remarque 2.1 Si dans un contexte où il est question de graphe orienté, pour désigner un graphe, "**graphe (simple)**" sera utilisé pour éviter toute ambiguïté.

Je note $\overline{D}$ le **graphe sous-jacent** (underlying graph en anglais) d'un graphe orienté (respectivement graphe orienté asymétrique ou multigraphe orienté). C'est le graphe

(respectivement multigraphe) obtenu en considérant les arcs comme des arêtes, c.à.d. les couples formant E comme des paires.

2.1.2 Caractéristiques des graphes

Étant donné un graphe G, le **voisinage** d'un sommet v de G, $\Gamma(v)$, est l'ensemble des sommets u tels que uv soit une arête du graphe. Le **voisinage sortant** (respectivement **voisinage entrant**) d'un sommet v de G, $\Gamma^+(v)$ (respectivement $\Gamma^-(v)$), est l'ensemble des sommets u tels que vu (respectivement uv) soit un arc.

Étant donné un graphe G, le **degré**, $d(v)$, d'un sommet v de G est le nombre d'arêtes contenant v. Étant donné un graphe orienté D, le **degré sortant** $d^+(v)$ (respectivement **degré entrant** $d^-(v)$) d'un sommet v de D est le nombre d'arcs ayant v pour source (respectivement destination) $d^+(v) = |\Gamma^+(v)|$, $d^-(v) = |\Gamma^-(v)|$.

Le **degré max** de G $\Delta(G)$, est le maximum des degrés des sommets du graphe. Le **degré sortant max** de D, $\Delta^+(D)$, est le maximum des degrés sortants des sommets du graphe orienté. Le **degré entrant max** de D, $\Delta^-(D)$, est le maximum des degrés entrants des sommets du graphe orienté. Le **degré total** de D (respectivement d'un sommet de D) est $\Delta(\overline{D})$ (respectivement $d^+(v) + d^-(v)$).

2.1.3 Quelques familles de graphes

Sous-graphe. Étant donné un graphe G, un graphe H ayant k sommets est un sous-graphe de G si il existe k sommets de G auxquels peuvent être associés les sommets de H de telle sorte que toutes les arêtes de H soient des arêtes de G. On parle de sous-graphe induit si les k sommets forment exactement H (c.à.d. qu'il n'y a pas d'arêtes en trop).

Graphe biparti. Un graphe est biparti si et seulement si ses sommets peuvent être partitionnés en deux ensembles V_1 et V_2 tels que $E \subset V_1 \times V_2$.

Graphe de Turan $T(n,r)$. Le graphe de Turan $T(n,r)$ est le graphe obtenu en partitionnant les n sommets en r ensembles de taille égale (à un près) et en formant une arête pour toute paire de sommets si ils n'appartiennent pas au même ensemble de la partition.

Graphe planaire. Un graphe est planaire si il peut être dessiné sur le plan sans que ses arêtes (représentées par des courbes continues) se croisent. Une **face** F est une région connexe maximale du plan délimitée par un ensemble d'arêtes et qui n'en contient aucune. On remarquera qu'une des faces n'est pas bornée.

Le **dual** d'un graphe planaire G (dessiné sur le plan) est le graphe dont les sommets sont les faces de G et les arêtes sont les paires de faces adjacentes à une même arête.

Le **dual faible** est le dual auquel a été enlevé le sommet correspondant à la face non bornée et les arêtes incidentes à ce sommet.

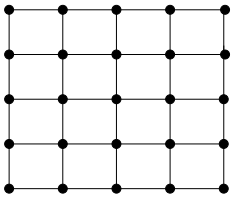
Graphe planaire extérieur. Un graphe est planaire extérieur si il peut être dessiné sur le plan sans que ses arêtes se croisent et que tous les sommets soient sur la face non bornée. Il s'agit donc d'un anneau avec des cordes ne s'intersectant pas.

Graphe k-régulier. Un graphe k-régulier est un graphe dans lequel tous les sommets ont degré k.

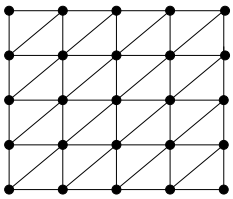
Arbre binaire complet. Un arbre binaire est un graphe acyclique dont les sommets ont degré 1 (les feuilles) ou 3 excepté un sommet qui a degré 2 (la racine).

Arbre complet. Un arbre complet est un arbre dont les feuilles sont toutes à la même distance de la racine.

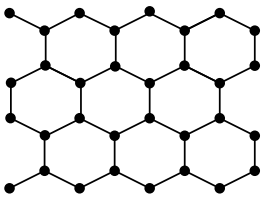
Grille carrée. Une grille carré de côté k est un sous-graphe induit, à k^2 sommets, de la grille infinie ayant pour sommets les points de Z^2 et dont les arêtes sont $((i,i),(i,i+1))$ et $((i,i),(i+1,i))$, pour i dans Z, comme représenté ci-dessous :



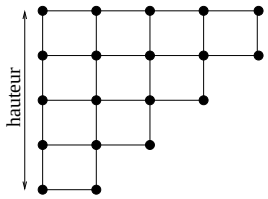
Grille triangulaire. Une grille triangulaire est obtenue à partir d'une grille carrée à laquelle a été rajoutés les diagonales, comme représenté sur la Figure ci-dessous. La grille triangulaire a côté k si la grille carré dont elle a été obtenu a côté k.



Grille hexagonale. Une grille hexagonale est définie comme étant le dual d'une grille triangulaire. Une grille hexagonale de côté k est représentée sur la Figure ci-dessus :



Pyramide. Une Pyramide de hauteur k est obtenue à partir de la grille carrée de côté k dans laquelle on a identifié les sommets (i, j) et $(j-1, i+1)$ pour $k \geq i \geq 1$ et $k \geq j \geq 1$.



2.2 Introduction sur les réseaux

Dans cette section, je décris brièvement le fonctionnement d'un réseau pour pouvoir préciser à quel niveau se situent les travaux que je présente dans les chapitres suivant.

2.2.1 Fonctionnement général d'un réseau, le modèle OSI, Open Systems Interconnection

La norme ISO 7498, datant de 1984, décrit les principes généraux du fonctionnement des communications entre différents systèmes reliés par un réseau. Le but de cette norme n'est pas de donner des solutions techniques mais seulement de décrire l'architecture des communications. Pour cela, la norme utilise principalement deux notions :

– la notion de service,
– la notion de protocole.

Un **service** est ce que l'on peut faire ou ce que l'on peut demander aux autres, le "on" et le "autres" étant volontairement laissé indéfini car ils dépendent du contexte dans lequel le mot service est employé.

Un **protocole** est un ensemble de règles et de formats permettant le déroulement des échanges lors de la réalisation de services dans un réseau distribué. Par exemple dans un réseau, c'est le protocole utilisé qui indique si lors d'un envoi de données le système destinataire doit indiquer à l'expéditeur si il a reçu les données.

A l'aide de ces deux notions, la norme décrit un modèle de communications en sept couches : de la couche 1 usuellement désignée comme la couche la plus basse à la couche 7, la couche la plus haute. Les sept couches sont les suivantes :

1. La couche *physique* est chargée de transmettre les signaux physiques entre les différents systèmes. Les services correspondant à cette couche sont l'émission et la réception de bits.
2. La couche *liaison de données* gère les communications entre deux systèmes directement reliés par un support physique. Elle manipule des paquets de bits appelés trames, contrôle leur synchronisation et détecte les erreurs.
3. La couche *réseau* gère les communications entre deux systèmes, elle gère les communications d'un système source à un système destination. Cela inclut le routage, l'adressage des paquets et leur séquencement.

4. La couche *transport* gère les communications entre les processus, c.à.d. le transport des données d'une application à une autre. Elle gère, entre autre, les erreurs de transmissions.
5. La couche *session* gère la synchronisation des échanges au travers de l'ouverture, du maintien et de la fermeture de connexions.
6. La couche *présentation*, code les données en chaînes d'octets pour qu'elles soient transmises. Ainsi elle est responsable de la syntaxe des informations transmises et assure l'indépendance des données par rapport au format utilisé par le réseau.
7. La couche *application* est l'interface avec l'utilisateur.

La norme prévoit qu'une couche i ne puisse communiquer qu'avec la couche qui lui est immédiatement au-dessus, la couche $i+1$ ou avec la couche qui lui est immédiatement au-dessous, la couche $i-1$. Le déroulement d'une telle communication est décrit par le protocole et les informations nécessaires à cette communication sont ajoutées aux données transmises sous la forme d'un en-tête.

Les sept couches sont divisées en deux grandes catégories : les couches d'information (les trois couches supérieures), et les couches de données (les quatre couches inférieures). Ces dernières traitent les données de façon brute, c'est-à-dire sans tenir compte de leur signification. Elles se chargent simplement de faire circuler les bits et les paquets de bits.

Bien que cette norme ne soit pas appliquée stricto sensu, elle me permet d'illustrer à quel niveau se situe mon travail. En l'occurrence, je me place principalement au niveau des couches de données, et entre autre de la couche 3 : la couche réseau. Ainsi, dans les problèmes que je présente dans les chapitres suivants, je considère un ensemble de requêtes caractérisées par leurs sources et leurs destinations et l'objectif est de trouver un routage ; un **routage** étant un ensemble de chemins permettant d'acheminer les messages de leurs émetteurs à leurs destinataires tout en respectant les contraintes physiques du réseau.

Ce problème a déjà été largement étudié dans son cas le plus général, la section suivante rappelle les principaux résultats.

2.2.2 Rappels sur le multiflot classique

De manière générale un réseau est modélisé par un graphe orienté $D = (V, E)$, les sommets du graphe représentent les différents systèmes présents dans le réseau et les arcs les liens qui existent entre ces différents systèmes, la direction de l'arc indique dans quel sens le lien peut être utilisé. Selon la couche à laquelle le problème se situe, les liens peuvent être soit réels soit virtuels. A chaque arc $e \in E$ est associé une capacité $c(e)$ qui représente la quantité d'information pouvant circuler sur le lien correspondant et un prix par unité de flot p_e, le coût induit pour y faire transiter une unité de flot. Ce coût peut avoir plusieurs significations comme un coût financier ou un taux d'atténuation.

Le problème de routage consiste à trouver des chemins, en tenant compte des contraintes de capacité du réseau, permettant de faire transiter des paquets à l'intérieur du réseau. De manière plus formelle, étant donné un ensemble de requêtes K,

chaque requête $k \in K$ étant définie par une source s_k, une destination t_k et une taille d_k, le but du problème est de trouver un ensemble de chemins reliant les sources aux destinations sans que la capacité des arêtes utilisées ne soit dépassée. Deux objectifs peuvent être considérés, il peut être de maximiser le nombre de requêtes routées, ou, si il est possible de router toutes les requêtes, l'objectif peut être de minimiser le coût du routage.

Ce problème est souvent illustré par un problème de tuyauterie dans lequel une requêtes k représentent un débit d_k (on se place dans un régime stationnaire) d'eau devant aller de s_k à t_k. Sous cette forme il peut être facilement formulé par un ensemble d'équations linéaires. Pour cela, étant donnée une requête k, l'ensemble $\mathcal{P}_k$, des chemins P_k pouvant acheminer l'eau correspondant à la requête k, est introduit. A chacun de ces chemins est associée une variable χ_{P_k} indiquant la quantité d'eau de la requête k transmise sur ce chemin.

$$\max \sum_{k \in K} \sum_{P_k} \chi_{P_k}$$

$$\sum_{k \in K} \sum_{P_k : e \in P_k} \chi_{P_k} \leq d_k \qquad \forall k \in K \tag{2.1}$$

$$\sum_{k \in K} \sum_{P_k : e \in P_k} \chi_{P_k} \leq c(e) \qquad \forall e \in E \tag{2.2}$$

$$\min \sum_{k \in K} \sum_{P_k} \chi_{P_k} \sum_{e \in P_k} p_e$$

$$\sum_{k \in K} \sum_{P_k : e \in P_k} \chi_{P_k} \geq d_k \qquad \forall k \in K \tag{2.3}$$

$$\sum_{k \in K} \sum_{P_k : e \in P_k} \chi_{P_k} \leq c(e) \qquad \forall e \in E \tag{2.4}$$

Seulement, dans ces formulations, appelées arc-chemin, il est nécessaire de connaître ou de calculer a priori l'ensemble des chemins que nos requêtes peuvent utiliser et il se trouve qu'il y en a un nombre exponentiel, ce qui signifie que cet ensemble d'équations est dur à résoudre directement.

Cependant il est possible de transformer cette formulation en une formulation sommets-arcs. Dans cette nouvelle formulation il n'y a plus une vision globale du chemin, au lieu de cela, chaque nœud du réseau vérifie que l'eau qui y entre en sort excepté pour les sources et les destinations des requêtes qui sont comme des robinets et des trous. Pour modéliser la progression de l'eau correspondant à une requête k, une variable positive χ_k^e est introduite pour chaque arc e du réseau. Ce qui donne :

$$\max \sum_{k \in K} \sum_{e \in \Gamma^+ s_k} \chi_k^e$$

$$\sum_{e\in\Gamma^+(s_k)} \chi_k^e = \sum_{e\in\Gamma^-(t_k)} \chi_k^e \leq d_k \qquad \forall k \in K \tag{2.5}$$

$$\sum_{e\in\Gamma^+(v)} \chi_k^e = \sum_{e\in\Gamma^-(v)} \qquad \forall v \in V, k \in K \tag{2.6}$$

$$\sum_{k\in K} \chi_k^e \leq c(e) \qquad \forall e \in E \tag{2.7}$$

$$\min \sum_{k\in K} \sum_{e} \chi_k^e p_e$$

$$\sum_{e\in\Gamma^+(s_k)} \chi_k^e = \sum_{e\in\Gamma^-(t_k)} \chi_k^e \geq d_k \qquad \forall k \in K \tag{2.8}$$

$$\sum_{e\in\Gamma^+(v)} \chi_k^e = \sum_{e\in\Gamma^-(v)} \qquad \forall v \in V, k \in K \tag{2.9}$$

$$\sum_{k\in K} \chi_k^e \leq c(e) \qquad \forall e \in E \tag{2.10}$$

Dans ces formulations, le nombre d'équations ainsi que le nombre de variables est polynomial, il est donc possible de résoudre le problème fractionnaire en temps polynomial. Seulement, dans le cas fractionnaire, l'eau d'une même requête peut utiliser un nombre de chemins quelconques. Si pour de l'eau cela ne pose a priori pas de problème, pour les données cela est plus problématique. La plupart du temps il est supposé qu'une requête ne peut pas être partitionnée indéfiniment, c.à.d. qu'il existe une taille minimale pour les paquets partitionnant une requête. Cela revient à imposer que les variables χ_k^e soient des variables entières, le problème est alors dit *entier*. Seulement, une relaxation de ce problème dans laquelle les équations (2.1) sont ignorées -qui est appelé le problème de multiflot entier- est NP-complet. Il existe tout de même des algorithmes d'approximation qui se basent sur la solution de sa version fractionnaire. Dans [GVY93] les auteurs donnent un algorithme avec un facteur d'approximation en $O(\log(|K|))$ du multiflot entier. Un autre algorithme se basant sur l'arrondi aléatoire du multiflot fractionnaire a été proposé par P. Raghavan [Rag88] et amélioré de manière empirique par C. Coudert et H. Rivano dans [CR02]. Dans l'amélioration, il est nécessaire de résoudre plusieurs multiflots fractionnaires, ainsi même si ce problème est polynomial il est intéressant d'utiliser de bons algorithmes d'approximation, surtout que la solution exacte n'est pas nécessaire. Dans [Fle00], L. Fleischer propose un algorithme d'approximation à un facteur ε, cet algorithme d'approximation a par la suite été repris et amélioré de manière empirique par H. Rivano, D. Coudert et X. Roche dans [CRR03]. [CLR05] est une étude sur le problème du multiflot qui en présente les variantes.

Un autre problème se pose quant à la pertinence de la solution trouvée, en effet dans le cas où c'est de l'eau qui circule dans des tuyaux, cela est bien égal si de l'eau venant de deux sources différentes se mélange. Dans le cas des données, cela n'est pas aussi facile et afin de contrôler leur flux, il est nécessaire que les nœuds du réseau où

cela a lieu soient équipés de manière adéquate. Ainsi avant d'écrire nos modèles, les hypothèses faites sur les équipements présents à chacun des nœuds seront présentées.

Chapitre 3

Conception de réseaux tolérants aux pannes

Dans ce chapitre, je présente un problème de **conception de réseau**. L'objectif est de concevoir des réseaux aussi petits que possible capables de router des signaux provenant de ports d'entrées vers des ports de sorties. Seulement, les entrées et sorties utilisées varient. Le rôle du réseau est, pour un nombre de signaux donnés, d'assurer l'existence d'un routage quelles que soient les entrées et sorties utilisées. Je commence par donner la motivation du problème et faire un rappel des principaux résultats connus avant de présenter des constructions pour différentes variantes du problème ainsi que des bornes inférieures sur la taille des réseaux.

Le problème que je présente dans cette section a été posé par Alcatel Space Industrie dans les années 1995 pour des satellites destinés à des transmissions télévisuelles comme les satellites des séries Eutelsat et Astra, et reste d'actualité. Le rôle de ces satellites est de recevoir un signal vidéo et de le réémettre en direction de la terre. Le problème concerne des réseaux destinés à ces satellites. Le but de ces réseaux est de connecter certaines de ses entrées (sur lesquelles le satellite reçoit des signaux) à des amplificateurs (pour réémettre les signaux reçus). Les connexions se font à l'aide de la technologie "Traveling Wave Tube Amplifiers" TWTA [EH78]. Par commodité, j'appelle les amplificateurs des **sorties** dans le reste de la section. À l'intérieur du satellite, les signaux sont sous la forme d'ondes, ainsi les ports d'entrées sont reliés aux sorties au travers d'un réseau de guides d'onde. De plus les amplificateurs ne sont pas équipés de composants permettant de séparer deux ondes différentes. Ainsi, les connexions entre les entrées et les sorties doivent s'établir via des chemins disjoints en terme de guides d'onde. Enfin, pour des raisons d'ingénierie, les guides d'onde sont interconnectés par des **commutateurs** à quatre entrées, chacun pesant 100g, ce qui est non négligeable

dans le poids total du satellite à envoyer dans l'espace. Les connexions qu'un commutateur peut réaliser sont représentées sur la Figure 3.1.

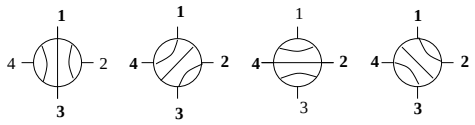


FIG. 3.1 – Un commutateur possède quatre états différents.

Au cours de la durée de vie d'un satellite, des commutateurs peuvent se bloquer dans une position donnée, limitant ainsi les connexions réalisables. De plus, des entrées peuvent être inutilisables si le satellite est mal orienté et des sorties peuvent tomber en panne. Dans tous les cas il est impossible de les réparer. Il est donc indispensable de mettre plus de ports d'entrées et de sorties que de signaux à router. Deux variantes de ce problème existent, dans la première variante, introduite par E. Darrot [Dar97] au cours de sa thèse, tous les signaux à router ont la même priorité ; dans la deuxième variante, certains signaux sont plus importants que d'autres. De tels signaux sont appelés des signaux prioritaires et ils doivent être routés vers les amplificateurs assurant le meilleur service. Les travaux que je présente ont été réalisés dans le cas où tous les signaux sont équivalents mais il est important de noter que des travaux ont été effectués sur la deuxième problématique. Ainsi, dans [Hav06, BHT06], les auteurs proposent, pour un nombre de pannes maximum en sortie donné et un nombre de signaux prioritaires donné, des réseaux de faible taille en terme de nombre de commutateurs. Ils proposent aussi pour certains cas pratiques (jusqu'à quatre signaux prioritaires et deux pannes en sorties) des réseaux de taille minimale. En effet, il est crucial de concevoir des réseaux ayant aussi peu de commutateurs que possible afin de réduire le coût des satellites. Le coût total induit par un commutateur (construction et lancement) était estimé à 15 000 €.

Il existe d'autre contraintes qui peuvent être prises en compte comme la distance parcourue par un signal, que ce soit en terme de nombre de commutateurs traversés ou en terme de longueur physique. Dans ce dernier cas, la représentation du réseau devient une contrainte. Le nombre de commutateurs à actionner lorsqu'une entrée ou une sortie tombe en panne influence l'énergie utilisée par le satellite, il peut donc aussi être intéressant de le rajouter comme contrainte. Cependant le travail que je présente s'inscrit dans la continuité des travaux déjà réalisés dans la thèse de E. Darrot [Dar97] et dans [BDD02], ainsi l'objectif est de minimiser le nombre de commutateurs.

C'est de cette contrainte sur la taille du réseau que provient la difficulté du problème. En effet, si la taille n'avait pas été une contrainte il aurait été possible de construire des réseaux à l'aide de graphes qui ont déjà largement été étudiés : les concentrateurs et les superconcentrateurs. Avant d'expliquer comment ils peuvent être utilisés pour notre problème, je vais donner leur définition ainsi que celle des graphes d'expansion, un autre type de graphes qui sera utile. Ensuite, je formaliserai le problème que je présente.

3.1 Concentrateurs, superconcentrateurs et graphes d'expansion

Une introduction sur les **concentrateurs**, les **superconcentrateurs** et les **graphes d'expansion** peut être trouvée dans [DN01]. Les définitions et les différents résultats connus y sont donnés ainsi que des questions ouvertes. Je rappelle ci-dessous les définitions ainsi que les principaux résultats dont j'ai besoin.

Les concentrateurs ont été introduits par M. Pinsker dans [Pin73] dans le cadre des réseaux de téléphone. Un $(p+k,p)$**-concentrateur** est un graphe orienté acyclique ayant $p+k$ entrées et p sorties, tel que pour tout ensemble de p entrées il existe p chemins arcs disjoints reliant les p entrées sélectionnées aux p sorties.

Un (p,p)**-superconcentrateur**, dont la définition a été introduite par A.V. Aho, J.E. Hopcroft et J.D. Ullman dans [AHU74], est aussi un graphe orienté acyclique. Les superconcentrateurs ont été introduits dans l'espoir d'établir des bornes non linéaires sur la complexité des circuits calculant des fonctions booléennes. Un (p,p)-superconcentrateur a p entrées et p sorties ; pour tout ensemble de $i \leq p$ entrées et i sorties, il existe i chemins arcs disjoints reliant les entrées sélectionnées aux sorties sélectionnées.

Pour les concentrateurs comme pour les superconcentrateurs, l'objectif est d'en construire avec le moins d'arcs possible.

Dans [Val75] L.G. Valiant montre pour la première fois qu'il existe des superconcentrateurs de taille linéaire (il montre qu'il en existe ayant de l'ordre de $238n$ arêtes), contredisant ainsi plusieurs conjectures de [AHU74]. Le lecteur intéressé pourra aussi se rapporter au travail de N. Pippenger [Pip77] dans lequel l'auteur montre l'existence de (n,n)-superconcentrateurs ayant $39n+O(\log n)$ arêtes, profondeur $O(\log n)$, et degré maximum (entrant et sortant) 16. Ce résultat a par la suite été amélioré plusieurs fois. Ainsi, U. Schöning montre l'existence de superconcentrateurs ayant de l'ordre de $28n$ arêtes dans [Sch06], ce résultat est pour le moment le meilleur connu. Cependant ces articles ne donnent pas de preuve constructive, pour une construction explicite d'un superconcentrateur, il faudra se référer à [AC03] dans lequel les auteurs présentent les plus petits superconcentrateurs explicites connus qui ont $44n+o(1)$ arêtes. D'autres constructions peuvent être trouvées dans [AM84, AGM87]. D'autre part, la meilleure borne inférieure connue sur le nombre d'arêtes d'un superconcentrateur est due à G. Lev et L.G. Valiant dans [LV83] et est de $(5-o(1))n$.

Les graphes d'expansion sont aussi très fortement liés aux réseaux étudiés dans cette section. Le lien entre les graphes d'expansion et nos réseaux est dû à la Proposition 3.1. En effet celle-ci exprime que le nombre d'arcs partant d'un sous-ensemble W doit être assez grand pour évacuer les signaux qui ne sont pas routés d'une entrée de W vers une sortie de W. Un (n,d,C)-graphe d'expansion est un graphe d régulier à n sommets tel que tout ensemble de sommets S de taille au plus $n/2$, a au moins $C \cdot |S|$ arêtes sortantes. Des exemples de graphes d'expansion bien connus sont les graphes de Ramanujan, c.f. [Mor94, DSV03], qui sont des (n,d,C)-graphes d'expansion ayant un coefficient d'expansion $C = 1/2(d-2\sqrt{d-1})$, ce qui est le plus grand coefficient d'expansion qu'un graphe d-régulier puisse avoir. Il existe des constructions explicites

de graphes d'expansion pour $d = q+1$ où q est la puissance d'un nombre premier et donc en particulier pour $d = 3$ et $d = 4$ qui sont les cas qui, dans [AGHP], nous ont servi pour construire des réseaux. Dans le cas général, trouver le coefficient d'expansion d'un graphe est NP-dur mais il est intéressant de noter que N. Alon démontre une condition nécessaire et suffisante pour qu'un graphe régulier biparti soit un graphe d'expansion dans [Alo86].

3.2 Réseaux (p, λ, k)

Notre problème peut être formalisé de la façon suivante : étant donnés p, λ et k où p représente le nombre de signaux reçus par le satellite, λ le nombre d'entrées inutilisables et k le nombre de sorties superflues (c.à.d. le nombre de pannes en entrées et en sorties tolérées), un **réseau** (p,λ,k) est un triplet $\{(V,E),i,o\}$ où (V,E) est un graphe, i et o sont des fonctions entières positives définies sur V appelées fonctions d'entrées et de sorties et qui vérifient les propriétés suivantes :

- les sommets du graphe représentent les commutateurs qui ont quatre sorties, donc pour tout sommet $v \in V$, $i(v) + o(v) + deg(v) \leq 4$.
- le nombre d'entrées est $i(V) = \sum_{v \in V} i(v) = p + \lambda$
- le nombre de sorties est $o(V) = \sum_{v \in V} o(v) = p + k$

De plus, le graphe doit vérifier la propriété que quelles que soient p des $p+\lambda$ entrées et p des $p+k$ sorties fonctionnelles, il existe p chemins arêtes disjoints reliant les p entrées aux p sorties. Cette dernière propriété est équivalente à la propriété suivante appellée propriété de coupe :

Proposition 3.1 *Un réseau (p,λ,k) vérifie, pour tout sous-ensemble $W \subset V$, l'***excès de** W*, défini par $\varepsilon(W) := \delta(W) + o(W) - \min(k, o(W)) - \min(i(W), p)$, est positif : $\varepsilon(W) \geq 0$.*

Cette propriété peut être comprise de la manière suivante : tout signal arrivant dans un sous-ensemble de sommets W doit, soit être routé vers une sortie fonctionnelle à l'intérieur de W (il y en a au moins $o(W) - \min(k, o(W))$), soit évacué de W par un des $\delta(W)$ liens sortants.

Ainsi pour vérifier qu'un réseau est bien un réseau (p,λ,k), il est possible, soit de résoudre un problème de flot pour chaque configuration des pannes, soit de vérifier cet invariant pour chaque sous-ensemble de sommets. Par sous-modularité de ε il est même possible de se restreindre aux sous-ensembles de sommets qui ont un complémentaire connexe.

$\mathcal{N}(p,\lambda,k)$ désigne le nombre minimum de commutateurs nécessaire pour construire un réseau (p,λ,k). Le **problème de conception de réseaux** consiste à déterminer $\mathcal{N}(p,\lambda,k)$ et à construire des réseaux (p,λ,k) minimaux.

Problème 3.2 (Conception de réseaux)

Entrées :	*Un nombre de signaux à router* p. *Un nombre d'entrées* $p+\lambda$. *Un nombre de sorties* $p+k$.
Sortie :	*Un réseau permettant de relier* p *entrées à* p *sorties quelles que soient les* λ *entrées et* k *sorties non fonctionnelles.*
Objectif :	*Minimiser la taille du réseau.*

Ce problème a initialement été étudié par E. Darrot au cours de sa thèse et dans [BDD02] dans le cas particulier où $\lambda = 0$. Dans ce contexte, un réseau $(p,0,k)$ est aussi appelé un $(p,p+k)$-**sélecteur**.

Pour fixer les idées, je présente maintenant quelque exemples. Dans toutes les figures, les entrées sont représentées par des flèches et les sorties par des carrés. Le réseau représenté dans la Figure 3.2 est un réseau $(3,1,1)$, c.à.d. un réseau capable de router trois signaux et ce même si une des entrées et une des sorties ne sont pas fonctionnelles. Les Figures 3.3 montrent que ce réseau est bien valide dans la mesure où, quelles que soient l'entrée et la sortie indisponibles, il est possible de router les trois signaux entrant vers les trois sorties disponibles avec trois chemins arêtes disjoints. La Figure 3.2 représente un (4,8)-sélecteur, autrement dit un réseau $(4,0,4)$. En effet, quelles que soient les quatre sorties en panne il est possible de router quatre signaux. La Figure 3.2 illustre deux des différentes situations possibles, le lecteur peut aisément finir de se convaincre de la validité de ce réseau en envisageant les autres situations possibles.

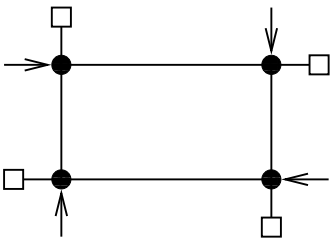


FIG. 3.2 – Un réseau $(3,1,1)$.

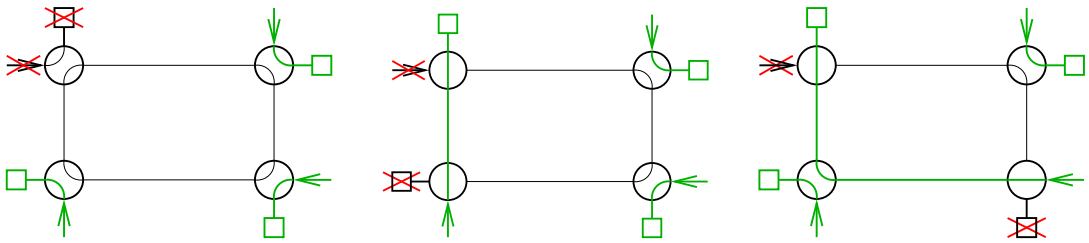
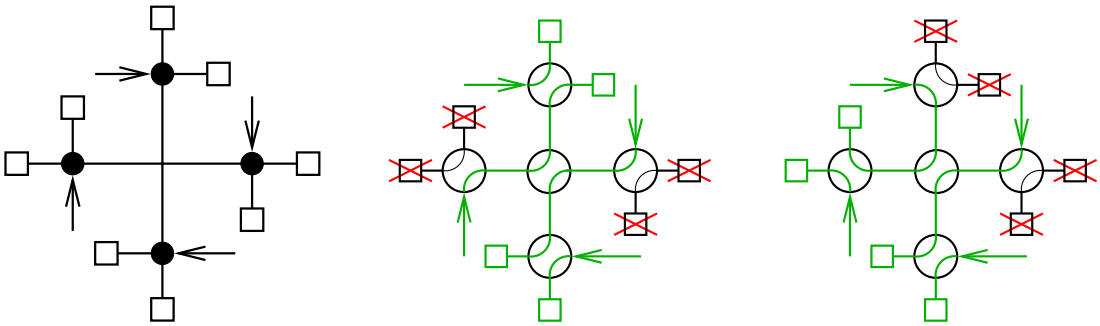


FIG. 3.3 – Validité du réseau $(3,1,1)$.



(a) Un réseau $(4,0,4)$. (b) Validité du réseau $(4,0,4)$.

FIG. 3.4 – Un réseau $(4,0,4)$ et sa validité.

Utilisation des concentrateurs et des superconcentrateurs. Maintenant que j'ai donné les définitions d'un concentrateur et d'un sélecteur, il est clair qu'un concentrateur :

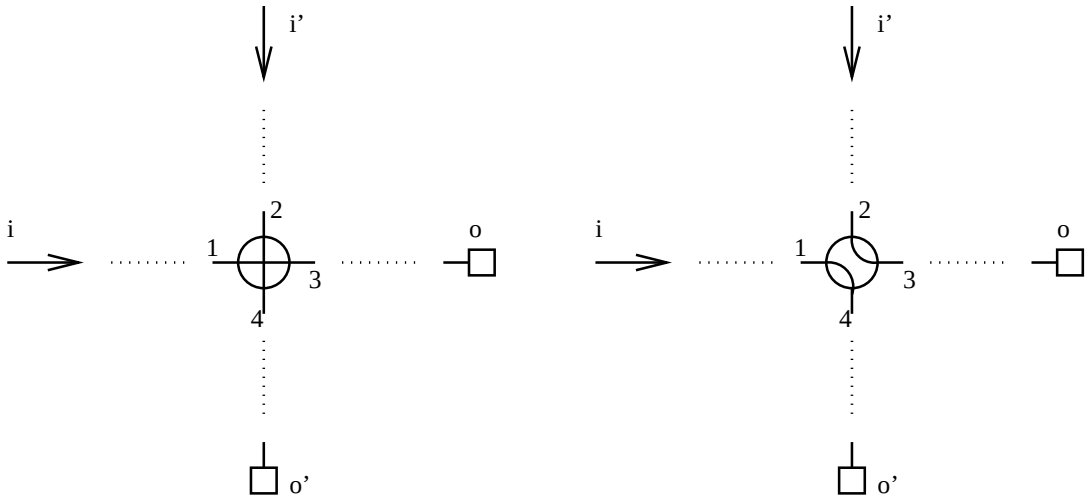
- dont le rôle des sorties et des entrées est échangé,
- dont l'orientation des arcs est ignorée,
- et qui est modifié pour tenir compte de la contrainte de degré,

donne un sélecteur. De même, il est clair qu'un superconcentrateur dont l'orientation des arcs est ignorée est un réseau (p,λ,k) pour toute valeur de $k=\lambda$, une fois modifié pour tenir compte de la contrainte de degré. Enfin, pour étendre ces constructions à des valeurs de $\lambda < k$, il suffit d'ignorer certaines des entrées. Cependant, si minimiser le nombre de commutateurs est proche du fait de minimiser le nombre d'arcs car seuls sont considérés des commutateurs de degré quatre, les sélecteurs et les réseaux (p,k,k) obtenus à l'aide de concentrateurs et de superconcentrateurs ne sont pas minimaux. Une des différences principales entre les superconcentrateurs et les réseaux (p,λ,k) est que le nombre d'entrées et de sorties qui peuvent tomber en panne dans le cadre des réseaux (p,λ,k) est borné. Le problème de concevoir des réseaux (p,λ,k) minimaux existe donc bel et bien.

Les réseaux (p,λ,k) et le problème posé par Alcatel. Cependant avant de continuer, il est important de se convaincre qu'un réseau (p,λ,k) peut effectivement être utilisé pour les satellites d'Alcatel. En effet, dans la définition d'un réseau (p,λ,k) une seule contrainte sur les chemins reliant les entrées aux sorties est imposée : qu'ils soient arêtes disjoints. Or, dans le problème initialement posé par Alcatel, il y a une autre contrainte : deux signaux ne peuvent pas se croiser n'importe comment car un commutateur ne peut pas connecter ses ports opposés simultanément (c.f. Figure 3.1). Dans le cas où certain signaux sont prioritaires cela est alors une contrainte, mais lorsque tous les signaux sont équivalents, cela n'en est plus une. En effet, considérons le cas d'un

signal routé de l'entrée i à la sortie o et d'un signal de i' à o' qui se croisent sur un commutateur de telle sorte que le premier signal arrive par le port 1 et sort par le port 3 et que le deuxième signal arrive par le port 2 et sort par le port 4 (Figure 3.5(a)). Cette situation n'est pas réalisable, mais si le premier signal est routé vers la sortie o' et le deuxième signal vers la sortie o, le premier signal va utiliser les ports 1 et 4 alors que le deuxième va utiliser les ports 2 et 3 comme le montre la Figure 3.5(b). Étant donné que les deux signaux sont équivalents la solution obtenue est toujours valide. C'est pour cette raison qu'il est possible d'ignorer les contraintes imposées par les commutateurs.

De plus, le problème étant symétrique du point de vue des entrées et des sorties dans le sens ou un réseau (p,λ,k) dont sont interverties les entrées et les sorties est un réseau (p,k,λ). Dans [AGHP], avec O. Amini, F. Giroire et S. Pérennes, nous avons considéré $k \geq \lambda$. Pour la suite de cette section, j'ai adopté la convention $n = p+k$.



(a) Deux signaux équivalents se croisant

(b) qui ne se croisent plus.

FIG. 3.5 – Le cas de deux signaux équivalents qui se croisent

3.3 Résultats antérieurs

Après avoir introduit le problème, je présente les différents résultats connus. Dans [BDD02], les auteurs donnent la valeur de $\mathcal{N}(p,\lambda,k)$ pour $\lambda = 0$ et $0 \leq k \leq 2$ (c.à.d. la taille minimale d'un $(p,p+k)$-sélecteur) : $\mathcal{N}(p,0,1) = \mathcal{N}(p,0,2) = p$. Les auteurs donnent également une construction récursive permettant d'obtenir des $(p,p+k)$-sélecteurs pour de plus grandes valeurs de k et qui donne entre autres $\mathcal{N}(4p,0,4) \leq 5p$. Ensuite J-C. Bermond, S. Pérennes et D. Tóth prouvent que cette dernière valeur est bien la bonne : en effet ils démontrent que $\mathcal{N}(p,0,3) = \mathcal{N}(p,0,4) = \lceil 5p/4 \rceil$. Ils donnent

également les résultats asymptotiques suivants : $\mathcal{N}(p,0,6) = \lceil 5p/4 \rceil + \sqrt{p/8} + O(1)$, $\mathcal{N}(p,0,8) = \lceil 4p/3 \rceil + 2/3\sqrt{p/3} + O(p^{1/4})$ et $\mathcal{N}(p,0,10) = 11p/8 + \Theta(\sqrt{p})$. Pour les plus grandes valeurs de k ils montrent que $\mathcal{N}(p,0,k) = 3p/2 + k/2$ et que cette borne est asymptotiquement serrée. D'autres résultats sur les sélecteurs peuvent être trouvés dans [Hav06]. F. Havet y étudie le cas où le nombre de pannes tolérées est élevé par rapport au nombre d'entrées. Il montre que pour un réseau destiné à router p signaux et tolérant k pannes en sorties (0 en entrées), si $p \geq k$, $\mathcal{N}(p,0,k) \leq n/2 + 17p + O(\log(n))$ et si $p \leq k$, $\mathcal{N}(p,0,k) \leq 17n - 16p + O(\log(n))$. Je rappelle la convention $n = p + k$. F. Havet montre également deux bornes inférieures suivant si le nombre d'entrées est pair ou impair :

- si le nombre d'entrées p est pair : $\mathcal{N}(p,0,k) \geq (2^{p/2} - 1)/2^{p/2} n + \Theta(1)$,
- si p est impair : $\mathcal{N}(p,0,k) \geq (2^{(p+1)/2} - 3)/2^{(p+1)/2} n + \Theta(1)$.

Il conjecture que cette dernière borne inférieure est en fait la bonne valeur et le prouve dans le cas où le nombre d'entrées est petit (de 1 à 6). Si le problème général dans lequel les $\lambda > 0$ est cité dans [BDD02], les premiers résultats sont obtenus par J-C. Bermond, F. Giroire et S. Pérennes dans [BGP07]. Ils donnent la valeur de $\mathcal{N}(p,\lambda,k)$ pour de petites valeurs de k et λ :

- $\mathcal{N}(p,1,2) = \mathcal{N}(p,2,2) = p + 2$.
- pour $k = 3$ ou $k = 4$, $0 < \lambda \leq k$, $\mathcal{N}(p,\lambda,k) = \lceil 5n/4 \rceil$.
- pour $k = 5$ ou $k = 6$, $0 < \lambda \leq k$, $\mathcal{N}(p,\lambda,k) \leq \lceil 3n/2 \rceil$.

3.4 Réseaux tolérant un grand nombre de pannes : contributions

Dans l'article [AGHP] présent dans l'Annexe A.1, avec O. Amini, F. Giroire et S. Pérennes, nous avons étudié le problème de conception de réseaux dans le cas où le nombre de pannes peut être élevé. De plus, nous avons introduit une variante, dite **problème de conception de réseaux simplifiés** qui est étudiée dans les mêmes conditions. Un réseau (p,λ,k) simplifié est tel que $p + \lambda$ commutateurs sont reliés à exactement une entrée et une sortie (de tels commutateurs sont appelés des **doublons**, c.f. Figure 3.6) et $k - \lambda$ commutateurs ont une sortie.

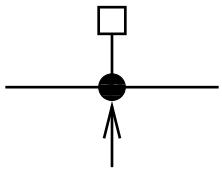


FIG. 3.6 – Un commutateur ayant une entrée et une sortie : un doublon.

Comme le signale déjà les auteurs de [BDD02] dans la conclusion, les réseaux dits simplifiés sont intéressants pour les applications pour plusieurs raisons. En effet, ils ont un processus de routage simple : la plupart des signaux peuvent être envoyés sur la sortie

présente sur le même commutateur. De plus, dans ce routage, les chemins empruntés par les signaux sont très courts et donc minimisent l'atténuation des signaux. Ces chemins limitent également les interférences entre signaux.

Concernant les preuves des résultats obtenus, le lecteur intéressé peut lire l'article [AGHP] présent en Annexe A.1, je me contente de présenter ici ces résultats, c.à.d. plusieurs bornes inférieures et supérieures sur la taille minimale des réseaux (p,λ,k) et des réseaux (p,λ,k) simplifiés. Dans de nombreux cas, ces bornes donnent asymptotiquement la taille minimale des réseaux (p,λ,k) et des réseaux (p,λ,k) simplifiés.

Afin d'obtenir les bornes inférieures, dans [AGHP] nous avons utilisé une nouvelle technique : la quasi-partition qui dépend du Lemme 3.4. Pour pouvoir exposer ce lemme, j'ai besoin de la définition suivante :

Définition 3.3 (q-quasi-partition) *Soit $G=(V,E)$ un graphe et q un entier positif. une q-*quasi-partition de G *est une famille $Q=\{A_1,A_2,\ldots,A_m\}$ de sous-ensembles de V, telle que :*

(i) pour tout $1\leq i\leq m$, le sous graphe $G[A_i]$ induit par A_i est connexe ;

(ii) pour tout $1\leq i\leq m$, $\frac{q}{3}\leq |A_i|\leq q$;

(iii) $V=\bigcup_{i=1}^{m}A_i$ et $\sum_{i=1}^{m}|A_i|\leq |V|+|\{A_i;|A_i|>\frac{2q}{3}\}|+1$.

Lemme 3.4 ([DHMP06]) *Soit q un entier positif et G un graphe connexe de taille au moins $\frac{q}{3}$. G admet une q-quasi-partition.*

Bornes inférieures. Le Lemme 3.4 permet d'obtenir des informations sur l'ensemble du réseaux à partir d'observations locales. Il sert pour prouver le théorème suivant :

Théorème 3.5 *Dans un réseau (p,λ,k) de taille N, pour $k\leq\frac{n}{2}$:*

$N\geq\left(\frac{3}{2}n-(k-\lambda)-4-\frac{12}{\sqrt{k}}\right)\left(1-\frac{9}{2\sqrt{k}}+O\left(\frac{1}{k}\right)\right)+\frac{d}{2}\left(1+\frac{9}{2\sqrt{k}}+O\left(\frac{1}{k}\right)\right)$.

avec $n=p+k$ et d le nombre de doublons, c.à.d. le nombre de commutateurs ayant exactement une entrée et une sortie.

Ce théorème est ensuite utilisé pour obtenir des bornes inférieures pour les différents problèmes. Dans le cas général, il donne le Corollaire 3.6 :

Corollaire 3.6 *(Cas Général). Pour $\lambda\to\infty$ et $k\to\infty$, un réseau (p,λ,k) contient au moins $n+\frac{2}{3}n+O(\frac{n}{\sqrt{\lambda}})$ commutateurs. Autrement dit, $\mathcal{N}(p,\lambda,k)\geq n+\frac{2}{3}n+O(\frac{n}{\sqrt{\lambda}})$.*

Dans le cas où le réseau ne tolère pas les pannes d'entrées, c.à.d. quand $\lambda = 0$, ou encore le cas des sélecteurs, cela donne le Corollaire 3.7.

Corollaire 3.7 *(Sélecteurs, $\lambda = 0$). Dans un réseau $(p,0,k)$ de taille N, pour $k \to \infty$ et $k \leq \frac{n}{2}$, on a :*

$$N \geq n + \frac{n}{2} + O(\frac{n}{\sqrt{k}})$$

Autrement dit :

$$\mathcal{N}(p,0,k) \geq n + \frac{n}{2} + O(\frac{n}{\sqrt{k}}).$$

Enfin dans le cas dit simplifié, cela donne le Corollaire 3.8.

Corollaire 3.8 *(Cas simplifié). Dans un réseau (p,λ,k) simplifié, pour $k \to \infty$ et $k \leq \frac{n}{2}$, il y a au moins $2n + O(\frac{n}{\sqrt{k}})$ commutateurs, c.à.d. $\mathcal{N}(p,\lambda,k) \geq 2n + O(\frac{n}{\sqrt{k}})$ dans le cas simplifié.*

Bornes supérieures. Pour montrer des bornes supérieures, dans [AGHP], nous construisons des réseaux (p,λ,k) à l'aide de graphes d'expansion. Seulement les réseaux construits de cette façon ne sont valables que pour des valeurs de k et λ bornées par $\log n$. L'ensemble des bornes supérieures est listé ci-dessous :

Théorème 3.9 *(Cas général). Pour $n = p + k$, $k \leq \frac{1}{15}\log n$, et n assez grand : $N(p,\lambda,k) \leq n + \frac{3}{4}n$*

Nous faisons d'ailleurs la conjecture que cette borne supérieure est la valeur de $\mathcal{N}(p,\lambda,k)$.

Conjecture 3.10 *Pour $n = p + k$, $k \leq \frac{1}{15}\log n$, et n assez grand : $N(p,\lambda,k) = n + \frac{3}{4}n$*

Théorème 3.11 *(Sélecteurs, $\lambda = 0$). Pour $n = p + k$, $k \leq \frac{1}{48}\log_2 n$, et n assez grand :*

$$N(p,0,k) \leq n + \frac{n}{2}.$$

Théorème 3.12 *(Cas simplifié). Pour $n = p + k$, $k \leq \frac{1}{6}\log n$, et n assez grand : $N(p,\lambda,k) \leq 2n$.*

Les résultats antérieurs ne donnaient la valeur de $\mathcal{N}(p,\lambda,k)$ que dans le cas des réseaux tolérant un nombre de pannes maximum borné par une constante, ainsi ces derniers théorèmes apportent des informations nouvelles en donnant la valeur asymptotique de $\mathcal{N}(p,\lambda,k)$ pour des valeurs plus grandes de k. Cependant k reste borné par $\log n$. Afin d'obtenir des résultats pour des valeurs de k plus grande encore, nous avons défini un nouveau paramètre appelé l'**α-robustesse**. Le but de ce paramètre est d'évaluer l'expansion locale d'un graphe. Étant donné un réel α, l'$\alpha-$robustesse d'un graphe,

r_α, est le plus grand entier tel que pour tout sous-ensemble $X \subset V$ de taille $|X| \leq |V|/2$, on a :

$$\delta(X) \geq \min(\alpha|X|, r_\alpha).$$

Cela signifie que les petits ensembles ont un facteur d'expansion α alors que les plus grands ensembles (de taille plus grande que r_α/α) ont au moins r_α arêtes sortantes. Cette notion diffère des approches usuelles dans la mesure où, au lieu de chercher le coefficient d'expansion d'un graphe, un coefficient d'expansion est fixé et le but est de chercher quels sont les ensembles qui le vérifient. Des notions proches sont présentées dans [CRVW02, AHK99].

Liens entre la robustesse et d'autres paramètres. Cette définition est motivée par le fait qu'un réseau (p,λ,k) n'a pas besoin de vérifier de bonnes propriétés d'expansion pour les grands ensembles de sommets. Cette notion est également intéressante en elle-même et généralise plusieurs invariants comme la bisection-width et la constante de Cheeger ([Chu97]). Je rappelle que la bisection-width est le plus petit nombre d'arêtes entre les deux ensembles d'une partition des sommets en deux ensembles de même taille. Elle a entre autres été étudiée par N. Alon dans [Alo93] et par B. Moniens et R. Preis dans [MP06]. L'α-robustesse est également reliée à la notion de graphe d'expansion, par exemple les graphes de Ramanujan qui ont un facteur d'expansion $1/2(d-2\sqrt{d-1})$ ont une $1/2(d-2\sqrt{d-1})$-robustesse de $n/2$.

Application de l'α-robustesse. Pour construire des réseaux (p,λ,k), il faut des graphes ayant une bonne 1-robustesse. Comme la robustesse est une notion proche de l'expansion et qu'il se trouve que les graphes aléatoires sont de bons graphes d'expansion, dans [AGHP], nous avons étudié l'$\alpha-$robustesse des graphes aléatoires 4-réguliers. Nous y généralisons des résultats de B. Bollobàs sur leurs coefficients d'expansion ([Bol88]). Nous n'avons pas trouvé d'articles antérieurs parlant de ce concept d'expansion locale, et les résultats que nous obtenons nous permettent de construire des réseaux (p,λ,k) ayant $3n$ commutateurs et tolérant jusqu'à $n/7$ pannes (d'entrées comme de sorties). Je présente ces preuves dans la Section 3.4.1. Elles peuvent également être trouvées dans la thèse de F. Giroire [Gir06] ou dans l'Annexe A.1.

Théorème 3.13 *Les graphes aléatoires 4-réguliers ont une 1-robustesse d'au moins* $n/14$ *avec probabilité* $1-O(\frac{\log n}{n})$.

Corollaire 3.14 *Pour* $k \leq \frac{n}{7}$, *il existe un réseau* (p,λ,k) *de taille* $3n$ *où* $n = p+k$.

Preuve. Soit G un graphe Hamiltonien 4-régulier sur $2n$ sommets ayant une 1-robustesse d'au moins $n/7$. Si un doublon (en bleu sur la Figure 3.7) est rajouté à chaque arête d'un couplage parfait (en pointillés rouges) obtenu à partir d'un cycle Hamiltonien, cela donne un réseau $(p = n-k,k,k)$ pour tout $k \leq n/7$ qui a $3n$ sommets.

Il est facile de vérifier que c'est bien un réseau (p,λ,k) à l'aide de la Propriété 3.1. Cette construction peut se généraliser pour tout $k \leq n/2$ si les arêtes auxquelles les doublons sont rajoutés, sont choisies à bonne distance. □

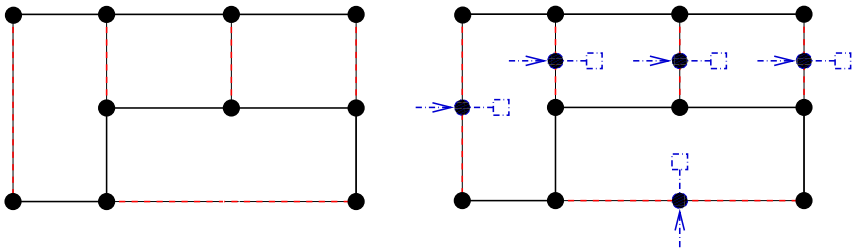


FIG. 3.7 – Un réseau $(p = n-k, k, k)$ obtenu à partir d'un graphe hamiltonien.

3.4.1 Robustesse des graphes 4-réguliers

Pour prouver le Théorème 3.13, il est nécessaire de prouver que dans les graphes aléatoires 4-réguliers, tous les ensembles contenant $s \leq n/14$ sommets ont une frontière de taille au moins s et que tous les ensembles de taille $s \geq n/14$ ont une frontière de taille au moins $n/14$. Ce sont les Lemmes 3.17 et 3.18.

Ces résultats ont été prouvés en utilisant la méthode du premier moment pour des éléments de l'espace de probabilité définit ci-dessous :

Définition 3.15 *$\mathcal{G}(2,n)$ désigne l'espace de probabilité de tous les graphes à n sommets qui sont l'union de deux cycles Hamiltoniens, tous les graphes ayant la même probabilité. $G_{2,n}$ désigne un élément de $\mathcal{G}(2,n)$.*

Remarque 3.16 Il y a $n!^2$ éléments dans $\mathcal{G}(2,n)$. Un élément de $\mathcal{G}(2,n)$ peut avoir des arêtes multiples.

Lemme 3.17 *Pour n grand, dans un graphe aléatoire 4-réguliers, tout sous-ensemble de taille $s \leq n/14$ a une frontière de taille au moins s avec probabilité $u > 1/2$.*

Lemme 3.18 *Pour n grand, dans un graphe aléatoire 4-réguliers, tout sous-ensemble de taille $s \in [\frac{n}{14}, \frac{13n}{14}]$ a une frontière de taille au moins $\frac{n}{14}$ avec une probabilité $u > 1/2$.*

Les preuves de ces résultats se trouvent dans l'Annexe A.1.

3.5 Conclusion et perspectives

Si le travail que nous avons réalisé dans [AGHP] a permis de répondre à certaines questions concernant la taille des réseaux (p, λ, k), de nombreuses questions intéressantes restent à étudier, entre autres concernant le paramètre que nous avons introduit dans ce contexte : la robustesse. En effet le Théorème 3.13 montre l'existence de graphes 4-régulier ayant une bonne α-robustesse, mais il est basé sur les graphes aléatoires. Ainsi le Corollaire 3.14 ne donne que l'existence d'un réseau (p, λ, k) tolérant $n/7$ pannes de taille $3n$. Afin d'avoir un exemple d'un tel réseau, il serait intéressant de trouver des constructions de graphes ayant une grande α-robustesse. Concernant les graphes

d'expansion, il est possible d'en construire à l'aide du produit Zigzag qui a été introduit par [RVW02] ou par le "lift" de [BL06], il est donc naturel de se demander si des techniques similaires permettent d'obtenir des graphes ayant une bonne robustesse.

Précédemment nous avons dit que l'α-robustesse généralise la bisection-width. Dans [MP06], les auteurs montrent que la bisection-width maximale d'un graphe à n sommets est au plus $2n/5$. Cela implique que l'α-robustesse est au plus $\frac{n}{2}$ quand $\alpha > \frac{4}{5}$. Est-il possible de généraliser ce résultat et, par exemple, de donner une condition sur α assurant une α-robustesse d'au plus $n/3$?

Chapitre 4

Algorithmes de routage

Dans ce chapitre, je m'intéresse à la couche réseau du système OSI, c.à.d. à la couche numéro trois. Je rappelle que le but de cette couche est de gérer les connexions entre les différents systèmes composant le réseau. Dans la première section, je présente des algorithmes de routage optimaux pour le routage de paquets dans les grilles hexagonales et triangulaires, pour le modèle Δ-port, half- ou full-duplex, dans le cas de la permutation, du routage r-central et du (ℓ,ℓ)-routage. Je présente également un algorithme d'approximation dans le cas du (ℓ,k)-routage. Dans la deuxième section, je présente des algorithmes d'approximation pour la coloration k-impropre des graphes hexagonaux. Ce problème modélise un problème d'affectation de fréquences pour des transmissions entre un satellite et des antennes. Dans la dernière section, je montre qu'un problème d'ordonnancement de connexions dans un réseau radio sans fils maillé peut être modélisé par un problème de coloration appelé coloration proportionnelle qui est NP-dur dans le cas général.

Étant donné un réseau et des requêtes entre les systèmes du réseau (chaque requête correspond à des données à transmettre entre des systèmes), le problème consiste à organiser le transfert de ces requêtes à l'aide des infrastructures disponibles. Ainsi, le problème dépend fortement du réseau considéré. Dans ce chapitre, le problème est étudié dans trois types de réseaux différents :

- Les premiers réseaux considérés sont des réseaux en forme de grilles qui représentent typiquement des processeurs reliés entre eux.
- Les seconds réseaux considérés sont des réseaux WiFi constitués d'antennes recevant des signaux d'un satellite.
- Les derniers réseaux étudiés sont également des réseaux WiFi, qui sont composés de routeurs communiquant entre eux.

Dans les trois cas, je considère uniquement des requêtes émises par un système et à destination de ceux-ci. Je ne porterai pas, dans ce chapitre, mon attention sur les

requêtes multicasts, qui seront traitées plus tard.

Dans chacune des sections sont utilisées des techniques différentes. Cela illustre la variété des approches disponibles pour résoudre des problèmes similaires. Dans la première section, je présente des résultats obtenus à partir d'algorithmes de type *big foot*, tandis que les problèmes des deux sections suivantes sont modélisés par deux problèmes différents de coloration de graphes.

4.1 Algorithmes de routage dans les grilles

Le travail que je présente dans cette section a été réalisé en collaboration avec O. Amini, I. Sau Valls et J. Žerovnik. L'article que nous avons soumis à JOIN peut être trouvé dans l'Annexe B.1. Le problème auquel nous nous sommes intéressés est un problème d'ordonnancement. Étant donné un ensemble de paquets, caractérisés par leur origine et leur destination, l'objectif est de trouver un chemin et un ordonnacement des déplacements de chacun des paquets afin de tous les acheminer à leur destination tout en minimisant le temps pris par l'acheminement de tous les paquets. Ce problème peut être formulé de la manière suivante :

Problème 4.1 (Problème de routage de paquets dans les réseaux)

Entrées :	*Un réseau.*
	Un ensemble de paquets, chacun caractérisé par une source et une destination.
Sorties :	*Un chemin pour chacun des paquets.*
	Un ordonnancement des déplacements de chacun des paquets.
Objectif :	*Minimiser le temps nécessaire pour acheminer tous les paquets.*

4.1.1 Contexte

En 1988, T. Leighton, B. Maggs et S. Rao prouvent dans [LMR88, LMR94], que quel que soit le réseau, pour tout ensemble de paquets dont on connait a priori les chemins sur lesquels ils vont être routés, il existe un **ordonnancement** permettant d'acheminer tous les paquets le long de leur chemin en un temps optimal de $O(C+D)$ étapes, où C représente la **charge du réseau** (le nombre maximum de chemins utilisant une même arête) et D la **dilatation** (la longueur du chemin le plus long).

Théorème 4.2 ([LMR88]) *Pour tout ensemble de paires paquet-chemin, il existe un protocole de routage hors-ligne nécessitant $O(C+D)$ étapes pour router tous les paquets.*

De plus, dans [ST97], les auteurs montrent qu'étant donné un ensemble de paquets, il est possible de trouver en temps polynomial un chemin pour chacun des paquets tel que la valeur de $C+D$ dans ce routage soit au plus à un facteur quatre de l'optimum. Ainsi on peut reformuler le Théorème 4.2 de manière plus générale.

Théorème 4.3 ([ST97]) *Pour tout ensemble de paquets, il existe un protocole de routage hors-ligne nécessitant $O(C+D)$ étapes pour router tous les paquets, où $C+D$ est le minimum de "la charge du réseau plus la dilatation" sur tous les routages.*

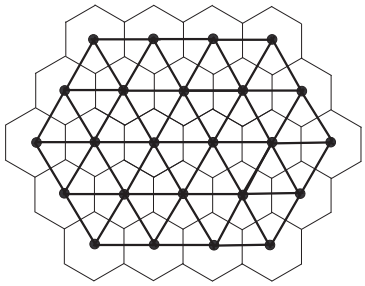


FIG. 4.1 – Grilles triangulaire (△) et hexagonale. (⬡)

Qui plus est, dans l'ordonnancement obtenu, à chaque étape, le nombre de paquets devant être stockés par un nœud est borné par une constante.

Bien que ces résultats soient asymptotiquement les meilleurs possibles, si l'on considère certains types de paquets et certains types de réseaux bien précis, il est possible de concevoir des algorithmes qui sont plus efficaces en pratique. En effet, un gros facteur multiplicatif peut se cacher derrière le O dans le théorème précédent.

4.1.2 Les différents scénarios de requêtes

Les différents types de scénarios pour les paquets sont définis à partir du nombre de paquets émis et reçus par chaque système du réseau. Les problèmes les plus couramment étudiés sont les suivants :

1. **Permutation** : chaque nœud est source et destination d'exactement un paquet.
2. (ℓ,k)**-routage** : chaque système est source d'au plus ℓ paquets et en est destination d'au plus k. Un autre cas particulier important est le cas $(1,k)$**-routage**.
3. $(1,tous)$**-routage** : chaque nœud est source d'au plus un paquet mais il n'y a pas de contrainte sur le nombre maximum de paquets dont un nœud est destination.
4. **routage** r**-central** : tous les nœuds à distance au plus r du nœud central lui adressent un paquet.

Dans chacun de ces scénarios, étant donnée une configuration initiale, l'objectif est de trouver des chemins et un ordonnancement permettant d'acheminer les paquets le plus rapidement possible.

4.1.3 Les différents types de réseaux

Topologie. Le réseau le plus étudié est la grille carrée. Une des raisons est qu'elle est utilisée dans les réseaux de processeurs. Les réseaux radios sont, quant à eux, souvent modélisés par des grilles hexagonales ou triangulaires dans lesquelles les nœuds représentent les stations de base. Avec O. Amini, I. Sau Vals et J. Žerovnik, ce sont à ces trois types de grilles (carrées, triangulaires et hexagonales) que nous nous sommes intéressés dans [AHSZ08].

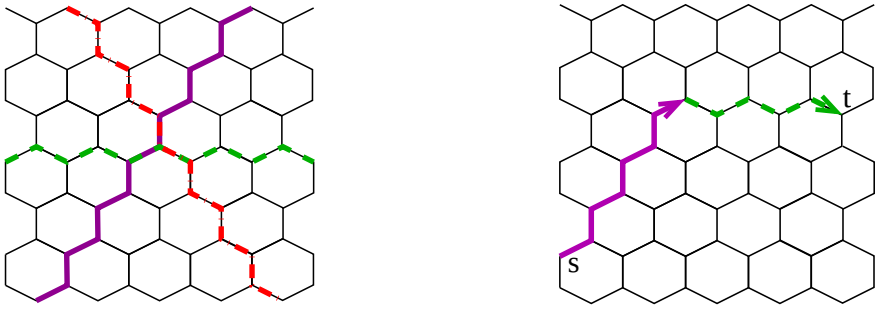


FIG. 4.2 – Trois directions dans la grille hexagonale et le principe de l'algorithme big foot.

La Figure 4.1 représente les grilles triangulaires et hexagonales. La grille hexagonale est un sous-graphe de la grille triangulaire. La grille triangulaire peut être obtenue à partir d'une grille carrée en rajoutant une diagonale, c'est aussi le dual de la grille hexagonale. Nous considérons des sous-grilles convexes (c.à.d. contenant tous les chemins les plus courts entre deux sommets) des trois grilles carrées, triangulaires et hexagonales.

Caractéristiques des composants. Nous nous plaçons dans le modèle où chaque système peut stocker et retransmettre les paquets reçus (**store-and-forward** model). Chaque système doit donc être capable de gérer une file d'attente de paquets (**queue**). Chaque système peut également recevoir et envoyer des paquets sur tous les liens qui le connectent au réseau en même temps (**Δ-port model**). Enfin, nous considèrons à la fois le cas où les liens sont bidirectionnels, et ne peuvent donc être utilisés que dans un sens à chaque étape (**half-duplex networks**), et le cas où chaque lien représente en fait deux liens, un dans chaque sens, ce qui permet à chaque étape de transmettre des données simultanément dans les deux directions (**full-duplex networks**).

4.1.4 Résultats connus dans la grille carrée

Je rappelle très brièvement les résultats connus concernant les grilles carrées.

Dans [HLJ97], les auteurs introduisent un algorithme appelé **big foot**. Étant données plusieurs directions (horizontale X et verticale Y dans la grille carrée), l'idée de l'algorithme est (dans une grille à deux dimensions) de router chacun des paquets aussi loin que utile le long d'une première direction puis de finir le routage en utilisant une seconde direction. Dans une grille carrée, ce routage est également connu sous le nom de routage XY. Les algorithmes que nous proposons utilisent ce principe, comme illustré sur la Figure 4.2, dans le cas de la grille hexagonale.

Permutation. Dans [SCK97], les auteurs proposent un algorithme trouvant un ordonnancement optimal (utilisant $2n-2$ étapes) et utilisant des files d'attente de taille au plus 81. Ils proposent également un algorithme trouvant un ordonnancement utilisant

$2n + O(1)$ étapes et utilisant des files d'attente de taille au plus 12.

$(1,k)$-routage. Dans [SK94], les auteurs proposent un algorithme donnant un ordonnancement quasi optimal qui utilise $\sqrt{k}\frac{n}{2} + O(n)$ étapes. Ils donnent un autre algorithme calculant un ordonnancement nécessitant un peu plus d'étapes mais qui utilise des files d'attente de taille maximale 3.

(ℓ,ℓ)-routage. Dans [SK94], les auteurs proposent également un algorithme calculant des ordonnancements utilisant $O(\ell n)$ étapes, sachant que dans [SK94], les auteurs proposent une borne inférieure en $\Omega(\sqrt{\ell k n})$ pour le (ℓ,k)-routage.

(ℓ,k)-routage. Dans [PP01], les auteurs donnent des algorithmes déterministes et aléatoires permettant de calculer un ordonnancement utilisant $O(\sqrt{\ell k n})$ étapes, ce qui est optimal.

4.1.5 Algorithmes de routage dans les grilles hexagonales et triangulaires : contributions

Les algorithmes que nous proposons dans [AHSZ08] (et qui peuvent être trouvés dans l'Annexe B.1) sont tous distribués, et peuvent ainsi être utilisés indépendamment par chaque nœud. Ils utilisent tous les chemins les plus courts, ce qui permet de minimiser la dilatation qui est alors la longueur du plus long plus court chemin, et est notée ℓ_{max}. Cependant, un routage utilisant les chemins les plus courts ne permet pas toujours d'avoir un ordonnancement utilisant le nombre minimum d'étapes.

Permutation. Nous proposons le premier algorithme permettant de calculer des ordonnancements utilisant le moins possible d'étapes dans les grilles hexagonales full-duplex ($2\ell_{max} - 2$ étapes) et half-duplex ($4\ell_{max} - 4$ étapes) ainsi que dans les grilles triangulaires half-duplex ($2\ell_{max}$ étapes).

Routage r-central. Nous proposons le premier algorithme permettant de calculer des ordonnancements utilisant le moins possible d'étapes dans les grilles carrées, hexagonales et triangulaires (full et half-duplex) en respectivement $\binom{r+1}{2}$, $\binom{3r+1}{2}$ et $\binom{6r+1}{2}$ étapes pour le cas full-duplex (le double dans half-duplex).

(ℓ,ℓ)-routage. Nous proposons le premier algorithme permettant de calculer des ordonnancements utilisant le moins possible d'étapes dans les grilles carrées, hexagonales ($2\ell\ell_{max}$ en full-duplex) et triangulaires ($\ell\ell_{max}$ en full-duplex) (full et half-duplex).

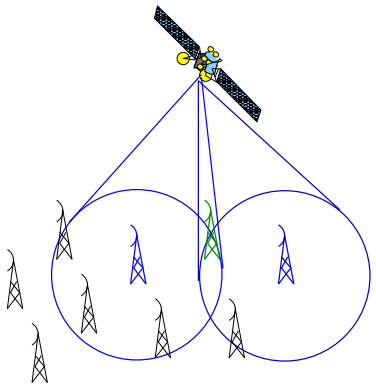
(ℓ,k)-routage. Nous proposons un algorithme permettant de calculer des ordonnancements utilisant un nombre d'étapes à un facteur constant garanti (mais dont la formule n'est pas très élégante) du nombre minimum d'étapes dans les grilles carrées, hexagonales et triangulaires (full et half-duplex).

4.2 Allocation de fréquences et coloration impropre des graphes hexagonaux pondérés

Dans cette section, je reprends un article qui a été présenté à Algotel 07. J'ai par la suite continué à travailler sur cette problématique et la version étendue de cet article est présente en annexe. Ce travail a été fait en collaboration avec Frédéric Havet, Jean-Claude Bermond et Cláudia Linhares-Sales. L'article en cours de préparation peut être trouvé dans l'Annexe B.2.

4.2.1 Origine du problème

Contexte. Ce travail est motivé par un problème posé par Alcatel Space Technologies. Il a déjà été étudié par J-F. Lalande au cours de sa thèse et dans [AAG+05, AAG+06], mais ici une approche différente est utilisée. Ce problème concerne un satellite qui envoie des informations vers des récepteurs terrestres qui captent chacun plusieurs fréquences. Techniquement, il est impossible de concentrer un signal envoyé par le satellite sur le récepteur auquel il est destiné. Ainsi, un signal est émis sur une zone autour du récepteur, créant du bruit pour les autres récepteurs situés dans cette zone. Chaque récepteur est capable de distinguer un signal qui lui est envoyé des bruits extérieurs qu'il reçoit si la somme de ces bruits n'est pas trop grande, c.à.d si elle ne dépasse pas une certaine limite T. Le problème est d'allouer des fréquences aux récepteurs de manière à ce que chaque récepteur puisse capter ses signaux et les distinguer des bruits, tout en utilisant le moins de fréquences possible. Les signaux sont émis en continu, ainsi il est nécessaire de maintenir toutes les connexions simultanément. Cela diffère du problème précédent dans lequel on ne s'intéressait qu'à la transmission d'un paquet. Une présentation plus technique du modèle peut être trouvée dans le chapitre 5 de la Thèse de J-F. Lalande [Lal04].



Généralement, la "relation de bruit" est symétrique (si un récepteur u reçoit du bruit du récepteur v alors v reçoit du bruit de u). Ainsi, les interférences peuvent être modélisées par un **graphe de bruit** dont les sommets sont les récepteurs et pour lequel deux sommets sont reliés si et seulement si les récepteurs correspondant interfèrent. De

plus, le graphe est muni d'une **fonction de poids** $w : V \to \mathbb{N}$, où le poids $w(v)$ du sommet v est égal au nombre de signaux que le récepteur correspondant doit recevoir. Cela donne un **graphe pondéré**, c.à.d. une paire (G,w) où G est un graphe et w une fonction de poids sur les sommets de G.

Énoncé du problème. Dans une version simplifiée, l'intensité I du bruit créé est indépendante de la fréquence et du récepteur. Ainsi, pour pouvoir distinguer son signal des bruits, un récepteur doit être dans la zone de bruit d'au plus $k = \lfloor T/I \rfloor$ récepteurs écoutant sur la même fréquence. Le problème revient donc à trouver une coloration du graphe (pondéré) de bruit qui soit k-impropre. Une **coloration** du graphe pondéré (G,w) est une fonction $C : V \to \mathcal{P}(S)$ telle que $|C(v)| \geq w(v)$, l'ensemble S étant l'ensemble de **couleurs** et $\mathcal{P}(S)$ l'ensemble des parties de S. Comme seule la cardinalité de S est intéressante, $S = \{1,2,\ldots,l\}$ pour un certain entier l. Une coloration de (G,w) avec un ensemble de couleurs de taille l est appelée **l-coloration**. Une coloration C de (G,w) est **k-impropre** si pour toute couleur i, l'ensemble des sommets ayant la couleur i induit un graphe de degré au plus k. En d'autres termes, tout sommet recevant une couleur i est adjacent à au plus k sommets recevant cette même couleur i. Le **nombre chromatique k-impropre** de (G,w), noté $\chi_k(G,w)$, est le plus petit entier l tel que (G,w) ait une l-coloration k-impropre. Puisque le but est de minimiser le nombre de fréquences, il faut trouver une coloration k-impropre de (G,w) ayant un nombre de couleurs aussi proche que possible de l'optimum $\chi_k(G,w)$.

Problème 4.4 (Coloration impropre d'un graphe pondéré)

Entrées :	*Un graphe pondéré* (G,w)*.*
	Un indice d'impropreté k*.*
Sortie :	*Une coloration* k*-impropre de* (G,w)*.*
Objectif :	*Garantir un nombre de couleurs utilisées aussi proche que possible de l'optimum.*

Modèle utilisé. Dans le problème posé par Alcatel, la zone de réception est découpée en cellules hexagonales, une cellule étant adjacente aux six cellules voisines. Ainsi, les récepteurs sont répartis comme les sommets du réseau triangulaire R, qui est représenté sur la Figure 4.3, et peut être décrit comme suit : les sommets sont les combinaisons linéaires $a\mathbf{e_1} + b\mathbf{e_2}$ des vecteurs $\mathbf{e_1} = (1,0)$ et $\mathbf{e_2} = (\frac{1}{2}, \frac{\sqrt{3}}{2})$: ainsi il est possible d'identifier les sommets avec les paires (a,b) d'entiers.

Deux sommets sont adjacents si la distance euclidienne entre eux est égale à 1. Ainsi chaque sommet $x = (a,b)$ a six voisins : $(a-1,b)$, $(a+1,b)$, $(a-1,b+1)$, $(a,b+1)$, $(a,b-1)$ et $(a+1,b-1)$. Les graphes de bruit étudiés dans ce chapitre, sont des sous-graphes induits du réseau triangulaire (Figure 4.3), appelés **graphes hexagonaux**. La Figure 4.4 donne une 5-coloration 1-impropre de R avec $w(v) = 2$ pour tout $v \in V(R)$.

Résultats connus. Pour $k \geq 6$, le nombre chromatique k-impropre d'un graphe hexagonal est son **poids maximum** $w_{\max} = \max\{w(v) \mid v \in V(G)\}$ car son degré maximum est au plus 6. Ainsi seules sont étudiées des impropretés inférieures à 6. Généralisant

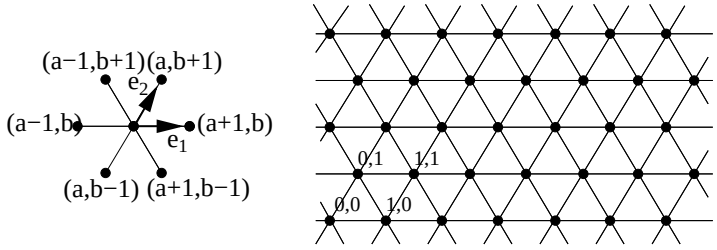


FIG. 4.3 – Les récepteurs sont répartis sur les sommets d'un réseau triangulaire.

un résultat de McDiarmid et Reed [MR00] pour la coloration propre, F. Havet, Kang et J-S. Sereni [HKS05, Ser06] ont montré que pour tout $0 \leq k \leq 5$, il est NP-complet de décider si le nombre chromatique k-impropre d'un graphe hexagonal pondéré est 3 ou 4. Ainsi il n'existe pas d'algorithme polynomial pour trouver le nombre chromatique k-impropre d'un graphe hexagonal pondéré (à moins que P=NP). Le but est de trouver des algorithmes **α_k-approchés**, c.à.d. qui trouvent une coloration k-impropre de tout graphe hexagonal pondéré (G,w) avec au plus $\alpha_k \times \chi_k(G,w) + \beta_k$ couleurs, où α_k et β_k sont deux constantes.

C. McDiarmid and B. Reed [MR00] ont donné un algorithme 4/3-approché pour la coloration propre des graphes hexagonaux pondérés. Un algorithme distribué garantissant ce même rapport de 4/3 est donné par Narayanan et Schende [NS01].

Dans [AAG+06], les auteurs proposent une formulation mathématique, sous forme de génération de colonnes, pour résoudre le problème d'affectation de longueur d'onde.

Plan. Dans cette section, je présente un algorithme α_k-approché pour la coloration k-impropre des graphes hexagonaux pondérés pour tout $1 \leq k \leq 5$, avec $\alpha_1 = \frac{25}{13}$, $\alpha_2 = \frac{12}{7}$, $\alpha_3 = \frac{18}{13}$, $\alpha_4 = \frac{80}{63}$ et $\alpha_5 = \frac{49}{43}$. La méthode proposée pouvant être appliquée à tous les graphes pondérés, je la présente dans un contexte général.

4.2.2 Algorithmes d'approximation

Soit q un entier. On note $\mathbf{q}$ la fonction de poids constante égale à q. Une méthode naturelle pour trouver une coloration k-impropre de (G,w) consiste à trouver une coloration k-impropre de $(G,\mathbf{q})$. Supposons que cette coloration utilise r couleurs, idéalement $r = \chi_k(G,\mathbf{q})$. les demandes sont ensuite découpées en $\left\lceil \frac{w_{\max}}{q} \right\rceil$ paquets de telle sorte que dans aucun des paquets un sommet ait plus de q demandes ; en utilisant r couleurs par paquets, cela donne une coloration en $r \times \left\lceil \frac{w_{\max}}{q} \right\rceil$ couleurs. Comme $\chi_k(G,w) \geq w_{\max}$, ceci donne un algorithme (r/q) approché.

La prochaine étape est de déterminer le nombre chromatique k-impropre ($1 \leq k \leq 5$) du réseau triangulaire avec poids constant. C'est une borne supérieure du nombre chromatique k-impropre de tout graphe hexagonal.

Théorème 4.5 *Pour le réseau triangulaire R :*

(i) $\chi_1(R,\mathbf{q}) = \left\lceil \frac{5q}{2} \right\rceil$

(ii) $\chi_2(R,\mathbf{q}) = 2q$

(iii) $\chi_3(R,\mathbf{q}) = \left\lceil \frac{3q}{2} \right\rceil$

(iv) $\chi_4(R,\mathbf{q}) = \left\lceil \frac{4q}{3} \right\rceil$

(v) $\chi_5(R,\mathbf{q}) = \left\lceil \frac{7q}{6} \right\rceil$

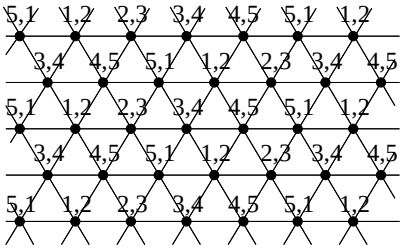


FIG. 4.4 – Une coloration 1-impropre de $(R,\mathbf{2})$

Ces colorations donnent des algorithmes 5/2- , 2-, 3/2-, 4/3- et 7/6-approchés pour la coloration 1-, 2-, 3-, 4- et 5-impropre respectivement des graphes hexagonaux pondérés. Il est cependant possible de faire mieux : une coloration de $(G,\mathbf{q})$ utilisant r couleurs, donne un algorithme de coloration de (G,w) avec un rapport d'approximation meilleur que r/q. Pour cela, au lieu de ne considérer que $w_{\max}$, il faut également considérer le nombre de couleurs imposé par un sommet et $k+1$ de ses voisins. Comme le montre la proposition suivante, celui-ci peut être supérieur à $w_{\max}$. Le graphe ayant un sommet u, appelé **centre**, adjacent à $k+1$ autres sommets, appelés **pointes**, est noté $K_{1,k+1}$.

Proposition 4.6 *Pour toute fonction de poids w,* $\chi_k(K_{1,k+1},w) \geq \dfrac{1}{k+1} \displaystyle\sum_{v \in V(K_{1,k+1})} w(v)$.

Preuve. Soit u le centre de $K_{1,k+1}$ et $v_1,\ldots,v_{k+1}$ ses pointes. Considérons une coloration k-impropre C de $K_{1,k+1}$. Pour tout $1 \leq i \leq k+1$, soit $q(v_i) = |C(v_i) \setminus C(u)|$. La coloration C utilise au moins $M = \max\{q(v_i)+w(u) \mid 1 \leq i \leq k+1\} \geq w(u) + \frac{1}{k+1}\sum_{i=1}^{k+1} q(v_i)$. Or une couleur de $C(u)$ est utilisée au plus par k des v_i car la coloration est k-impropre. Donc $\sum_{i=1}^{k+1} q(v_i) \geq \sum_{i=1}^{k+1} w(v_i) - kw(u)$. Il vient $M \geq \frac{1}{k+1}(w(u) + \sum_{i=1}^{k+1} w(v_i))$. □

On appelle $(k+1)$**-étoile**, ou simplement **étoile**, un sous-graphe de G isomorphe à $K_{1,k+1}$. Le **poids d'une étoile** H est $w(H) = \sum_{v \in V(H)} w(v)$. On note $\theta_k(G,w) = \max\{w(H)/(k+1) \mid H \text{ étoile de } G\}$ et $\omega_k(G,w) = \max\{w_{\max}, \theta_k(G,w)\}$. D'après la Proposition 4.6, $\omega_k(G,w) \leq \chi_k(G,w)$.

Théorème 4.7 *Soient* $\alpha_k(r,q) = \frac{(k+1)r^2}{(k+2)rq-q^2}$ *et* $\beta_k(r,q) = \max\{(k+2)r^2 - rq, (k+1)r^2 + krq\}$.

Étant donné une coloration C *de* $(G,\mathbf{q})$ *avec* r *couleurs, il existe un algorithme polynomial qui colore* (G,w) *avec au plus* $\alpha_k(r,q) \times \omega_k(G,w) + \beta_k(r,q)$ *couleurs.*

Remarques finales. Les Théorèmes 4.5 et 4.7 permettent d'obtenir un algorithme α_k-approché pour calculer la coloration k-impropre ($1 \leq k \leq 5$) d'un graphe hexagonal pondéré. Pour $k = 1$ ou $k = 5$, il est possible d'améliorer le facteur d'approximation en analysant plus précisément la répartition des gros sommets et des sommets contraints. Dans l'article en cours présenté en Annexe B.2, nous avons obtenu des algorithmes $\frac{20}{11}$- et $\frac{41}{36}$-approchés pour la coloration 1- et 5-impropre respectivement des graphes hexagonaux pondérés. Nous avons également des versions distribuées de chacun de ces algorithmes. Parmi les problèmes à considérer, il est intéressant d'examiner d'autres topologies et surtout d'autres modèles de bruit pour lesquels l'intensité dépend de la distance et/ou de la fréquence utilisée.

4.3 Optimisation des buffers dans les réseaux radios et coloration proportionnelle

Dans cette section, je présente un travail réalisé en collaboration avec C. Linharès Salés et H. Rivano, sur un nouveau problème de coloration : la coloration proportionnelle. Ce travail a été présenté à LAGOS 07 au Chili ([HLR08]) et une version longue a été soumise au numéro spécial associé à cette conférence. Elle peut être trouvée dans l'Annexe B.3.

Le problème de coloration proportionnelle est motivé par l'optimisation de l'ordonnancement des connexions radios dans un réseau maillé sans fils. On montre, entre autres, que décider si un graphe admet une coloration proportionnelle est un problème polynomial tandis que trouver l'indice chromatique proportionnel est NP-dur.

4.3.1 Origine du problème

Contexte. Les réseaux radios sans fils maillés **WMN** (Wireless Mesh Network) sont des solutions rentables pour fournir des services omniprésents à très haut débit [AWW05]. Ce sont des réseaux auto-organisés ayant une structure fixe de routeurs sans fils interconnectés, dont le but est d'offrir une connexion Internet aux utilisateurs de téléphones portables. Cette infrastructure est elle-même connectée à Internet par des routeurs spéciaux, appelés point d'accès, comme illustré par la Figure 4.5.

Chaque routeur est équipé d'une carte permettant d'envoyer et de recevoir des paquets sur l'unique fréquence disponible. La fréquence est donc partagée entre les différents nœuds [LZ05]. Par la suite, le réseau est supposé synchrone (c.à.d. que les routeurs ont un temps commun), et l'étude se focalise sur l'état stable du réseau. Le réseau utilise le multiplexage temporel (voir Section 5.1.1.5).

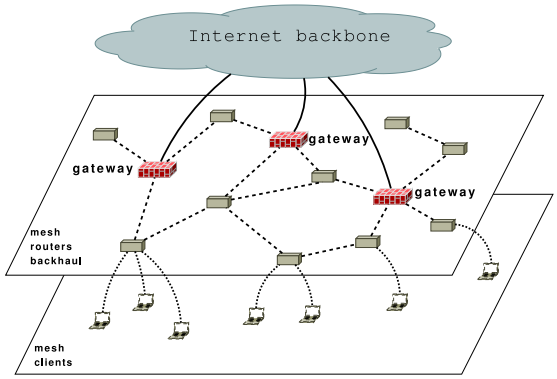


FIG. 4.5 – Topologie d'un réseau WMN : les clients mobiles accèdent à Internet au travers d'un réseau de routeurs et points d'accès sans-fils multi-hop.

Modèle utilisé. La structure fixe d'un réseau WMN est modélisée par un graphe $G = (V, E)$, dans lequel V représente les routeurs et les arcs représentent les connexions possibles entre deux antennes de routeurs. Dans ce travail, les antennes sont supposées *directionnelles*. Allouer, à un lien e, du temps d'accès à la fréquence d'émission correspond à donner de la bande passante à ce lien e. Au cours du temps, la bande passante moyenne d'un lien est son temps d'accès à la fréquence d'émission divisé par le temps total [KMP].

Dans la suite, étant donné un réseau WMN G, on suppose donné un ensemble de requêtes émises périodiquement à l'identique (ces requêtes sont dites à taux constant (constant bit rate : **CBR request**)) ainsi que les chemins sur lesquels elles sont routées. Cela permet de calculer la bande passante nécessaire sur chaque lien pour permettre de satisfaire les requêtes dans ce routage. Chaque requête, pour être satisfaite, nécessite l'établissement des connexions le long du chemin qu'elle utilise et le maintien de ces connexions pendant un temps proportionnel à la taille de la requête. On étudie le réseau dans un état stable, ce qui signifie que les requêtes ne varient pas.

Objectif du problème. L'objectif est de trouver un ordonnancement allouant à chaque lien assez de bande passante. Comme le réseau est supposé dans un état stable du réseau, un ordonnancement va être constitué d'une période de base répétée indéfiniment. Lors de chaque période, un paquet va traverser exactement un lien. Ainsi, à chaque période, les paquets présents dans le réseau avancent tous d'un lien vers leur destination. Autrement dit, les routeurs envoient les paquets qu'ils ont au début de la période et stockent ceux qu'ils reçoivent. Les paquets générés au cours d'une période (qui correspondent à des requêtes qui s'activent durant la période) sont également stockés et attendront la période suivante pour être envoyés. Avec un tel fonctionnement, le temps mis pour satisfaire une requête est égal à la longueur du chemin utilisé multipliée par la durée d'une période.

La Figure 4.6 représente quatre routeurs positionnés sur une ligne. Les routeurs s_1,

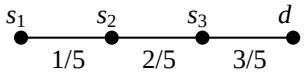


FIG. 4.6 – Un exemple de requêtes à taux constant dans un réseau composé de quatre routeurs.

s_2 et s_3 envoient un paquet à destination du routeur d toutes les cinq unités de temps. Ainsi le lien s_1s_2 doit être activé un cinquième du temps, le lien s_2s_3 deux cinquièmes du temps et le lien s_3d trois cinquièmes du temps, comme indiqué par le poids de chaque arête.

Pour cet ensemble de requêtes CBR, il est possible d'utiliser un ordonnancement dont la période de base est la suivante :

- Les routeurs s_1 et s_3 envoient le paquet généré lors de la période précédente.
- Le routeur s_2 envoie la requête générée lors de la période précédente.
- Le routeur s_2 envoie la requête reçue de s_1 lors de la période précédente.
- Le routeur s_3 envoie la requête reçue de s_2 lors de la période précédente.
- Le routeur s_3 envoie la requête reçue de s_2 (et originaire de s_1) lors de la période précédente.

Cette période dure 5 unités de temps et est optimale car minimale. Elle nécessite un buffer à s_2 et deux à s_3 pour stocker les paquets lors d'une période.

Plus la période de base va être longue, plus le temps de parcours d'une requête va être élevé. De plus, le nombre de paquets à stocker à un routeur est également proportionnel à la durée de la période de base. Or ce nombre de paquets est la taille des buffers à installer aux routeurs. Ainsi, diminuer le coût du réseau et améliorer la qualité de service revient à minimiser la longueur de la période de base de l'ordonnancement périodique. Le problème que je présente est donc le suivant :

Problème 4.8 (Optimisation des buffers dans un réseau WMN)

Entrées :	*Un réseau WMN.*
	Un ensemble de requêtes CBR et leur chemin.
Sortie :	*Un ordonnancement périodique allouant assez de bande passante à chaque lien.*
Objectif :	*Minimiser la durée de la période.*

Il est important de remarquer qu'un ordonnancement doit tenir compte des interférences. Dans notre problème, les routeurs sont supposés équipés d'antennes directionnelles. Les interférences peuvent donc être modélisées par le fait qu'un nœud ne peut être impliqué que dans une connexion à la fois. Ainsi à chaque instant, les connexions seront modélisées par un matching dans G. Le problème d'ordonnancement classique est modélisé par la coloration propre [CBF+03]. Pour modéliser notre problème, un autre type de coloration a été introduit : la coloration proportionnelle.

Définition 4.9 (Coloration proportionnelle) *Étant donné un graphe pondéré (G,w), une* **coloration proportionnelle** *de (G,w) est une fonction $\mathcal{C}:E\to\mathcal{C}(\{1,\dots,c\})$ telle que pour tout $e\in E$, $|\mathcal{C}(e)|\geq cw(e)$ et pour tout $e,f\in E^2$, $e\cap f\neq\emptyset\Rightarrow\mathcal{C}(e)\cap\mathcal{C}(f)=\emptyset$. On appelle* **indice chromatique proportionnel** *de G, $\chi'_\pi(G,w)$, le plus petit nombre de couleurs pour lequel une coloration proportionnelle de (G,w) existe. S'il n'en existe pas, on note : $\chi'_\pi(G,w)=\infty$.*

La Figure 4.7[1], donne un autre exemple de coloration proportionnelle. On voit dans cet exemple qu'une arête peut recevoir plus de couleurs que ce qu'elle demande.

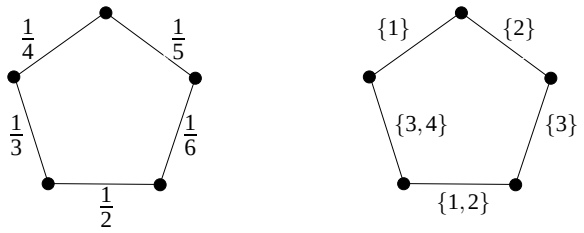


FIG. 4.7 – Un C_5 pondéré et une coloration proportionnelle utilisant quatre couleurs.

4.3.2 Complexité

La coloration proportionnelle est divisée en deux sous-problèmes : le premier consiste à prouver qu'il existe un nombre de couleurs c tel qu'une coloration proportionnelle de (G,w) utilisant c couleurs existe. Le second consiste à trouver le plus petit nombre de couleurs pour lequel une coloration proportionnelle existe. Dans [HLR08], nous avons prouvé que le premier sous-problème est polynomial alors que le deuxième est NP-dur.

Les quelques cas suivants sont simples :

Fait 4.10 *Soit (G,w) un graphe pondéré.*
- *S'il existe un sommet u avec $\sum_{uv\in E(G)}w(uv)>1$ alors $\chi'_\pi(G,w)=\infty$.*
- *Si pour tout $uv\in E(G)$, $w(uv)\leq 1/(\Delta+1)$ alors $\chi'_\pi(G,w)\leq\Delta+1$.*

Comme énoncé précédemment, dans le cas général, le théorème suivant est vrai :

Théorème 4.11 *Soit (G,w) un graphe pondéré avec w à valeur dans $\mathbb{Q}$.*
i) Déterminer s'il existe une coloration proportionnelle de (G,w) est polynomial.
ii) Déterminer l'indice chromatique proportionnel de (G,w) est NP-dur.

En faisant la remarque simple qu'en chaque sommet, le nombre de couleurs utilisées ne peut pas dépasser le nombre total de couleurs, on obtient la borne inférieure suivante :

[1]Figure réalisée par C. Molle

Théorème 4.12 (Borne inférieure) *Soit m le plus petit entier vérifiant les équations suivantes :*

$$\sum_{v \ st \ uv \in E} \lceil mw(uv) \rceil \leq m \tag{4.1}$$

On a $m \leq \chi_\pi(G,w)$

Dans [HLR08], nous montrons également comment utiliser la coloration fractionnaire pour obtenir une borne supérieure sur l'indice chromatique proportionnel. Nous montrons aussi que pour certaines classes de graphes, dont les bipartis (Théorème 4.13), le problème devient polynomial.

Théorème 4.13 *Soit (G,w) un graphe pondéré biparti. S'il existe une solution m aux équations (4.1), alors (G,w) accepte une coloration proportionnelle utilisant m couleurs.*

Remarques finales. Le problème d'ordonnancement qui a motivé l'introduction de la coloration proportionnelle laisse d'autres questions ouvertes. En effet, il serait intéressant de trouver des algorithmes d'approximations pour certaines classes de graphes qui apparaissent en télécommunications telles que les grilles triangulaires. Avec Joseph Yu, nous nous intéressons à la version de la coloration proportionnelle dans laquelle les arêtes ayant la même couleur forment un matching induit, c.à.d. qu'il n'y a pas d'arête entre deux arêtes de même couleur. Ce type de coloration modélise les interférences entre des antennes non directionnelles.

4.4 Conclusion et Perspectives

Dans cette section, j'ai présenté trois problèmes différents concernant le routage de requêtes à l'intérieur d'un réseau. Pour chacun de ces problèmes, les caractéristiques du réseau sont utilisées pour avoir une approche théorique efficace. Le travail effectué sur les deux premiers types de réseaux a abouti à des algorithmes de routage permettant de garantir un écart maximum par rapport à la solution optimale. Dans la troisième section, j'ai présenté des résultats concernant la complexité d'un problème d'optimisation de l'ordonnancement des requêtes dans un réseau WiFi. Il reste encore à trouver des algorithmes d'approximation pour ce dernier problème.

Chapitre 5

Réseaux d'accès et de cœur

Dans ce chapitre, je m'intéresse aux requêtes multicasts dans les réseaux optiques. J'y présente la technologie utilisée dans un réseau tout optique. Ensuite, je m'intéresse au dimensionnement d'un tel réseau. Ce problème est modélisé par un problème de coloration des graphes orientés et je présente les résultats obtenus sur un problème de coloration des graphes orientés qui y est lié : le *directed star colouring (dst)*, entre autres, pour tout graphe orienté D, $dst(D) \leq 2\Delta^- + 1$ et $dst(D) \leq 2\max(\Delta^+(D), \Delta^-(D))$. Ces résultats permettent d'estimer le gain induit par l'utilisation de multicasts au lieu d'unicasts. Enfin, j'étudie des problèmes de routage des multicasts dans les réseaux WDM d'accès et de cœur, pour lesquels je propose plusieurs formulations mathématiques.

Le travail que je présente dans ce chapitre a été réalisé dans le cadre des réseaux optiques mais il peut être généralisé à d'autre types de réseaux tels que les réseaux WiFi. Je présente différents problèmes de routage, ce chapitre concerne donc la couche 3 (réseau) si l'on se réfère au système OSI. Je m'intéresse à des variantes du problème de routage et d'affectation de longueurs d'onde, Problème 5.1. Il s'agit donc d'établir et de maintenir un ensemble de connexions pour satisfaire des requêtes données.

Je commence par présenter les technologies que je considère équipant les réseaux que j'étudie. Ensuite, je m'intéresse au problème du calcul du gain occasionné par l'utilisation de requêtes multicasts à la place de requêtes unicast. Pour cela, je présente une modélisation permettant de ramener ce problème à un problème de coloration de graphe orienté. Je présente de nombreuses avancées concernant cette coloration. Dans la section suivante, je m'intéresse au problème de routage de multicasts proprement dit. Je présente plusieurs formulations pour ce problème, certaines adaptées aux réseaux de cœurs, d'autres aux réseaux d'accès ; un réseau de coeur sert à interconnecter plusieurs réseaux, les données y transitent assemblées sous forme de paquets et à haut débits par des routes réservées (en utilisant des protocoles de type MPLS -Multiprotocol Label

Switching-), tandis qu'un réseau d'accès sert à relier directement les utilisateurs au réseau de cœur et a souvent une faible conenctivité (voir Section 5.2.6 pour plus de détails). Je ne fais pas d'hypothèse sur la structure des réseaux de cœur.

5.1 Introduction sur les réseaux optiques

Dans cette première section, je décris brièvement le fonctionnement d'un réseau optique et le matériel disponible aujourd'hui. Je me suis entre autres basé sur [Alw04, Cor03] et la thèse de D. Coudert [Cou01], ainsi que celle de A. Guitton [Gui05]. Je me place dans le cadre de l'optique guidée, en opposition à l'optique libre.

5.1.1 Technologies optiques

De manière générale, les données sont codées par des bits, c.à.d. des 0 et des 1. Dans un circuit électrique, une technique consiste à représenter le bit 1 par une tension non nulle et le bit 0 par une tension nulle (dans d'autres codages, le bit 0 est représenté par une tension négative). En optique, le principe est le même, le bit 1 est codé par une impulsion lumineuse et le bit 0 par une absence de lumière. La durée de l'impulsion lumineuse dépend du débit de transmission, ainsi à un débit de 10 Gbits/s, étant donné que la vitesse de la lumière est d'environ 300 000 km/s, un bit est codé par un segment dont la longueur est environ 3 cm. Cette hypothèse est par exemple utilisée dans [Siu03].

5.1.1.1 Fibres optiques

Une fibre optique est composée d'un cœur en verre ou en plastique au sein duquel se propage la lumière. Le phénomène physique permettant cette propagation est la réflexion de la lumière. En effet, si l'angle que fait le rayon lumineux avec la paroi de la fibre (appelé **angle d'incidence**) n'est pas trop grand, alors la paroi se comporte comme un miroir et le rayon est totalement réfléchi, voir Figure 5.1. On peut donc considérer qu'une fibre optique se comporte comme un tuyau transmettant une impulsion lumineuse d'une extrémité à une autre. Il existe plusieurs types de fibres. Par exemple, les fibres multi-modes permettent la transmission simultanée de plusieurs rayons lumineux ayant des angles d'incidence différents. Ce procédé provoque cependant des interférences et je ne vais pas le considérer par la suite, je considère donc que les rayons lumineux ne se différencient pas par leur angle d'incidence, ceci est le cas dans les fibres mono-mode. A la place, je considère un autre type de multiplexage : le multiplexage en longueur d'onde que je présente dans la Section 5.1.1.5.

Outre le fait de permettre des transmissions très rapides (à une vitesse de l'ordre de la vitesse de la lumière), la communication optique présente de nombreux avantages. En effet, une transmission optique est insensible au bruit électromagnétique, elle induit une faible distorsion du signal et elle ne s'atténue que peu sur de longues distances (de 0,2 à 1 dB/km sur les longueurs d'onde utilisées, voir Figure 5.2[1]), il faut tout de même utiliser des amplificateurs (c.f. Section 5.1.1.4). L'atténuation est en partie due à des

[1] Source : [Alw04]

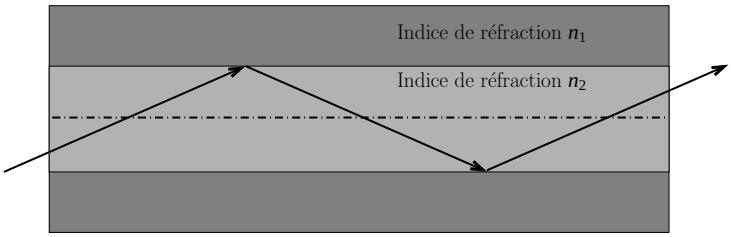


FIG. 5.1 – Principe de transmission dans une fibre optique.

impuretés dans le verre utilisé pour fabriquer la fibre. Les longueurs d'onde utilisées pour transmettre les données s'étalent de 800 à 1600 nanomètres, les plus utilisées sont 850 nm, 1310 nm et 1550 nm ; la lumière visible s'étale quant à elle de 400 à 700 nm.

Enfin, les fibres optiques sont économiques en matière d'énergie utilisée et ne coûtent pas très cher.

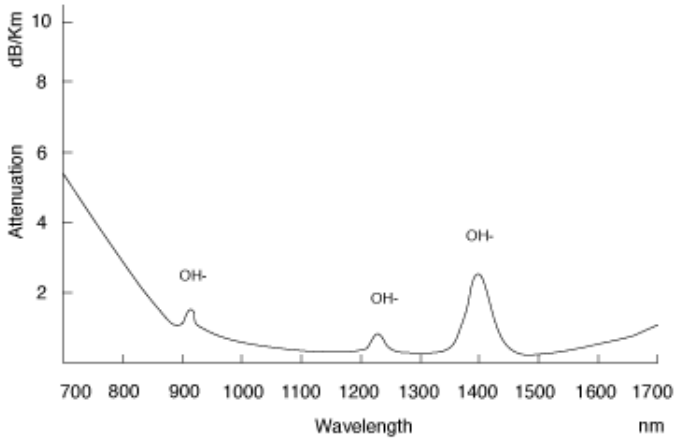


FIG. 5.2 – Atténuation du signal lumineux dans une fibre optique.

5.1.1.2 Émetteurs

En dehors du réseau, les données existent sous forme électromagnétique. Les impulsions lumineuses monochromatiques permettant de coder les informations sont injectées dans la fibre optique à l'aide de diodes laser. Ce sont les caractéristiques de la diode laser utilisée qui définissent la longueur d'onde sur laquelle l'impulsion lumineuse est émise.

Il y a de nombreux types de diodes laser. Je ne cite que les VCSEL (Vertical Cavity Surface Emitting Laser) qui sont des lasers émettant perpendiculairement à la surface et qui sont utilisées pour les applications de communication par fibre optique à courte

portée telles que le protocole Gigabit Ethernet. Les diodes VCSEL présentent l'avantage d'être plus facile à fabriquer et peuvent être disposées en matrices. Ces diodes sont également utilisées pour les réseaux en espace optique libre, réseaux dont il est question dans la thèse de D. Coudert [Cou01].

5.1.1.3 Récepteurs

A une extrémité de la fibre se trouvent les diodes laser et à l'autre se trouvent les récepteurs. Leur rôle est de transformer une impulsion lumineuse en une impulsion électrique. Il est possible de faire cela avec des diodes photosensibles, leur fonction étant de convertir de la lumière en courant électrique. Le courant induit est ensuite amplifié et soumis à un test de seuil pour savoir quel bit (0 ou 1) il code.

5.1.1.4 Amplificateurs

Au cours de sa transmission dans une fibre optique, un signal s'atténue, le phénomène correspondant est appelé la dispersion Rayleigh. On considère généralement que l'atténuation est de 0.2 dB par kilomètre et il est communément admis qu'un signal peut être correctement interprété tant qu'il a été atténué de moins de 20 dB. Ainsi, pour permettre des transmissions sur de longues distances, il est nécessaire de réémettre le signal environ tous les 100 km. L'inconvénient de réémettre le signal est que cela induit un délai supplémentaire car il faut faire une conversion opto-électronique avant de le réémettre. Il existe une autre solution qui consiste à utiliser des amplificateurs optiques. Ce sont des dispositifs permettant d'amplifier un signal lumineux sans à avoir à le convertir en signal électrique. Il existe plusieurs types d'amplificateurs optiques dont les amplificateurs à fibres dopées et les amplificateurs à effet Raman.

L'amplification optique a cependant des limites, en effet, à chaque amplification, du bruit est ajouté au signal, *il faut* donc, au bout d'un certain temps, *réémettre le signal*. Les amplificateurs à effet Raman induisent moins de bruit que les amplificateurs à fibres dopées.

En plus d'atténuer les impulsions lumineuses, la dispersion Rayleigh les déforme. La Figure 5.3[2] illustre un signal optique lors de son émission et après sa transmission par une fibre optique. Pour permettre une bonne réception il est donc nécessaire de remodeler les impulsions lumineuses.

Un troisième problème dont il faut tenir compte est la désynchronisation des signaux. En effet, la vitesse de propagation d'un signal lumineux dépend de sa longueur d'onde et de son angle d'incidence. Pour garantir que le délai entre deux impulsions lumineuses censées arriver simultanément reste raisonnable, il est nécessaire de les resynchroniser régulièrement.

La déformation et la désynchronisation des signaux ne sont pas critiques par rapport à l'atténuation et il est suffisant de remodeler et de synchroniser un signal tous les 500 km. Il est donc possible de combiner cela avec la réémission rendue nécessaire par l'amplification du bruit lors de l'amplification du signal.

[2] inspirée de [Gui05]

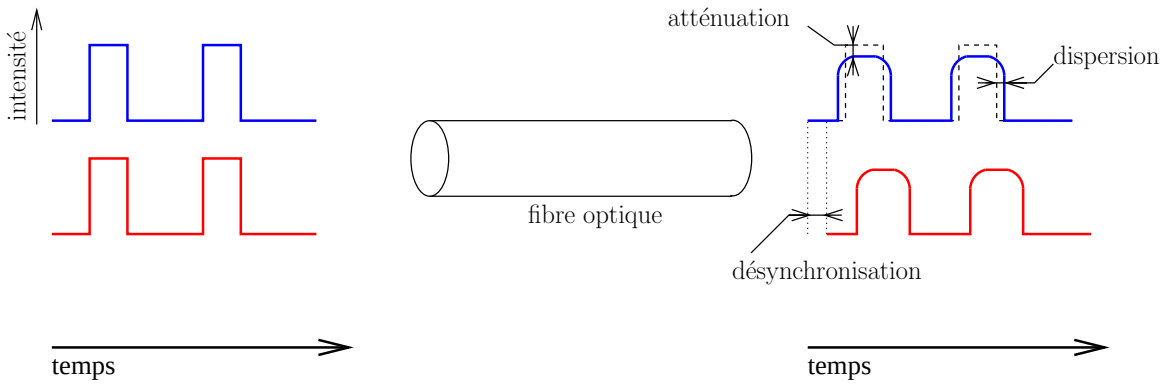


FIG. 5.3 – Atténuation, déformation et désynchronisation des signaux lumineux.

5.1.1.5 Multiplexage

Le multiplexage est la capacité à transmettre sur un seul support (dans notre cas la fibre optique) des données provenant de plusieurs sources. Il existe plusieurs types de multiplexages, ici j'en présente deux.

Multiplexage temporel. Il s'agit d'allouer le temps d'utilisation de la fibre optique entre les différentes sources qui ont des données à transmettre. Ainsi l'accès à la fibre optique est partagé en intervalles de temps qui sont répartis à chacune des sources, c.f. Figure 5.4.

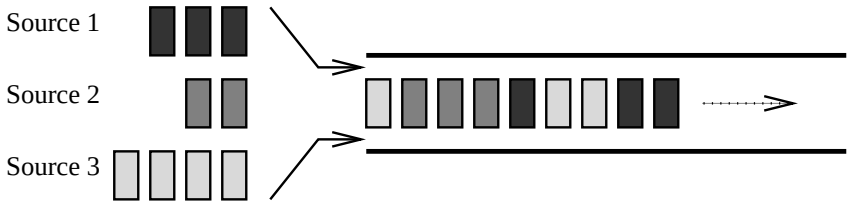


FIG. 5.4 – Multiplexage temporel.

Multiplexage en longueur d'onde. Dans le cas du multiplexage en longueur d'onde (Wavelength Division Multiplexing) WDM, les données sont émises dans la fibre optique sur différentes longueurs d'onde. Le multiplexage en longueur d'onde nécessite l'usage d'un multiplexeur en entrée, pour réunir les longueurs d'onde à transmettre en un seul signal lumineux, et d'un démultiplexeur en sortie, pour les dissocier, c.f. Figure 5.5.

Dans le cas des réseaux tout optique, les multiplexeurs/démultiplexeurs peuvent être des prismes. Un réseau permettant le multiplexage en longueurs d'onde est équipé

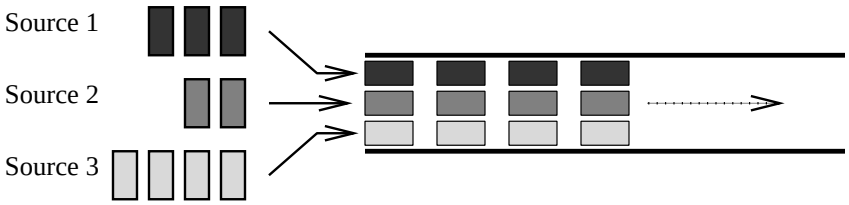


FIG. 5.5 – Multiplexage en longueurs d'onde.

d'**ADM** (Add/Drop Multiplexer) à ses nœuds, c.f. Figure 5.6[3], on parle d'**OADM** (Optical Add/Drop Multiplexer).

Si le réseau n'est pas tout optique, les ADM peuvent effectuer une conversion optoélectronique, ce qui peut servir à changer la longueur d'onde d'un signal reçu lors de sa réémission.

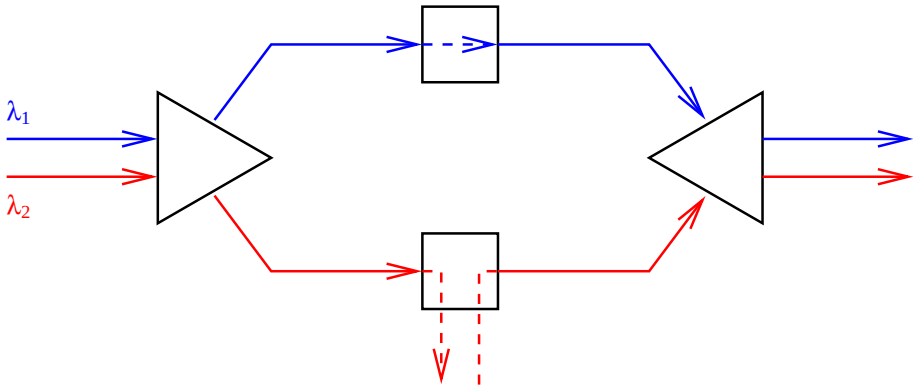


FIG. 5.6 – Fonctionnement d'un OADM.

De nos jours, en laboratoire, il est possible de transmettre jusqu'à 1024 longueurs d'onde sur une même fibre. A 10Gb, en laboratoire, 320 longueurs d'onde sont utilisées tandis que des réseaux utilisant 160 longueurs d'onde sont d'ores et déjà exploités. Plus le nombre de longueurs d'onde est élevé, plus l'écart entre deux longueurs d'onde est faible. L'utilisation de peu de longueurs d'onde (16 par exemple) permet d'utiliser des infrastructures moins sophistiquées et donc moins coûteuses. Plus de détails peuvent être trouvés dans les deux études [Muk06, RS02].

5.1.1.6 Splitters

Dans certaine applications - le routage multicast en particulier - il est intéressant de dupliquer un signal arrivant sur une fibre pour l'envoyer sur plusieurs fibres. Une

[3] inspirée de [Gui05]

solution consiste à convertir le signal arrivant en un signal électrique et de le réémettre sur les fibres sortantes. Pour éviter cette conversion, il est possible d'utiliser des splitters.

Les splitters que je considère dans les Sections 5.2.2 et 5.2.6 sont des composants optiques passifs. Ils reçoivent un signal en entrée sur une fibre donnée et divisent ce signal sur plusieurs fibres en sortie. Dans ce type de splitters, tous les signaux de toutes les longueurs d'onde de la fibre d'entrée sont divisés sur les fibres de sortie. On appelle le nombre de fibres de sorties la taille du splitter. Un splitter de taille k est appelé un $1:k$ splitter. La puissance du signal d'entrée est répartie sur les différentes fibres de sorties, on considère qu'un splitter $1:2$ induit une atténuation de 3 à 3,5 dB. Dans la Section 5.2.1, je considère des splitters plus performants permettant de sélectionner les longueurs d'onde dupliquées. J'appelle de tels splitters des **splitters en longueurs d'onde**.

5.1.2 Modélisation d'un réseau optique WDM

Un réseau optique WDM est représenté par un graphe orienté $D = (V,E)$. A chaque routeur (ou système) du réseau physique est associé un sommet du graphe $V = \{v_1, v_2, \ldots, v_n\}$. De manière similaire, chaque fibre optique du réseau est associée à un arc du graphe orienté $E = \{e_1, e_2, \ldots, e_m\}$. La capacité de chaque arc représente le nombre de longueurs d'onde disponibles. En effet, on fait l'hypothèse simplificatrice que toutes les longueurs d'onde ont la même capacité et que les requêtes sont routées seules sur une longueur d'onde (c.à.d. qu'il n'y a pas de *groupage*, c.f. Section 6.2.1), sauf si le contraire est spécifié.

Les connexions et les fibres optiques sont supposées orientées et le trafic asymétrique. Le trafic est composé d'un ensemble de requêtes qui sont caractérisées par leur source et leur destination. L'objectif est d'établir des connexions pour ces requêtes et de les maintenir. L'ensemble des longueurs d'onde disponibles dans le réseau est noté $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_W\}$ avec $W = |\Lambda|$.

Dans ce contexte, le problème de routage présenté dans la Section 2.2.2 se complexifie. En effet, en plus de trouver un ensemble de chemins pour les requêtes, il faut également assigner à chaque chemin une longueur d'onde. Excepté dans la Section 6.2, je fais l'hypothèse de continuité en longueur d'onde, autrement dit les données sont transmises par un signal lumineux dont la longueur d'onde ne varie pas le long du chemin. Le problème de routage peut être présenté comme suit :

Problème 5.1 (Problème de routage et d'affectation de longueurs d'onde (RWA))

Entrées :	*Un réseau optique WDM.* *Un ensemble de requêtes.*
Sorties :	*Un chemin réalisable pour chaque requête.* *Une longueur d'onde associée à chaque chemin de telle sorte qu'aucun chemin utilisant une même arête n'ait la même longueur d'onde.*
Objectif :	*Minimiser le coût du routage.*

5.2 Multicast

Dû au développement des applications multi-utilisateurs, telles que la télévision haute définition ou sur Internet, les vidéoconférences, les applications à la demande et les jeux en réseau, il est crucial d'économiser le plus possible de bande passante en ne transmettant pas de données redondantes. C'est pour cette raison que de plus en plus d'attention se porte sur les requêtes ayant une source et plusieurs destinations. On appelle de telles requêtes des requêtes multicasts, une requête ayant une source et une seule destination est, elle, appelée requête **unicast**. Par abus de langage, j'appelle une requête multicast un **multicast**. A l'heure actuelle, les multicasts sont le plus souvent effectués sous la forme d'autant de requêtes unicast (une source, une destination) qu'ils ont de destinations. Cette technique, appelée **multi-unicast**, encombre à la fois la source et le réseau. L'intérêt du véritable multicast est de permettre d'économiser de la bande passante sur une partie du chemin à parcourir (les sections de routage communes) et de permettre à la source de ne pas réémettre plusieurs fois le même message. En effet, il suffit d'envoyer le message sur une longueur d'onde et de le dupliquer à l'aide d'un splitter là où c'est nécessaire (c.à.d. à chaque point de séparation des routes à emprunter) pour atteindre chacune des destinations. L'utilisation de splitters permet donc d'étendre les possibilités de routage en permettant, en plus de l'utilisation de chemins, l'utilisation d'arbres pour router les requêtes. Un lecteur intéressé peut se référer aux articles [MZQ98, SM99] pour plus de détails sur les apports du multicast, ainsi qu'à la thèse de A. Guitton : Communications multicast [Gui05].

Déploiement du multicast. Comme dans le cas du routage des unicasts, il est nécessaire de décrire le routage et l'adressage des paquets. Pour les unicasts, ce problème est résolu de manière efficace. Pour les multicasts, afin de décrire l'adressage et le routage des paquets, c'est le modèle proposé par Deering dans [Dee91] qui a été adopté par l'organisme de standardisation de l'IETF. Dans sa thèse, A. Guitton, présente ce modèle ainsi que ses limites. Il présente et compare d'autre protocoles dont les protocoles Xcast [BFI+07], Xcast+ [SKPK01], GXcast [BGC04] et Linkcast.

Les protocoles cités précédemment nécessitent l'interprétation des en-têtes des paquets transmis, or cela n'est pas réalisable dans un réseau tout optique très haut débit. Dans ce chapitre, j'envisage donc le cas où le routage est fait physiquement.

Objectifs. Il est important de pouvoir implémenter un routage de multicasts efficace tant en termes de bande passante utilisée que de coût d'infrastructure. En effet, l'installation de splitters induit un surcoût au réseau. Cela mène à l'étude de plusieurs problèmes dont le problème de **Dimensionnement de Réseaux** et le problème **Routage de Multicasts et Affectation de Longueurs d'Ondes** (**MultiCast Wavelength Routing and Assignment**) **MC-RWA**. Deux surveys sont disponibles sur ce dernier problème : [HCT02] et [Rou03] qui est une extension du Problème 5.1.

Problème 5.2 (Dimensionnement de réseau)
Entrée : *Caractéristiques des requêtes à router.*
Sortie : *Caractéristiques suffisantes du réseau.*
Objectif : *Minimiser le coût du réseau.*

Problème 5.3 (Routage de Multicasts et Affectation de Longueurs d'Ondes : MC-RWA)
Entrées : *Un réseau avec des coûts et des capacités sur les liens.*
Un ensemble de requêtes multicasts.
Sortie : *Un routage (arbres et longueur d'onde) des requêtes multicasts.*
Objectif : *Minimiser le coût.*

Plan. De nos jours, il existe des splitters en longueurs d'onde mais ils ne sont pas encore utilisés dans les réseaux existants ou, si ils le sont, ils le sont de manière anecdotique. Dans la Section 5.2.1, j'étudie le problème 5.2 en évaluant l'impact qu'aurait l'utilisation des splitters en longueurs d'onde. Nous y étudions le nombre de longueurs d'onde nécessaire au routage des multicasts en fonction de leur nombre et du nombre de destinataires. Cela permet de quantifier l'économie induite par l'installation d'un splitter en longueurs d'onde.

Par la suite, dans les Sections 5.2.2 et 5.2.6, je m'intéresse au Problème 5.3 MC-RWA dans des réseaux réels. Ainsi je ne considère que les splitters dupliquant l'intégralité d'une fibre, tels qu'ils sont décrits dans la Section 5.1.1.6. Cette hypothèse a très peu été prise en compte car elle est très contraignante, elle semble pourtant bien plus réaliste d'après les contacts que nous avons eus avec Nicolas Gagnon de NORTEL Networks - EXFO. Dans la Section 5.2.2, je décris plusieurs formulations mathématiques permettant de résoudre MC-RWA dans un réseau optique de cœur. Je considére l'objectif de minimiser le nombre de requêtes rejetées puis celui de minimiser le nombre d'utilsateurs non satisfaits. Dans la Section 5.2.6, je m'intéresse à MC-RWA dans les réseaux d'accès, l'objectif de cette section étant de prendre en compte les caractéristiques des réseaux d'accès pour pouvoir améliorer le temps de résolution.

5.2.1 Dimensionnement d'un réseau en étoile

Cette section concerne un travail soumis à Combinatorics, Probability and Computing, réalisé avec O. Amini, F. Havet et S. Thomassé sur un réseau bien particulier : un réseau en étoile. L'article soumis peut être trouvé dans l'Annexe C.1.

Un réseau en étoile est un réseau composé de plusieurs systèmes, par exemple des ordinateurs, tous reliés à un nœud central par le même nombre de fibres optiques. On note l'ensemble des systèmes V et le nombre de fibres n, ces informations caractérisent notre réseau. On suppose de plus que le réseau est un réseau WDM, c'est-à-dire que les transmissions se font sur différentes longueurs d'onde à l'intérieur d'une même fibre (Section 5.1.1.5). Enfin, on suppose que le nœud central est un transmetteur tout optique qui peut renvoyer sur un ensemble de fibres un message reçu mais qui ne peut pas en modifier la longueur d'onde. Autrement dit, le nœud central est un splitter

capable de dupliquer un signal optique sans en modifier la longueur d'onde (Section 5.1.1.6).

Nous nous sommes intéressés au dimensionnement d'un tel réseau. Les entrées de notre problème sont le nombre de fibres reliant chaque système au nœud central et un ensemble de contraintes sur les multicasts à router. Les contraintes peuvent porter sur le nombre maximum de multicasts envoyés et reçus par chaque système ou encore sur le nombre de destinataires de chaque multicast. L'objectif est de trouver le nombre minimum de longueurs d'onde nécessaires pour assurer l'existence d'un routage pour tout ensemble de multicasts satisfaisant les contraintes que l'on s'est fixées.

Ces travaux s'effectuent dans la lignée de ceux réalisés par R. Brandt et T.F. Gonzalez [BG05]. Une des conséquences de ce travail est de permettre *de calculer l'impact d'un splitter en longueurs d'onde sur le coût d'un réseau.* Cet impact sur le coût a déjà été évalué dans l'article [MZQ98]. Les auteurs y étudient également des réseaux équipés de splitters en longueurs d'onde et ont une approche empirique. Nos travaux ont également un fort intérêt théorique puisque nous avons grandement amélioré les bornes supérieures sur le nombre maximum de couleurs nécessaires dans un problème de coloration des graphes orientés.

Problème 5.4 (Dimensionnement d'un réseau en étoile)

Entrées :	*Nombre de fibres.* *Nombre de multicasts envoyés par un système.* *Nombre de multicasts reçus par un système.* *Nombre maximum de destinataires d'un multicast.*
Sortie :	*Un nombre de longueurs d'onde suffisant pour router tout ensemble de requêtes envisagées.*
Objectif :	*minimiser le nombre de longueurs d'onde.*

Plan. Je présente en premier lieu les résultats dans le cas où chaque système est relié par une seule fibre au nœud central et envoie au plus un multicast. Cette étude est effectuée dans le cas où le nombre de multicasts reçus par chaque nœud est borné, puis quand pour un nœud donné la somme du nombre de multicasts reçus et envoyés est bornée, et enfin quand le nombre de multicasts reçus et le nombre de multicasts envoyés sont tous les deux bornés. Dans ces différents cas, le problème se ramène à l'étude d'une coloration naturelle des graphes orientés. Dans la dernière section, je présente le cas plus général où il y a plusieurs fibres reliant chaque système au nœud central et où le nombre maximum de multicasts envoyés par un même système est borné. Dans ce cas, le problème est une extension du problème de coloration précédent pour les graphes orientés labélisés.

5.2.1.1 Une fibre, un multicast

Dans cette section, je suppose que chaque système est relié par une unique fibre au nœud central et qu'il envoie un multicast $M(v)$ à un ensemble de systèmes $S(v) : M(v) = \{v, S(v)\}$. Trouver le nombre minimum de longueurs d'onde nécessaires au routage de

cet ensemble de multicasts peut se ramener simplement à un problème de coloration des arêtes d'un graphe orienté. Pour cela, on considère le graphe orienté D ayant pour sommet l'ensemble V des systèmes et tel que le voisinage sortant d'un sommet $v \in V$ est $S(v)$. On peut remarquer que ce graphe orienté peut avoir des cyles de longueur deux mais qu'il n'a pas d'arêtes parallèles. De plus, le degré entrant d'un sommet est le nombre de multicasts reçus par le système correspondant à ce sommet. Le problème initial est équivalent à trouver le nombre minimum de couleurs pour colorer les arêtes du graphe orienté tel que pour chaque sommet, chaque arête entrante ait une couleur distincte de toutes les autres arêtes incidentes à ce sommet (voir Figure 5.7 pour un exemple de coloration valide à un sommet). Pour obtenir une solution à notre problème initial, il suffit de considérer que chaque couleur correspond à une longueur d'onde différente.

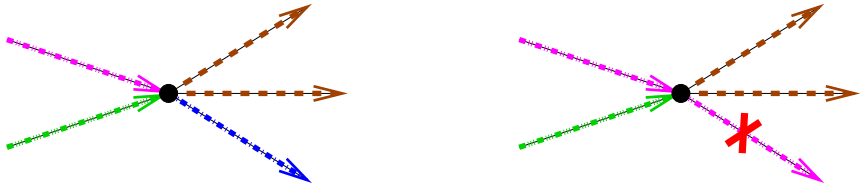


FIG. 5.7 – Une fibre, un multicast : deux exemples de coloration, l'un valide (à gauche), l'autre non (à droite).

Une autre formulation est la suivante : il faut trouver le plus petit entier k tel qu'il existe une fonction $\phi : V(D) \to \{1, \ldots, k\}$ telle que :

(i) $\phi(uv) \neq \phi(vw)$;

(ii) $\phi(uv) \neq \phi(u'v)$.

Une telle fonction est appelée un **directed star k-colouring**.

La **directed star arboricity** d'un graphe orienté D, notée **dst(D)**, est l'entier minimum k tel qu'il existe un directed star k-colouring. Cette notion a été introduite par Guiduli [Gui97] et est analogue à la *star arboricity*, une décomposition des graphes introduite par I. Algor et N. Alon dans [AA89].

Notations. Une **arborescence** est un graphe orienté connexe tel que chaque sommet a un degré entrant de un excepté un que l'on appelle la **racine** et qui a degré entrant zéro.

Une **forêt** est une union disjointe d'arborescences.

Une **étoile** est une arborescence dans laquelle la racine domine tous les sommets. Une **galaxie** est une forêt d'étoiles.

Remarque 5.5 Il est clair que dans un directed star colouring, tous les sommets ayant la même couleur forment une galaxie. On obtient ainsi une troisième façon de définir notre problème : *la directed star arboricity d'un graphe orienté D est le nombre minimum de galaxies permettant de partitionner les arcs de D.*

Complexité. Dans [AHHT07], nous montrons que trouver la directed star arboricity d'un graphe orienté est un problème *NP*-dur. Nous le montrons en ramenant le problème d'arêtes coloration des graphes cubique à notre problème dans le cas où le graphe orienté a un degré entrant et sortant d'au plus 2.

Cela implique que le problème suivant est NP-dur, alors même que le nombre de longueurs d'onde nécessaires est au plus cinq :

Problème 5.6 (Dimensionnement d'un réseau en étoile, complexité)

Entrées :	*Chaque système est relié par une fibre au sommet central.* *Chaque système est source d'au plus un multicast.* *Chaque système reçoit au plus deux multicasts.* *Chaque multicast est adressé à au plus deux systèmes.*
Sortie :	*Nombre de longueurs d'onde nécessaires pour router un ensemble de requêtes.*
Objectif :	*Minimiser le nombre de longueurs d'onde nécessaires.*

Sachant ce résultat, il est intéressant de trouver des bornes supérieures sur le nombre de longueurs d'onde nécessaires dans le Problème 5.4 en fonction des différents paramètres du problème. La section suivante s'intéresse au cas où le degré entrant est borné.

Degré entrant borné.

Problème 5.7 (Dimensionnement d'un réseau en étoile, degré entrant borné)

Entrées :	*Chaque système est relié par une fibre au sommet central.* *Chaque système est source d'au plus un multicast.* *Chaque système reçoit au plus k multicasts.*
Sortie :	*Un nombre de longueurs d'onde suffisant pour router tout ensemble de requêtes envisagées.*
Objectif :	*Minimiser le nombre de longueurs d'onde.*

R. Brandt et T.F. Gonzalez [BG05] ont montré que $dst(D) \leq \lceil 5\Delta^-/2 \rceil$. Dans le cas où $\Delta^- = 1$ cette borne est serrée car les circuits impairs ont une star arboricity de 3. Cependant pour de plus grandes valeurs de Δ^- il est possible de l'améliorer. Nous conjecturons que si $\Delta^- \geq 2$, alors $dst(D) \leq 2\Delta^-$.

Conjecture 5.8 Tout graphe orienté D de degré entrant au plus $\Delta^- \geq 2$ vérifie $dst(D) \leq 2\Delta^-$.

Cette conjecture serait serrée car Brandt [Bra03] montre que pour tout Δ^-, il existe un graphe orienté acyclique D_{Δ^-} de degré maximum Δ^- et tel que $dst(D_{\Delta^-}) = 2\Delta^-$.

On peut remarquer qu'afin de prouver cette conjecture, il est suffisant de la prouver pour $\Delta^- = 2$ et $\Delta^- = 3$. En effet, un graphe orienté de degré entrant maximum $\Delta^- \geq 2$ admet une partition de ses arêtes en $\Delta^-/2$ graphes orientés de degré entrant maximum 2 si Δ^- est pair et en $(\Delta^- - 1)/2$ graphes orientés de degré maximum 2 et un de degré entrant maximum 3 si Δ^- est impair. Pour trouver cette partition, il suffit d'enlever au graphe orienté un sous-graphe eulérien maximum de manière récursive.

Nous montrons dans [AHHT07] le théorème suivant :

Théorème 5.9 $\forall D, dst(D) \leq 2\Delta^-(D)+1$ *et la Conjecture 5.8 est vraie pour les graphes orientés acycliques.*

Remarque 5.10 Il est important de remarquer que l'on se restreint au cas où il n'y a pas d'arêtes multiples. En effet, si les arêtes multiples sont autorisées, les bornes mentionnées ci-dessus ne sont plus valables. Etant donné un entier Δ^-, le multigraphe orienté T_{Δ^-} formé de trois sommets u, v et w et de Δ^- arcs parallèles uv, vw et wu, a $dst(T_{\Delta^-}) = 3\Delta^-$.

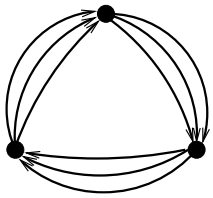


FIG. 5.8 – Le multigraphe orienté T_{Δ^-} nécessite $3\Delta^-$ couleurs.

De plus cet exemple est extrémal car tout multigraphe orienté vérifie $dst(D) \leq 3\Delta^-$. Cela peut être prouvé par récurrence : étant donné un sommet v de degré sortant au plus Δ^- dans une composante fortement connexe terminale (une composante fortement connexe C étant **terminale** si elle n'a pas d'arc sortant).

Si v n'a pas de voisin entrant, il est isolé et on peut le supprimer. Si v a des voisins entrants, on considère un arc uv. Sa couleur doit être différente des couleurs des $d^-(u)$ arcs entrant en u, des $d^+(v)$ arcs sortant de v et des $d^-(v)-1$ autre arcs entrant en v, soit au plus $3\Delta^- - 1$ arcs au total. Ainsi on peut enlever l'arc uv, appliquer l'induction, et étendre la coloration à uv. Pour les multigraphes orientés, la borne $dst(D) \leq 3\Delta^-$ est donc serrée.

Degré total borné. Nous étudions ensuite la directed star arboricity d'un graphe orienté de degré total borné, (je rappelle que le degré total d'un sommet v est $d(v) = d^-(v)+d^+(v)$ c.f. Section 2.1).

Problème 5.11 (Dimensionnement d'un réseau en étoile, degré total borné)

Entrées :	*Chaque système est relié par une fibre au sommet central. Le nombre de multicasts reçus et envoyés par chaque système est au plus* Δ.
Sortie :	*Un nombre de longueurs d'onde suffisant pour router tout ensemble de requêtes envisagées.*
Objectif :	*Minimiser le nombre de longueurs d'onde.*

On note $\mu(G)$, la multiplicité maximale d'une arête dans un multigraphe G. Par le Théorème de Vizing, on sait que tout multigraphe peut être coloré avec $\Delta(G)+\mu(G)$

couleurs de telle manière que toutes les arêtes incidentes aient deux couleurs différentes. Comme le multigraphe sous-jacent à un graphe orienté a une multiplicité d'au plus deux, tout graphe orienté D est tel que $dst(D) \leq \Delta + 2$.

Nous faisons la conjecture suivante :

Conjecture 5.12 *Soit D un graphe orienté de degré maximum $\Delta \geq 3$. On a $dst(D) \leq \Delta$.*

Cette conjecture serait serrée puisque tout graphe orienté ayant $\Delta = \Delta^-$ a une star arboricity d'au moins Δ. Nous montrons dans [AHHT07] le théorème suivant :

Théorème 5.13 *La conjecture 5.12 est vraie quand $\Delta = 3$.*

Degré entrant et degré sortant borné. Une première approche pour résoudre les Conjectures 5.8 et 5.12 serait de les prouver dans le cas où les degrés entrant et sortant sont bornés.

Problème 5.14 (Dimensionnement d'un réseau en étoile, degré entrant et degré sortant borné)

Entrées :	*Chaque système est relié par une fibre au sommet central.* *Chaque système est source d'au plus un multicast.* *Chaque système reçoit au plus k multicasts.* *Chaque multicast est adressé à au plus k systèmes.*
Sortie :	*Un nombre de longueurs d'onde suffisant pour router tout ensemble de requêtes envisagées.*
Objectif :	*Minimiser le nombre de longueurs d'onde.*

Dans ce cadre là, on a les conjectures plus faibles suivantes :

Conjecture 5.15 *Pour tout $k \geq 2$, et tout graphe orienté D, si $\max(\Delta^-, \Delta^+) \leq k$ alors $dst(D) \leq 2k$.*

Ces conjectures sont vraies et loin d'être serrées pour k grand. En effet, B. Guiduli [Gui97] a montré que si $\max(\Delta^-, \Delta^+) \leq k$ alors $dst(D) \leq k + 20\log(k) + 84$.

La preuve de B. Guiduli se base sur le fait que si les degrés entrant et sortant sont bornés, la coloration d'un arc ne dépend que des colorations d'un nombre limité d'autres arcs. Cette dépendance bornée permet d'utiliser le Lemme Local de Lovász. Cette idée a initialement été utilisée par I. Algor et N. Alon [AA89] pour la star arboricity des graphes non orientés. On peut également remarquer que le résultat de Guiduli est presque serré puisqu'il existe des graphes orientés D avec $\max(\Delta^-, \Delta^+) \leq k$ et $dst(D) \geq k + \Omega(\log(k))$ (voir [Gui97]).

Enfin, il est intéressant de remarquer que comme pour la Conjecture 5.8, il est suffisant de prouver la Conjecture 5.15 pour $k = 2$ et $k = 3$. Nous montrons dans [AHHT07] le théorème suivant :

Théorème 5.16 *La Conjecture 5.15 est vraie pour $k = 2$, ce qui l'implique pour toute valeur paire de k.*

Il ne reste donc plus qu'à la prouver pour $k = 3$.

Directed star arboricity acircuitique. A. Pinlou et É. Sopena [PS06] ont étudié une version plus forte de la directed star arboricity, appellée directed star arboricity acircuitique. Il rajoute la condition que tout circuit doit avoir trois couleurs distinctes. Cette notion ne s'applique qu'aux graphes orientés asymétriques, c'est-à-dire aux graphes orientés sans circuit de longueur 2. En effet, un circuit de longueur 2 ne peut pas recevoir 3 couleurs. Les auteurs ont montré que la directed star arboricity acircuitique d'un graphe de degré maximum 3 est au plus 4. Dans [AHHT07], nous proposons une preuve plus courte de ce résultat.

5.2.1.2 Plusieurs fibres, plusieurs multicasts

Nous étudions, dans cette section, le cas plus général où le nœud central est relié à chaque système par n fibres optiques et où chaque système peut envoyer plusieurs multicasts.

On note $M_1(v),\dots,M_{s(v)}(v)$ les $s(v)$ multicasts envoyés par un système $v \in V$. Pour $1 \leq i \leq s(v)$, l'ensemble des nœuds destinataires du multicast $M_i(v)$ est noté $S_i(v)$. Le problème est toujours de trouver le nombre minimum de longueurs d'onde pour assurer l'existence d'un routage des multicasts.

Problème 5.17 (Dimensionnement d'un réseau en étoile multifibre)

Entrées :	*Chaque système est relié par n fibres au sommet central.* *Chaque système est source d'au plus m multicasts.* *Chaque système reçoit au plus k multicasts.*
Sortie :	*Un nombre de longueurs d'onde suffisant pour router tout ensemble de requêtes envisagé.*
Objectif :	*Minimiser le nombre de longueurs d'onde.*

De nouveau, on peut modéliser ce problème par un problème de coloration d'un graphe orienté, mais cette fois on a besoin d'un graphe orienté étiqueté.

Comme précédemment, on considère un graphe orienté ayant V comme ensemble de sommets. Pour tout multicast $M_i(v) = (v, S_i(v))$, $v \in V$, $1 \leq i \leq s(v)$, on ajoute l'ensemble d'arcs $A_i(v) = \{vw, w \in S_i(v)\}$ étiquetés i. L'étiquette d'un arc a est noté $l(a)$.

On remarque que pour tout couple de sommets (u,v) et pour toute étiquette i il y a au plus un arc uv étiqueté i. Si le nombre maximum de multicasts envoyés par un système est m, alors il y a au plus m étiquettes différentes sur les arcs. Un tel graphe orienté est dit **m-étiqueté**.

Notre objectif est de trouver un **n-fiber wavelength assignment** de D, qui est un fonction $\Phi : A(D) \to \Lambda \times \{1,\dots,n\} \times \{1,\dots n\}$ qui associe tout arc uv à un triplet $(\lambda(uv), f^+(uv), f^-(uv))$ tel que :

(i) $\forall uv, vw \in E, (\lambda(uv), f^-(uv)) \neq (\lambda(vw), f^+(vw))$;

(ii) $\forall uv, u'v \in E, \lambda(uv), f^-(uv)) \neq (\lambda(u'v), f^-(u'v))$;

(iii) $\forall vw, vw' \in E$, si $l(vw) \neq l(vw')$ alors $(\lambda(vw), f^+(vw)) \neq (\lambda(vw'), f^+(vw'))$.

$\lambda(uv)$ correspond à la longueur d'onde sur laquelle le message originaire de u et destiné à v est envoyé. $f^+(uv)$ et $f^-(uv)$ sont les numéros des fibres utilisées en u et v respectivement.

Les équations précédentes peuvent être décrites de la manière suivante :

(i) correspond au fait que tout multicast correspondant à un arc entrant v et tout multicast correspondant à un arc quittant v doivent : soit avoir une longueur d'onde différente, soit être routés sur des fibres différentes ;

(ii) correspond au fait que deux multicasts correspondants à deux arcs entrant v doivent avoir soit une longueur d'onde différente soit être routés sur des fibres différentes ;

(iii) correspond au fait que deux multicasts correspondants à deux arcs quittant v et n'ayant pas la même étiquette doivent avoir soit une longueur d'onde différente soit être routés sur des fibres différentes.

Le problème consiste à trouver la cardinalité minimale $\lambda_n(D)$ de Λ telle qu'il existe un ***n*-fiber wavelength assignment** de D.

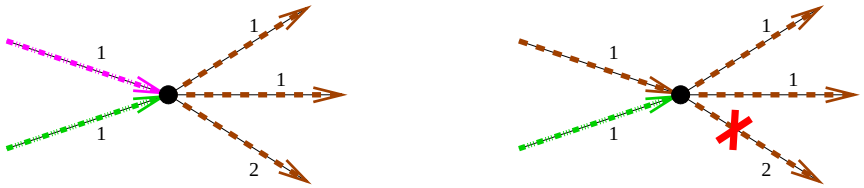


FIG. 5.9 – Deux fibres, plusieurs multicasts : deux exemples de coloration, l'un valide (à gauche), l'autre non (à droite).

Remarque 5.18 Le point crucial dans un n-fiber wavelength assignment est la fonction λ qui associe les couleurs (longueurs d'onde) aux arcs. Ce doit être un ***n*-fiber colouring**, c.à.d. une fonction $\phi : A(D) \to \Lambda$, telle qu'à tout sommet v, pour toute couleur $\lambda \in \Lambda$, $in(v,\lambda) + out(v,\lambda) \leq n$, avec $in(v,\lambda)$ le nombre d'arcs ayant reçu la couleur λ entrant v, et $out(v,\lambda)$ le nombre d'étiquettes l tel qu'il existe un arc quitant v avec la couleur λ. Etant donné un n-fiber colouring, il est facile de trouver une répartition convenable des longueurs d'onde. En effet, pour tout sommet v et toute couleur λ, il suffit de donner une fibre différente à tout arc de couleur λ entrant v et à tout ensemble d'arcs sortant de couleur λ qui ont la même étiquette. Ainsi, $\lambda_n(D)$ est le nombre minimum de couleurs tel qu'un n-fiber colouring existe.

Nous sommes plus particulièrement intéressés par $\lambda_n(m,k) = \max\{\lambda_n(D) \mid D \text{ est } m\text{-étiqueté et } \Delta^-(D) \leq k\}$ qui est le nombre maximum de longueurs d'onde nécessaires s'il y a n fibres et que chaque nœud envoie au plus m multicasts et reçoit au plus k multicasts. En particulier, $\lambda_1(1,k) = \max\{dst(D) \mid \Delta^-(D) \leq k\}$. Avec cette formulation, nos résultats impliquent que $2k \leq \lambda_1(1,k) \leq 2k+1$. Brandt et Gonzalez ont montré que pour $n \geq 2$ on a $\lambda_n(1,k) \leq \lceil \frac{k}{n-1} \rceil$.

$n \geq 2$ et $m \geq 2$. Nous montrons que si $m \geq n$ alors il existe une constante C telle que :

$$\left\lceil \frac{m}{n} \left\lceil \frac{k}{n} \right\rceil + \frac{k}{n} \right\rceil \leq \lambda_n(m,k) \leq \left\lceil \frac{m}{n} \left\lceil \frac{k}{n} \right\rceil + \frac{k}{n} \right\rceil + C\frac{m^2 \log(k)}{n}$$

Nous conjecturons que la borne inférieure est la bonne valeur de $\lambda_n(m,k)$ quand $m \geq n$. Nous montrons également que si $m < n$ alors :

$$\left\lceil \frac{m}{n} \left\lceil \frac{k}{n} \right\rceil + \frac{k}{n} \right\rceil \leq \lambda_n(m,k) \leq \left\lceil \frac{k}{n-m} \right\rceil.$$

Les bornes inférieures généralisent les résultats de Brandt et Gonzalez [BG05] qui établissaient ces inégalités dans le cas particulier $k \leq 2$, $m \leq 2$ et $k = m$. Les graphes orientés utilisés pour obtenir les bornes inférieures sont tous acycliques et nous montrons que si $m \geq n$, alors les bornes inférieures donnent en effet la bonne valeur pour les graphes orientés acycliques. De plus, les graphes orientés atteignant les bornes inférieures ont un grand degré. En généralisant les résultats de Guiduli [Gui97], nous montrons qu'un graphe orienté D m-étiqueté ayant son degré entrant et sortant borné par k peut être coloré avec peu de couleurs :

$$\lambda_n(D) \leq \frac{k}{n} + C'\frac{m^2 \log(k)}{n} \qquad \text{pour une constante } C'.$$

5.2.1.3 Apport d'un splitter

A partir de l'analyse précédente, il est possible de quantifier l'apport d'un splitter en longueurs d'onde dans un réseau en étoile. Prenons un exemple pour fixer les idées : dans le cas où chaque système est relié par une fibre au nœud central, si on note k le nombre maximum de multicasts envoyés et reçus par un nœud du réseau, alors pour assurer l'existence d'un routage sans splitter, il faut $2k+1$ longueurs d'onde par fibre. Avec un splitter en longueurs d'onde au nœud central il suffit de $k + O(\log(k))$ longueurs d'onde. Ainsi, un splitter en longueurs d'onde permet de diviser par deux le nombre de longueurs d'onde nécessaires.

Selon une remarque d'O. Amini, ce résultat peut se généraliser à un réseau quelconque qui possède à chaque nœud des splitters capable de dupliquer chacune des longueurs d'onde indépendamment. Le problème est, étant donné un routage de requêtes (c.à.d. un ensemble de chemins et d'arbres permettant de router chacune des requêtes), de savoir combien de longueurs d'onde sont nécessaires pour pouvoir le réaliser, autrement dit de savoir combien il faut de longueurs d'onde pour pouvoir en donner une à chacun des chemins et des arbres de telle sorte qu'il n'y ait pas de conflit. Dans le cas où le réseau est équipé de splitters, si le routage est tel que la profondeur de chaque arbre de routage est bornée et que le nombre maximum de feuilles de chaque arbre de routage est borné par une constante fois la charge maximale du réseau ℓ, il suffit de $\ell + O(\log(\ell))$ longueurs d'onde. Dans le cas où le réseau n'est pas équipé de splitters, il peut falloir jusqu'a $(\Delta+1)\ell$ longueurs d'onde où Δ est le degré maximum du réseau.

Cela permet de déduire que l'utilisation de splitters en longueurs d'onde permet un gain théorique d'un facteur $1/(\Delta+1)$ concernant le nombre de longueurs d'onde

nécessaires. D'autre part, équiper les nœuds de convertisseurs de longueurs d'onde permet d'économiser de l'ordre de $\log(\ell)$ longueurs d'onde.

Dans [MZQ98], les auteurs avaient estimés que l'utilisation de multicasts permet de réduire d'au moins 50 % le nombre de longueurs d'onde nécessaires. Ils montrent également que la conversion en longueur d'onde ne permet qu'un gain de l'ordre de 10 %.

Ces résultats vont dans le même sens que nos résultats. En effet le gain théorique de la conversion en longueur d'onde est faible : de seulement un facteur additif de $O(\log(\ell))$, à comparé au gain empirique de 10 %. Si l'apport théorique semble plus faible que celui mesuré de manière empirique, cela est surement dû à l'utilisation d'algorithmes gloutons dans [MZQ98].

Enfin, l'apport du multicast mis en évidence dans [MZQ98] est élevé (facteur 2) tout comme dans notre étude. Il est cependant plus faible que l'apport théorique (facteur 3,2 en moyenne pour leur instance utilisant NFSnet). Cela peut être expliqué par au moins deux raisons. La première est que les cas pour lesquels sont calculés le gain maximal théorique sont pathologiques. La seconde raison est que les auteurs de [MZQ98], ne calculent pas des routages optimaux.

5.2.2 Multicast dans les réseaux de cœur

Dans la section précédente, j'ai présenté les apports d'un splitter en longueurs d'onde par rapport à l'absence de splitters. Dans la pratique, les réseaux ne sont pas encore équippés de tels splitters car cela a été jugé moins rentable que d'utiliser des splitters simples. C'est pourquoi, dans cette section, je m'intéresse à des réseaux équippés de splitters dupliquant intégralement une fibre et je présente des résultats sur le routage de multicasts dans les réseaux WDM.

Le problème MC-RWA (Problème 5.3) consiste à trouver des chemins ou des arbres et de leur associer une longueur d'onde afin d'établir et de maintenir des connexions pour satisfaire les requêtes. Selon les hypothèses, chaque requête peut être satisfaite à l'aide de plusieurs arbres de routage. MC-RWA peut être vu comme la succession de deux problèmes : premièrement, trouver un arbre pour chacune des requêtes, et une fois les arbres trouvés, leur assigner des longueurs d'onde. Comme dans le cas du routage sur un chemin, deux arbres de routage utilisant une même fibre ne peuvent pas avoir la même longueur d'onde. MC-RWA généralise le problème de **routage et affectation de longueurs d'onde** classique (**Routing and Wavelength Assignment**, **RWA**), ce dernier problème étant maintenant bien étudié et pouvant être résolu de manière efficace [JMT06].

Noeuds MC et MIC. Dans les réseaux optiques permettant le routage de multicasts, seuls certains nœuds sont équipés de splitters (multicast-capable nodes notés MC), les autres nœuds (multicast-incapable nodes) sont notés MIC. Les nœuds auxquels sont installés les splitters (les nœuds MC) sont choisis lors de la conception du réseau. Tous les nœuds n'en sont pas équipés afin de ne pas trop augmenter le coût du réseau. Les caractéristiques des splitters que l'on considère sont données dans la

Section 5.1.1.6. Dans l'étude qui suit, je me limite au cas de continuité en longueur d'onde, c.à.d. que la longueur d'onde d'un signal n'est pas modifiée au cours de la transmission du signal. Ce choix est justifié par le fait que les technologies optiques permettant la conversion de longueur d'onde sont coûteuses et que le gain occasionné est faible, comme je l'ai montré dans la section précédente (Section 5.2.1.3).

Deux objectifs différents. Je présente plusieurs formulations mathématiques permettant de résoudre le problème MC-RWA avec plusieurs objectifs. Toutes ces formulations ont été développées avec B. Jaumard. Le premier objectif est plus spécifique aux réseaux de cœur dans lesquels toutes les destinations doivent être servies simultanément. Dans un tel contexte, l'objectif est de maximiser le nombre de requêtes satisfaites. J'appelle ce problème le problème de **Routage Complet MC-RWA** (**fully routed MC-RWA**, **FR-MC-RWA**).

Problème 5.19 (Routage Complet MC-RWA : FR-MC-RWA)

Entrées :	*Un réseau avec des coûts et des capacités sur les liens.* *Un ensemble de requêtes multicasts.*
Sortie :	*Un routage (arbre et longueur d'onde) des requêtes multicasts utilisant un seul arbre de routage par requête.*
Objectif :	*Maximiser le nombre de requêtes dont toutes les destinations sont servies.*

Dans la littérature, une solution au problème FR-MC-RWA est souvent calculée en deux étapes. La première consiste à trouver les arbres de routage et la deuxième à leur assigner des longueurs d'onde. Le problème de trouver les arbres de routage de coût minimum est le problème de Steiner qui est un problème NP-dur. Dans cette section, je propose une formulation mathématique permettant de résoudre ces deux problèmes simultanément.

Le second objectif que je considère est plus adapté aux réseaux d'accès car dans ce contexte il est plus pertinent de maximiser le nombre d'utilisateurs recevant leurs données et d'autoriser une requête à être satisfaite partiellement. J'appelle ce problème le **problème de Routage Partiel MC-RWA** (**Partially Routed MC-RWA** Problem) **PR-MC-RWA**. L'un des premiers articles à considérer cet objectif pour le problème de routage des multicasts est [HCT01], qui propose une formulation non linéaire. Dans la Section 5.2.5 je propose la première formulation linéaire sous forme de génération de colonnes.

Problème 5.20 (Routage Partiel MC-RWA : PR-MC-RWA)

Entrées :	*Un réseau avec des coûts et des capacités sur les liens.* *Un ensemble de requêtes multicasts.*
Sortie :	*Un routage (arbre et longueur d'onde) des requêtes multicasts pouvant utiliser plusieurs arbres de routage par requêtes.*
Objectif :	*Maximiser le nombre de destinations servies.*

Plan. Dans la prochaine section, je détaille les contraintes du problème MC-RWA. Dans la Section 5.2.4, je propose une première formulation pour le problème FR-MC-RWA avec pour objectif de maximiser le nombre de requêtes satisfaites. À la fin de cette section je discute de la manière dont l'atténuation peut être incorporée à ce modèle. Dans la Section 5.2.5, je propose une autre formulation pour le problème PR-MC-RWA avec pour objectif de minimiser le taux de blocage des utilisateurs, c.à.d. le nombre d'utilisateurs non servis.

5.2.3 Entrées des problèmes MC-RWA

Étant donné un réseau optique WDM représenté par un graphe orienté $\overrightarrow{D} = (V,E)$ comme décrit dans la Section 5.1.2, les nœuds de ce réseau sont partionnés en deux parties : les ensembles MC et MIC. L'ensemble MC est partitionné en plusieurs sous-ensembles : on note MC_k l'ensemble des nœuds équipés d'un splitter $1:k$; l'entrée et les sorties de chaque splitter sont également supposées connues.

L'ensemble des requêtes est défini par une matrice $T = (T_{sD})$ où T_{sD} est le nombre de connexions requises du nœud source v_s vers l'ensemble de nœuds destinations V_D. Soit $\mathcal{SD} = \{(v_s, V_D) \in V \times \mathcal{P}(V) : T_{sD} > 0\}$ l'ensemble des requêtes, où $\mathcal{P}(V)$ est l'ensemble des parties de V. On note $\mathcal{T}_{sD}$ l'ensemble des arbres de racine v_s couvrant V_D pour $(v_s, V_D) \in \mathcal{SD}$ et par $\mathcal{T}$ l'ensemble des arbres, c.à.d. $\mathcal{T} = \bigcup_{(v_s,V_D)\in\mathcal{SD}} \mathcal{T}_{sD}$.

Comme dit précédemment, on suppose que les requêtes sont routées sur une unique longueur d'onde. Autrement dit, étant donnés une requête et un chemin entre sa source v_s et un des nœuds destination $v_d \in V_D$, le chemin de v_s à v_d est routé sur une unique longueur d'onde.

Le problème MC-RWA peut être formellement décrit comme suit : étant donnés un graphe orienté $\overrightarrow{D}$ associé à un réseau optique WDM et un ensemble de requêtes, trouver un ensemble d'arbres et de longueurs d'onde (t, λ) pour chacune des requêtes satisfaites de telle manière qu'aucun arbre partageant un arc ne soit associé à une même longueur d'onde. Dans FR-MC-RWA l'objectif est de minimiser le taux de blocage et dans PR-MC-RWA, il est de maximiser le nombre d'utilisateurs servis.

5.2.4 Le problème FR-MC-RWA

Je présente dans cette section deux formulations pour le problème FR-MC-RWA qui utilisent la méthode de génération de colonne, méthode que je présente également. Il est aussi proposé une extension de ces formulations prenant en compte l'atténuation des signaux.

Dans le cas des réseaux de cœur, il est nécessaire d'utiliser des lasers permettant d'émettre les signaux à une puissance suffisante. Or le coût de ces lasers est élevé. Ainsi les opérateurs choisissent d'émettre chaque requête en une seule fois. Chaque requête satisfaite doit donc l'être par un arbre de multicast unique.

5.2.4.1 Principe de la génération de colonnes

?? Dans cette section, je présente le principe de la génération de colonnes. Il s'agit d'une technique de résolution de programme linéaire adaptée aux problèmes de grande taille. La technique de génération de colonnes repose sur la décomposition du programme linéaire initial en deux problèmes : le **problème maître** et le **problème auxiliaire**. Le problème maître correspond au problème initial dans lequel seule une partie des variables est considérée, chaque variable correspondant à une **colonne**. L'intérêt de résoudre le problème initial sur un petit ensemble de variables est que l'algorithme du simplexe est alors plus rapide.

Le problème auxiliaire est défini à partir du problème maître. Son rôle est de construire des variables -les colonnes de la matrice du problème maître- qui peut-être vont améliorer la solution du problème maître.

Ainsi, la génération de colonnes est un procédé itératif. À chaque étape, le problème maître est résolu, puis le problème auxiliaire afin de savoir si une colonne doit être rajoutée à la matrice du problème maître. Si une colonne est trouvée, le procédé se répète. Sinon il se termine et la solution trouvée par le problème maître est optimale.

La décomposition du problème initial en deux problèmes, maître et auxiliaire, repose sur la décomposition de Dantzig-Wolfe. Le livre *Column Generation* [DDS05] présente de manière très didactique la théorie sur laquelle repose cette technique ainsi que des applications à plusieurs problèmes. La manière dont est découpé le problème initial pour former le problème maître et le problème auxiliaire est importante. L'article [JMT05] illustre cela. Les auteurs y comparent plusieurs formulations du problème de routage et d'affectation de longueurs d'onde, une variante du problème que j'ai présenté précédemment dans lequel il y a plusieurs couches identiques en parallèle et tel qu'une requête doit être routée intégralement sur l'une d'entre elles.

5.2.4.2 Problème FR-MC-RWA : formulation avec configuration de routage

La formulation que je propose pour le problème FR-MC-RWA est basée sur le concept de configurations de routage. Une **configuration de routage pour FR-MC-RWA** est un ensemble d'arbres pouvant tous être routés sur une même longueur d'onde tout en satisfaisant les contraintes du problème FR-MC-RWA. Ainsi, une configuration de routage peut être utilisée pour router une partie des requêtes. Afin de diminuer la taille de l'espace des configurations de routage à explorer, on ne considère que celles qui sont maximales, c.à.d. celles auxquelles on ne peut pas ajouter d'autres arbres de routage tout en respectant les contraintes de FR-MC-RWA.

Ce modèle, écrit sous forme de génération de colonnes, mène à décomposer FR-MC-RWA en deux problèmes : le problème maître et le problème auxiliaire. Le problème maître sélectionne les configurations les plus adaptées, une par longueurs d'onde, alors que le problème auxiliaire construit les configurations les plus prometteuses. C'est le problème auxiliaire qui prend en compte les contraintes spécifiques à FR-MC-RWA pour une longueur d'onde donnée. En particulier, deux arbres de routage ne peuvent pas partager un arc dans une configuration.

Notations. Une configuration de routage est donnée par un vecteur $C = (C_{sD})_{(v_s,v_D)\in\mathcal{SD}}$ tel que C_{sD} est le nombre d'arbres de racine v_s couvrant V_D dans cette configuration. La plupart du temps, ce sera un vecteur de 0 et de 1. Chaque variable $w_C \in \mathbb{N}$ du problème maître est associée à une configuration C et sa valeur indique le nombre de longueurs d'onde sur lesquelles cette configuration est utilisée dans la solution. Une configuration peut en effet être utilisée plusieurs fois, bien que ce ne soit pas le cas en pratique.

$\mathcal{C}^{FR-MC-RWA}$ (ou simplement $\mathcal{C}$ s'il n'y a pas de confusion) est l'ensemble des configurations. Cet ensemble est énorme, cependant, en pratique, seule une petite fraction de cet ensemble est générée. Cela est dû à la rapidité de convergence des formulations sous forme de génération de colonnes (voir C. Barnhart *et al.* [BJN+98]).

Problème maître. Le problème maître de FR-MC-RWA peut être formulé comme suit :

$$\max \quad z_{FR-MC-RWA}(w) = \sum_{C\in\mathcal{C}} \sum_{(v_s,V_D)\in\mathcal{SD}} C_{sD}\, w_C$$

sujet à :

$$\sum_{C\in\mathcal{C}} w_C \leq W \qquad (u^0) \tag{5.1}$$

$$\sum_{C\in\mathcal{C}} C_{sD}\, w_C \leq T_{sD} \qquad (v_s,V_D)\in\mathcal{SD} \qquad (u_{sD}) \tag{5.2}$$

$$w_C \in \mathbb{N} \qquad C\in\mathcal{C}, \tag{5.3}$$

où u^0 et u_{sD} sont les variables duales associées aux contraintes (5.1) et (5.2-sD).

Comme le nombre de configurations est exponentiel, on résoud le problème maître restreint à un petit ensemble de variables. Ensuite, le problème auxiliaire est résolu pour savoir si la solution trouvée est optimale ou non.

Problème auxiliaire. Le problème auxiliaire génère les configurations de routage les plus prometteuses en se basant sur une métrique héritée du problème maître au travers des variables duales. Il prend également en compte les contraintes imposées par le problème FR-MC-RWA concernant les requêtes routées sur une même longueur d'onde.

Le coût d'une configuration de routage C, noté $\bar{c}(C)$, est calculé comme suit :

$$\bar{c}_{FR-MC-RWA}(C) = \sum_{(v_s,V_D)\in V_D} (1-u_{sD})C_{sD} - u^0.$$

S'il existe une configuration qui peut améliorer la solution actuelle du problème maître (c.à.d. réduire le taux de blocage), elle a un coût négatif.

Le problème auxiliaire est écrit sous la forme d'un problème de flot dans un graphe orienté auxiliaire $\overrightarrow{D}_{aux}$ défini comme suit : $\overrightarrow{D}_{aux}$ est composé d'une copie du graphe orienté $\overrightarrow{D}$ et d'une copie v' de chaque sommet v relié au sommet dont il est la copie par un arc vv'. La construction du graphe orienté auxiliaire est illustrée par la Figure 5.10

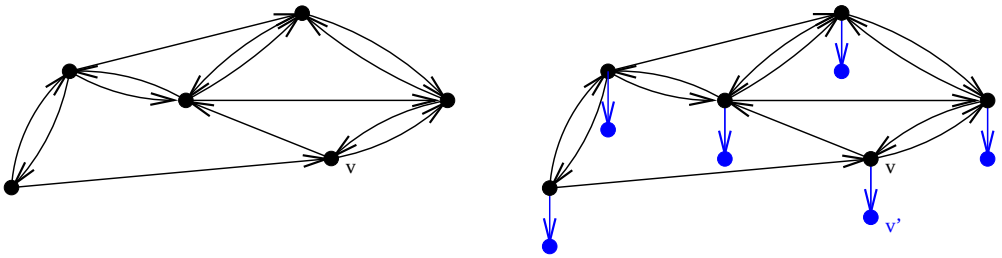


FIG. 5.10 – Le graphe orienté auxiliaire.

Le but est de router les requêtes $R_{sD} \in \mathcal{SD}$ de leur source v_s aux copies des nœuds destination V_D. Les arbres de routage peuvent avoir des nœuds de degré supérieur à deux uniquement au niveau des nœuds de MC. On appelle φ_e^{sD} la variable associée au flot de la requête R_{sD} sur l'arête e. Cette première formulation ne prend pas en compte l'atténuation du signal.

L'objectif du problème auxiliaire est :

$$\min \quad \overline{c}_{FR-MC-RWA}(\varphi) = \sum_{(v_s,V_D)\in\mathcal{SD}} (1-u_{sD})\varphi^{sD}_{v'_s v_s} - u^0.$$

Les contraintes sont :

$$\phi^{sD}_{v'_s v_s} \leq T_{sD} \qquad (v_s,V_D) \in \mathcal{SD} \tag{5.4}$$

$$\phi^{sD}_{v'_s v_s} = \phi^{sD}_{v_d v'_d} \qquad (v_s,V_D) \in \mathcal{SD}, v_d \in V_D \tag{5.5}$$

$$\sum_{(v_s,V_D)\in\mathcal{SD}} \phi^{sD}_e \leq 1 \qquad e \in E(\overrightarrow{D}) \tag{5.6}$$

$$\sum_{e\in\Gamma^+(v)} \phi^{sD}_e \leq \sum_{e\in\Gamma^-(v)} \phi^{sD}_e$$
$$(v_s,V_D) \in \mathcal{SD},\ v \in MIC \tag{5.7}$$

$$\sum_{e\in\Gamma^+(v)\setminus E^{out}_v} \phi^{sD}_e \leq \sum_{e\in\Gamma^-(v)\setminus\{e^{\text{in}}_v\}} \phi^{sD}_e$$
$$(v_s,V_D) \in \mathcal{SD},\ v \in MC_k \tag{5.8}$$

$$\phi^{sD}_e \in \{0,1\} \quad (v_s,V_D) \in \mathcal{SD},\ e \in E(\overrightarrow{D}) \tag{5.9}$$

$$\phi^{sD}_{e^{in}_v} \geq \phi^{sD}_e \quad (v_s,V_D) \in \mathcal{SD},\ v \in MC_k,\ e \in E^{\text{out}}_v \tag{5.10}$$

$$\phi^{sD}_e = 0 \qquad (v_s,V_D) \in \mathcal{SD},\ e = (v'_d, v_d) \tag{5.11}$$

Un arbre de racine v_s couvrant V_D est construit avec les arcs e tels que $\phi^{sD}_e = 1$. Les contraintes (5.7) traduisent la contrainte de flot aux nœuds qui ne sont pas équipés de splitters. Il est intéressant de remarquer qu'un flot peut se terminer à un nœud intermédiaire. Les contraintes (5.8) traduisent la contrainte de flot aux nœuds qui sont équipés de splitters, entre les arcs entrant et sortant des splitters. Les contraintes (5.10)

traduisent le fait qu'un signal arrivant sur la fibre d'entrée d'un splitter est transmise à ses fibres de sortie. De cette manière, un ensemble d'arbres arcs-disjoints tel que les nœuds de degré $k+1$ sont situés aux nœuds de $\cup_{\ell \geq k} MC_\ell$ est défini. Cela correspond bien à une configuration.

Atténuation du signal. La formulation suivante permet de prendre en compte l'atténuation du signal optique. À chaque arc est associé un coût a_e correspondant à l'atténuation d'un signal utilisant cet arc. L'atténuation est exprimée en décibels. Il est également possible d'inclure à ce coût l'atténuation induite par un splitter puisque leur position est fixe. Enfin, si des amplificateurs sont utilisés afin de régénérer le signal, leur effet peut aussi être incorporé à ce coût. L'atténuation du signal est une contrainte importante dans le routage car un signal trop dégradé n'est pas exploitable par les destinations. La limite habituellement utilisée est une atténuation de 20 dB. Un nouvel ensemble de variables similaires aux variables ϕ_e^{sD} est introduit pour prendre en compte l'atténuation : pour tout $e \in E$, tout $(v_s, V_D) \in \mathcal{SD}$ et tout $d \in V_D$, la variable $\varphi_e^{sD,d}$ est introduite. Elle compte l'atténuation du signal allant de s à $d \in V_D$.

$$\sum_{d \in D} \varphi_e^{sD,d} \geq \varphi_e^{sD} \qquad (v_s, V_D) \in \mathcal{SD}, e \in E \tag{5.12}$$

$$\sum_e \varphi_e^{sD,d} a_e \leq 20 \qquad (v_s, V_D) \in \mathcal{SD}, v_d \in V_D \tag{5.13}$$

$$\varphi_e^{sD,d} \leq \varphi_e^{sD} \qquad (v_s, V_D) \in \mathcal{SD}, e \in E(\overrightarrow{D}) \tag{5.14}$$

$$\varphi_{v'_s v_s}^{sD,d} = \varphi_{v_d v'_d}^{sD,d} \qquad (v_s, V_D) \in \mathcal{SD}, v_d \in V_D \tag{5.15}$$

$$\varphi_e^{sD,d} \in \{0,1\} \qquad (v_s, V_D) \in \mathcal{SD},\ e \in E(\overrightarrow{D}) \tag{5.16}$$

$$\sum_{e \in \Gamma^+(v)} \varphi_e^{sD,d} = \sum_{e \in \Gamma^-(v)} \varphi_e^{sD,d} \qquad (v_s, V_D) \in \mathcal{SD},\ v_d \in V_D, e \in E(\overrightarrow{D}) \tag{5.17}$$

Ces équations sont adaptables aux autres formulations que je propose. Les variables $\varphi_e^{sD,d}$ définissent un chemin de v_s à v_d. Les équations 5.15 assurent qu'elles utilisent un chemin réservé pour la requête. L'équation 5.14 s'assure que sur ce chemin, l'atténuation globale ne dépasse pas 20 dB ; si c'est le cas les variables $\varphi_e^{sD,d}$ sont forcées d'être à 0, ce qui, par les équations 5.12, empêche la requête d'être satisfaite.

5.2.4.3 Problème FR-MC-RWA : formulation par arbres de routage

Problème maître. Dans cette formulation, le problème maître est résolu sur un sous-ensemble de $\mathcal{T}$ (c.f. Section 5.2.3) et sélectionne un ensemble d'arbres de routage et de longueurs d'onde afin de satisfaire un maximum de requêtes. La contrainte que deux arbres de routage ayant la même longueur d'onde ne partagent pas d'arc est vérifiée dans le problème maître. Le problème auxiliaire inclut les autres contraintes de FR-MC-RWA qui indiquent que les nœuds de grand degré des arbres de routage doivent correspondre avec les nœuds de MC.

Un arbre de routage t est représenté par un ensemble de variables $\delta_{e,t}$ qui valent 1 si l'arc e est utilisé par l'arbre de routage et 0 sinon. Chaque variable w_t^λ du problème maître est associée à un arbre de routage t et à une longueur d'onde λ. w_t^λ vaut 1 si l'arbre de routage t est utilisé sur la longueur d'onde λ, 0 sinon.

Le problème maître est :

$$\max \quad z_{max\ MC-RWA-LT}(w) = \sum_{\lambda\in\Lambda}\sum_{t\in\mathcal{T}} w_t^\lambda$$

sujet à :

$$\sum_{t\in\mathcal{T}} \delta_{e,t} w_t^\lambda \leq 1 \qquad \lambda\in\Lambda,\ e\in E \qquad (u_{\lambda e}^0) \tag{5.18}$$

$$\sum_{\lambda\in\Lambda}\sum_{t\in\mathcal{T}_{sD}} w_t^\lambda \leq T_{sD} \quad (v_s,V_D)\in\mathcal{SD} \qquad (u_{sD}^1) \tag{5.19}$$

$$w_t^\lambda \in \mathbb{N} \qquad t\in\mathcal{T},\ \lambda\in\Lambda. \tag{5.20}$$

$(u_{\lambda e}^0)$ et (u_{sD}^1) sont les variables duales associées aux contraintes (5.18-e) et (5.19-(v_s,V_D)).

Problème auxiliaire. Il y a un problème auxiliaire différent pour chaque longueur d'onde. Étant donnée une longueur d'onde λ, on considère le graphe orienté associé au réseau. À chaque arc e on associe comme coût la variable duale $(u_{\lambda e}^0)$. Le problème auxiliaire est un problème de flot sur ce graphe orienté. Deux familles de variables sont utilisées : φ_e^λ est la variable de flot associée à l'arc e et φ_{sD}^λ indique le nombre d'arbres de routage de racine v_s couvrant V_D.

Étant donnée une longueur d'onde λ, l'objectif du problème auxiliaire est :

$$\max \qquad 1 - \sum_{e\in t} u_{\lambda e}^0 \varphi_e^\lambda - \sum_{sD} u_{sD}^1 \varphi_{sD}^\lambda$$

sous les contraintes :

$$\sum_{sD} \varphi_{sD}^\lambda = 1 \quad e\in E \tag{5.21}$$

$$\sum_{v_s:(v_s,V_D)\in\mathcal{SD},v_i\in V_D} \varphi_{sD}^\lambda + \sum_{e\in\Gamma^+(v_i)} \varphi_e^\lambda = \sum_{e\in\Gamma^-(v_i)} \varphi_e^\lambda + \sum_{V_D:(v_i,V_D)\in\mathcal{SD}} \varphi_{sD}^\lambda, \quad v_i \in MIC \tag{5.22}$$

$$\sum_{v_s:(v_s,V_D)\in\mathcal{SD},v_i\in V_D} \varphi_{sD}^\lambda + \sum_{e\in\Gamma^+(v_i)\setminus E_{v_i}^{out}} \varphi_e^\lambda = \sum_{e\in\Gamma^-(v_i)\setminus e_{v_i}^{in}} \varphi_e^\lambda + \sum_{V_D:(v_i,V_D)\in\mathcal{SD}} \varphi_{sD}^\lambda, \quad v_i\in MC_i \tag{5.23}$$

$$\phi_{e_v^{in}}^{sD} \geq \phi_e^{sD}, \qquad (v_s,V_D)\in\mathcal{SD},\ v\in MC_k,\ e\in E_v^{out} \tag{5.24}$$

$$\varphi_{sD}^\lambda \in \{0,1\} \qquad (v_s,V_D)\in\mathcal{SD} \tag{5.25}$$

$$\varphi_e^\lambda \in \{0,1\} \qquad e\in D \tag{5.26}$$

Le problème auxiliaire consiste à trouver un arbre de coût minimum, il s'agit donc de trouver un arbre de Steiner pour chaque requête, or ce problème est NP-dur. Cependant si les multicasts sont suffisamment petits, on peut utiliser un algorithme exact (de complexité en $2^k n^3$ où k est le nombre de destinataires du multicast), mais il ne faut pas oublier les contraintes imposées par les splitters.

5.2.5 Problème PR-MC-RWA

Dans les réseaux d'accès les signaux sont émis avec moins de puissance que dans le cas des réseaux de cœur, le coût des lasers utilisés est alors beaucoup plus faible ce qui permet d'en utiliser plusieurs par multicast. Ainsi, dans un réseau d'accès, un multicast peut être émis en plusieurs fois. Cela signifie qu'une requête peut être satisfaite par une union d'arbres. Ce problème introduit dans [HCT01], a encore été peu étudié.

5.2.5.1 Problème PR-MC-RWA : formulation avec configuration de routage

Je présente une formulation pour le problème PR-MC-RWA basée sur le concept de configuration de routage. Le principe est similaire à celui de la Section 5.2.4.2. Une **configuration de routage pour PR-MC-RWA** est associée à un ensemble d'arbres qui peuvent tous être routés sur une même longueur d'onde. Chaque arbre sert une requête, il peut éventuellement la servir partiellement. Je rappelle que dans PR-MC-RWA, une source peut envoyer plusieurs fois le même message et une requête ne doit pas forcément être satisfaite complètement. Soit $\mathcal{C}^{PR-MC-RWA}$ l'ensemble de toutes les configurations de routage pour PR-MC-RWA. Une configuration de routage est représentée par un vecteur C. Chaque variable w_C du problème maître est associée à l'une d'entre elles. La valeur de w_C indique le nombre de fois où la configuration C est utilisée dans la solution. Étant donné un vecteur C associé à une configuration de routage, à chaque requête (v_s, V_D) et à chaque sous-ensemble de nœuds de destination $V'_D \subset V_D$ est associé une ligne de $C : C(sD')$, elle indique le nombre d'arbres de routage de racine v_s et couvrant V'_D dans cette configuration.

Cette formulation permet de partager une requête en plusieurs requêtes pour la router. En particulier, on peut router une requête sur plusieurs longueurs d'onde.

Problème maître. L'objectif du problème maître est :

$$\max \quad z_{PR-MC-RWA-\mathrm{IRC}}(w) = \sum_{C \in \mathcal{C}} \sum_{(v_s,V_D) \in \mathcal{SD}, V'_D \subset V_D} |V'_D| z^{sD}_{sD'} \tag{5.27}$$

sujet à :

$$\sum_{C \in \mathcal{C}} w_C \leq W \qquad (u^0) \tag{5.28}$$

$$w_C \in \mathbb{N} \qquad C \in \mathcal{C}(u^1_{sD}). \tag{5.29}$$

$$\sum_{C\in\mathcal{C}} C(sD')w_C \quad \geq \sum_{V_D \text{ tq } V_{D'}\subset V_D} z_{sD'}^{sD} \qquad (v_s,V_{D'}) \text{ tq } \exists (v_s,V_D)\in\mathcal{SD},\ V_{D'}\subset V_D \qquad (5.30)$$

$$\sum_{V'_D\subset V_D \ st\ d\in V'_D} z_{sD'}^{sD} \leq T_{sD} \qquad (v_s,V_D)\in\mathcal{SD},\ d\in V_D \qquad (5.31)$$

où $z_{sD'}^{sD}$ est le nombre d'arbres de s à V'_D utilisés pour router les requêtes de s à V_D. Seuls les ensembles V'_D tels que $V'_D \subset V_D$ sont considérés car d'une solution vérifiant $V'_D \setminus V_D \neq \emptyset$, on peut construire une solution ayant le même coût avec $V'_D \subset V_D$. Soit u^0 et u^1_{sD} les variables duales des contraintes (5.27) et (5.30-(v_s,V'_D)) respectivement.

Problème auxiliaire. Le coût réduit d'une colonne C est noté $\bar{c}_{PR-MC-RWA-\text{IRC}}(C)$. Il vaut :

$$\bar{c}_{PR-MC-RWA-\text{IRC}}(C) = \sum_{\substack{(v_s,V_{D'}):\\ \exists (v_s,V_D)\in\mathcal{SD}\\ \text{et } V_{D'}\subset V_D}} C(sD')u^1_{sD'} - u^0.$$

Afin de savoir s'il existe une configuration avec un coût réduit négatif, on considère un problème auxiliaire formulé sous la forme d'un problème de flot dans le même graphe orienté auxiliaire $\overrightarrow{D}_{aux}$ que dans la Section 5.2.4.2. Pour chaque requête (v_s,V_D) et pour chaque sous-ensemble de nœuds de destination $V'_D \subset V_D$, est associée une variable de flot $\varphi_e^{sD'}$ à chaque arc e.

L'objectif du problème auxiliaire est :

$$\min\ \bar{c}_{PR-MC-RWA-\text{IRC}}(\varphi) = \sum_{\substack{(v_s,V_{D'}):\\ \exists (v_s,V_D)\in\mathcal{SD}\\ \text{et } V_{D'}\subset V_D}} \varphi^{sD'}_{v'_s v_s} u^1_{sD'} - u^0.$$

sujet à :

$$\phi^{sD'}_{v'_s v_s} \leq T_{sD} \qquad (v_s,V_D)\in\mathcal{SD}, V'_D\subset V_D \qquad (5.32)$$

$$\phi^{sD'}_{v_d v'_d} \leq T_{sD} \qquad (v_s,V_D)\in\mathcal{SD},\ v_d\in V'_D, V'_D\subset V_D \qquad (5.33)$$

$$\sum_{(v_s,V_D)\in\mathcal{SD}, V'_D\subset V_D} \phi_e^{sD'} \leq 1 \qquad e\in E(\overrightarrow{D}) \qquad (5.34)$$

$$\sum_{e\in\Gamma^+(v)} \phi_e^{sD'} \leq \sum_{e\in\Gamma^-(v)} \phi_e^{sD'} \qquad (v_s,V_D)\in\mathcal{SD},\ V'_D\subset V_D,\ v\in MIC \qquad (5.35)$$

$$\sum_{e\in\Gamma^+(v)\setminus E_v^{out}} \phi_e^{sD'} \leq \sum_{e\in\Gamma^-(v)\setminus\{e_v^{in}\}} \phi_e^{sD'} \qquad (v_s,V'_D)\in\mathcal{SD},\ V'_D\subset V_D,\ v\in MC_k \qquad (5.36)$$

$$\phi^{sD'}_{e^{in}_v} \geq \phi^{sD'}_e \qquad (v_s, V_D) \in \mathcal{SD}, V'_D \subset V_D, v \in MC_k,\ e \in E^{\text{out}}_v \tag{5.37}$$

$$\phi^{sD'}_e = 0 \qquad (v_s, V_D) \in \mathcal{SD}, V'_D \subset V_D, e = (v'_d, v_d) \tag{5.38}$$

$$\phi^{sD}_{v'_s v_s} = \phi^{sD}_{v_d v'_d} \qquad (v_s, V_D) \in \mathcal{SD},\ V'_D \subset V_D \tag{5.39}$$

$$\phi^{sD'}_e \in \{0,1\} \qquad (v_s, V_D) \in \mathcal{SD},\ V'_D \subset V_D\ e \in E \tag{5.40}$$

Les arcs e avec $\phi^{sD'}_e = 1$ forment un arbre de routage de racine v_s couvrant V'_D. Les contraintes (5.35) traduisent la contrainte de flot aux nœuds qui ne sont pas équipés de splitters. Les contraintes (5.36) traduisent la contrainte de flot aux nœuds qui sont équipés de splitters. Les contraintes (5.37) traduisent le fait qu'un signal arrivant sur la fibre d'entrée d'un splitter est transmise à ses fibres de sortie. De cette manière, on définit un ensemble d'arbres arcs-disjoints tel que les nœuds des arbres de degré $k+1$ sont situés aux nœuds du réseau appartenant à $\cup_{\ell \geq k} MC\_\ell$. Cela correspond bien à une configuration.

Remarque 5.21 J'ai décrit deux formulations mathématiques pour résoudre le problème MC-RWA. La dernière formulation a pour but de résoudre ce problème dans les réseaux d'accès. Cependant cette formulation ne prend pas en compte la faible densité des réseaux d'accès, or cette particularité peut aider dans le choix des arbres de routage. Dans la section suivante, notre but est de proposer une autre manière de résoudre le problème MC-RWA en prenant en compte la structure des réseaux d'accès.

5.2.6 Réseaux d'accès

Dans la continuité du travail précédent, je regarde le problème du routage de multicasts dans les réseaux d'accès. Le but de ces réseaux est de relier individuellement les clients aux réseaux de cœur. Le besoin de bande passante des réseaux d'accès a rapidement augmenté ces dernières années.

De nos jours, la majorité des réseaux déployés sont de type DSL (digital suscriber, ligne d'abonné numérique), ou CATV (Cable Television, télévision par câble). Ces deux technologies ont été développées pour transporter la voix et les signaux télévisuels respectivement et ne sont pas adaptés aux besoins actuels.

5.2.6.1 Réseaux Passifs Optiques

Les **réseaux passifs optiques** (**PON**, **passive optical networks**) ont pour but d'offrir une plus grande bande passante dans les réseaux d'accès. Leur rôle est de relier un **terminal optique** (**OLT**, **optical line terminal**) aux clients (**ONU**, **optical network units**) et pour cela, des splitters sont utilisés. Le réseau entre les ONUs et l'OLT est passif : il ne nécessite pas d'énergie. Ces dernières années, diverses versions des PONs sont apparues. L'ATM Passive Optical Network (APON) est le premier standard pour les réseaux passifs optiques suivi par l'Ethernet PON (EPON) qui est un standard IEEE/EFM permettant d'utiliser le protocole Ethernet. Le Broadband

PON (BPON), un standard basé sur APON et le Giga PON (GPON) une évolution de BPON, permettent un débit plus élevé, une meilleure sécurité et un choix du protocole de la couche 2 (ATM, GEM ou Ethernet).

Les réseaux PON permettent un plus grand débit que les traditionnels réseaux en cuivre. Il est pourtant nécessaire d'augmenter encore leur bande passante en utilisant le multiplexage en longueur d'onde (WDM). Un tel réseau est alors appelé un réseau WDM-PON. Le développement des réseaux WDM-PON reste encore limité malgré leurs avantages car les technologies utilisées sont encore immatures et coûteuses. Une des clefs du succès est donc de pouvoir les exploiter au mieux. Ainsi, avec B. Jaumard, A. Houle, A. Shaikh et D. Coudert, nous avons étudié le problème de routage de connexions au sein de ces réseaux.

5.2.6.2 Structure des réseaux PON

Réseaux PON en arbre. La plupart des réseaux PON ont une structure d'arbre dont la racine est l'OLT et les feuilles sont les ONUs. Ainsi, il existe un unique chemin reliant l'OLT à chaque ONU. Les fibres optiques utilisées sont bidirectionnelles et chaque OLT a à sa disposition deux longueurs d'onde, une pour les signaux OLT-ONU et une pour les signaux ONU-OLT. Chaque nœud du réseau qui n'est ni un ONU ni l'OLT est équipé d'un splitter dupliquant la fibre d'entrée sur toutes ses fibres de sorties. Il est supposé que le réseau est dimensionné en tenant compte de l'atténuation, ainsi les signaux atteignant les ONUs peuvent être interprétés correctement.

Comme le chemin de l'OLT à chaque ONU est unique et que tous les nœuds internes sont équipés de splitters, il est possible modéliser un réseau PON par une étoile. Toutes les arêtes incidentes à l'OLT sont conservées et leurs extrémités représentent l'ensemble des ONUs qui peuvent être atteintes par cette arête. La Figure 5.11 représente un réseau PON et sa simplification.

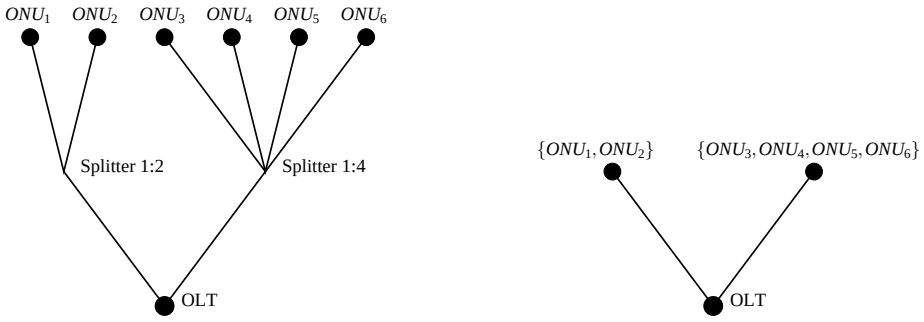
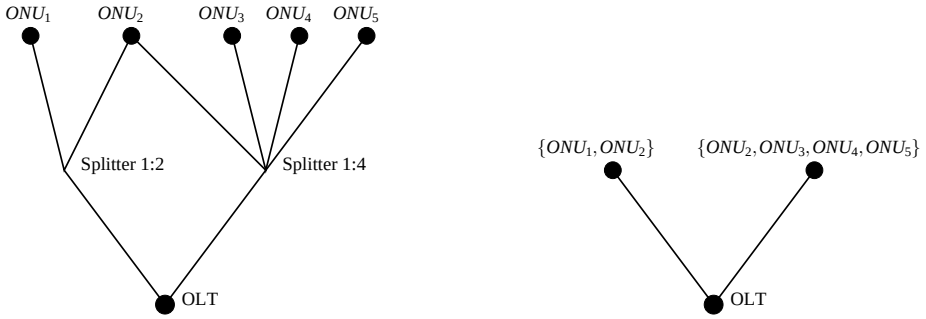


FIG. 5.11 – Un réseau PON.

Réseaux PON faiblement maillés. Dans certains cas, l'utilisateur requiert une garantie du service fourni. Un tel client est alors connecté par deux chemins disjoints à l'OLT. On parle de réseau PON faiblement maillé. Sa structure est celle d'un arbre dont

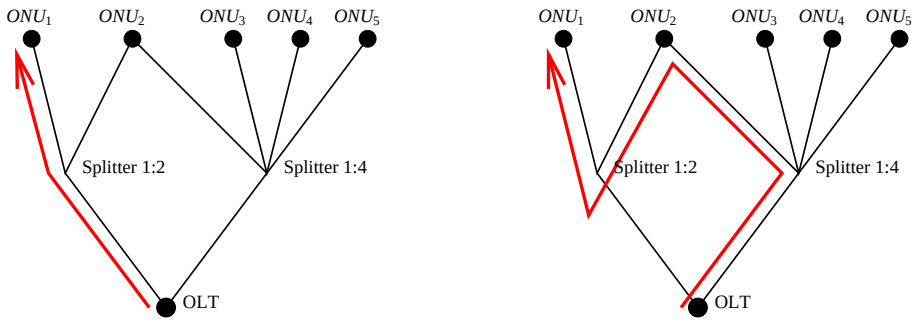
certaines feuilles sont fusionnées. L'ONU placé à un tel nœud peut alors recevoir ses messages par deux chemins différents. Cependant, un message destiné à un autre ONU ne peut pas être routé par ce nœud car l'équipement installé ne le permet pas. Comme pour les réseaux PON en forme d'arbre, les réseaux PON faiblement maillés peuvent être représentés par une étoile. La différence est qu'un ONU peut être présent dans les ensembles de plusieurs feuilles. La Figure 5.12 illustre un réseau PON faiblement maillé dans lequel l'ONU_2 est relié à l'OLT par deux chemins disjoints, et sa modélisation. La Figure 5.13, représente un routage autorisé et un routage interdit dans un réseau PON faiblement maillé.



(a) Un réseau PON faiblement maillé.

(b) Modélisation d'un réseau PON faiblement maillé.

FIG. 5.12 – Un réseau PON faiblement maillé et sa modélisation.



(a) Un routage valide.

(b) Un routage non valide.

FIG. 5.13 – La validité des routages dans un réseau PON faiblement maillé.

5.2.6.3 Modèles

Dans l'Annexe C.2 sont décrits quatre modèles différents pour le problème MC-RWA dans les réseaux PONs.

Le premier objectif considéré est de maximiser le nombre de requêtes servies dans les réseaux PON ayant une structure d'arbre puis dans les réseaux PON faiblement maillés. On modélise à la fois le cas où le multiplexage temporel n'est pas utilisé et le cas où il est utilisé.

Ensuite on étudie les mêmes problèmes avec pour objectif de maximiser le nombre d'ONUs (clients) servis. Comme dit dans la section précédente, cet objectif est le plus pertinent dans les réseaux d'accès. L'implémentation de ces modèles est en cours, ce qui va permettre de comparer les différents modèles entre eux, et en particulier d'estimer le gain obtenu en maximisant le nombre d'ONUs servis au lieu du nombre de requêtes satisfaites.

5.3 Conclusion et Perspectives

Dans ce chapitre, j'ai commencé par présenter l'intérêt de l'usage du multicast dans les réseaux. Dans la première section, j'ai décrit les technologies optiques utilisées aujourd'hui et qui concernent les problème que j'étudie. Dans la Section 5.2.1, j'ai décrit un problème de coloration de graphes orientés relié au routage de multicasts et qui permet d'évaluer le gain apporté par des splitters en longueurs d'onde. Ensuite j'ai présenté des formulations mathématiques permettant de résoudre le problème MC-RWA dans les réseaux de cœurs et les réseaux d'accès. Dans la dernière section, j'ai présenté les travaux en cours sur des formulations du problème MC-RWA dans les réseaux PON.

Il reste à finir l'implémentation des formulations mathématiques afin de pouvoir comparer les solutions que l'on propose à des solutions déjà existantes. Il est également intéressant de considérer le problème de fiabilité des connexions établies. Dans le prochain chapitre, on s'intéresse à ce problème dans le cas des requêtes unicasts.

Chapitre 6

Fiabilité et tolérance aux pannes

Dans ce chapitre j'étudie des problèmes reliés à la fiabilité des communications dans un réseau. Je commence par présenter le problème des pannes multiples dues aux groupes de risques et une formulation mathématique permettant de prendre en compte ces risques lors du routage. Dans une seconde section, je présente des modèles pour un mode de protection des connexions par segments.

Lors de l'exploitation d'un réseau, la minimisation des capacités utilisées n'est pas le seul critère à prendre en compte. En effet, il est nécessaire de s'assurer que les clients sont satisfaits du service rendu. Par conséquent, il faut garantir une continuité du service et ce, même si des composants du réseau tombent en panne. Ce chapitre est dédié à cette problématique. Il est composé de deux sections, dans la première je présente les problèmes induits par un réseau multicouche et je propose un moyen pour en tenir compte. Ces travaux font suite à la thèse de M-E. Voge, [Vog06b] et à [CPRV06, CDP+07]. Dans la deuxième section, je présente une formulation mathématique permettant de calculer des protections pour un ensemble de chemins préétablis.

6.1 Réseaux colorés

Les pannes multiples sont intrinsèquement liées aux réseaux multicouches. Les réseaux colorés ont été introduits pour modéliser de telles pannes. Dans cette section, je présente une formulation mathématique du problème de routage permettant de prendre en compte les pannes multiples. Ces travaux ont donné lieu à un article qui a été présenté lors la conférence ICC 2008 et qui peut être trouvé dans l'Annexe D.1.

6.1.1 Origine du problème

Une technique utilisée pour assurer qu'un routage soit tolérant aux pannes des liens est de trouver deux chemins disjoints pour chaque requête et de réserver assez de bande passante sur chacun de ces chemins. Il existe d'autres schémas plus efficaces, tels la protection par "p-cycles" ou la protection par chemins partagés. Plusieurs formulations mathématiques permettant de calculer des protections sont regroupées dans les chapitres 4 et 7 du livre de A.K. Somani [Som06] ainsi que dans les chapitres 9 et 10 du livre de M. Pióro et D. Medhi [PM04]. Dans le chapitre 8 de [Som06], il est question du cas où il y a deux pannes simultanées. Dans ce cas, le chemin utilisé par la connexion et le chemin de protection peuvent tous les deux tomber en panne. Garantir la protection pour toutes les paires de pannes nécessite beaucoup de bande passante et cela va en augmentant quand on augmente le nombre de pannes à protéger simultanément. Or, en pratique, les pannes simultanées sont corrélées. Il est donc intéressant de se limiter à protéger les pannes multiples les plus probables. Dans cette section, j'étudie le problème de routage dans un réseau dont les pannes multiples les plus probables ont été identifiées et représentées par des groupes de risques.

Réseaux multicouche. Les pannes multiples se produisent typiquement dans les réseaux multicouches. En effet une couche virtuelle est plongée dans un réseau physique. Chaque nœud du réseau virtuel correspond à un nœud du réseau physique et les liens du réseau virtuel sont routés sur des liens du réseau physique comme illustré par la Figure 6.1.

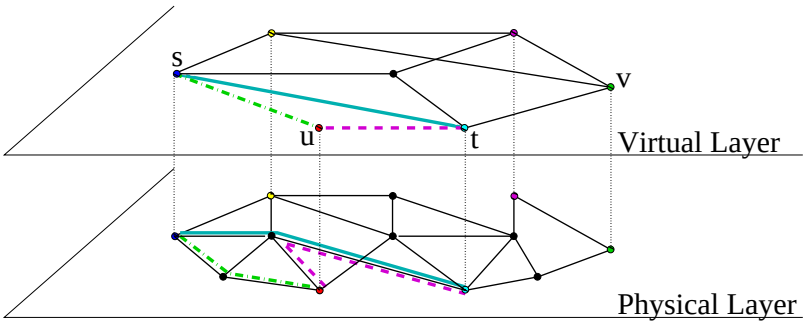


FIG. 6.1 – Groupes de risques dans un réseau multicouches.

Dans cet exemple, les liens virtuels bleu (en trait continu) et violet (en pointillés) utilisent le même lien physique. Ainsi, si ce dernier tombe en panne, les deux liens virtuels tombent en panne simultanément. Il est question de **panne multiple** et les deux liens sont dits appartenant au même **groupe de risques** (Shared Risk Group SRG).

Réseaux WiFis. Les groupes de risques sont présents dans d'autres domaines. Par exemple, dans un réseau WiFi, si un nœud défaillant émet en continu ou de manière

aléatoire, à cause des interférences, ses autres voisins sont incapables de recevoir un message, comme indiqué sur la Figure 6.2.

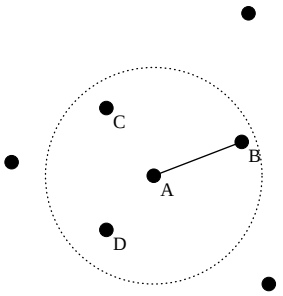


FIG. 6.2 – Groupes de risques dans un réseau radio.

Réseaux routiers. Les groupes de risques sont également présents dans les réseaux routiers : si deux routes A et B se croisent et que la route A est bloquée, dès qu'une voiture de la route B veut tourner sur la route A, la route B se retrouve bloquée, c.f. Figure 6.3.

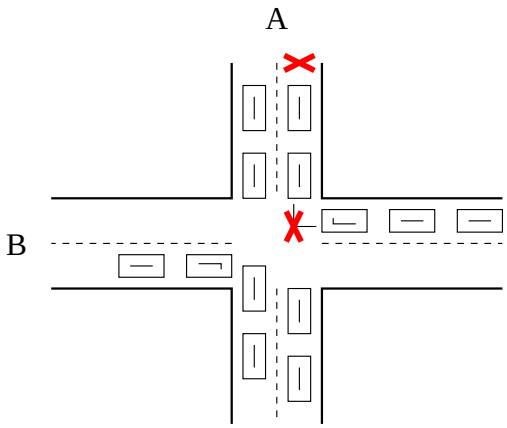


FIG. 6.3 – Groupes de risques dans un réseau routier.

Plan. Dans cette section, mon objectif est de proposer une formulation mathématique permettant de trouver un routage des requêtes qui prend en compte les groupes de risques. Une connexion qui utilise des liens qui font partie de peu de groupes de risques différents a en effet moins de chance d'être interrompue. Par la suite, il sera également plus aisé de rajouter des chemins de protection n'utilisant pas les même groupes de risques.

6.1.2 Notations

Tout au long de cette section, je considère un réseau composé de deux couches : une couche physique et une couche virtuelle. Cela permet d'illustrer l'origine des groupes de risques, mais les résultats présentés sont valables pour tout réseau dont les groupes de risques sont connus.

La couche physique est représentée par un graphe orienté $D_{\text{PN}} = (V_{\text{PN}}, E_{\text{PN}})$, où V_{PN} est l'ensemble des sommets et $E_{\text{PN}} = \{e_1, \dots e_m\}$ l'ensemble des liens (c.à.d. les fibres unidirectionnelles). De même, le réseau virtuel est représenté par un graphe orienté $D_{\text{VN}} = (V_{\text{VN}}, E_{\text{VN}})$. Chaque arc $e \in E_{\text{VN}}$ a un coût p_e par unité de flot et une capacité u_e.

À chaque arc e_i du réseau physique D_{PN} correspond un groupe de risques dans le réseau virtuel c_i, désigné par une couleur. Plus précisément, une couleur c_i correspond à un sous-ensemble d'arcs (du réseau virtuel) routés sur l'arc e_i (dans le réseau physique). Autrement dit, c'est l'ensemble des liens virtuels coupés en cas de panne de l'arc e_i. $\mathcal{C} = \{c_1, \dots, c_m\}$ est l'ensemble des couleurs (c.à.d. des SRGs) du réseau. Un même arc peut appartenir à plusieurs groupes de risques différents. Pour pallier à cela chaque arc est remplacé par une chaîne d'arcs, chacun appartenant à un unique groupe de risques, comme proposé dans [CDP+07, Vog06a]. Par simplicité, le graphe orienté modifié est toujours appelé D_{VN}. Un graphe orienté dans lequel tout arc a exactement une couleur est appelé **graphe orienté coloré**.

T est la matrice des requêtes à router dans D_{VN}. A chaque requête $t_{sd} \in T$ sont associées sa source v_s, sa destination v_d et sa taille T_{sd}. Je note $|T| = \sum_{t_{sd} \in T} T_{sd}$. La taille de chaque requête est supposée entière, et une requête peut être partagée en requêtes unitaires qui peuvent être routées indépendamment. Pour chaque requête $t_{sd} \in T$, $\mathcal{P}^{sd}$ est l'ensemble de tous les chemins de v_s à v_d dans D_{VN} et $\mathcal{P} = \cup_{t_{sd} \in T} \mathcal{P}^{sd}$.

Je rappelle que $\Gamma^-(v)$ et $\Gamma^+(v)$ représentent l'ensemble des arcs entrant et sortant de v.

Enfin, étant donné un graphe (respectivement graphe orienté) coloré, le **span** d'une couleur est le nombre de composantes connexes induites par cette couleur. Si dans un graphe (respectivement graphe orienté) coloré toutes les couleurs ont "span 1", cela signifie que les groupes de risques sont locaux et dans ce cas, certains problèmes comme le MCP (problème de trouver un chemin avec le moins de couleurs possible pour une requête) devient polynomial.

6.1.3 Problème et complexité

Problème 6.1 (Flot minimum en nombre de couleurs moyen (Minimun Average Color Flow) MACF)

Entrées :	*Un graphe orienté coloré représentant un réseau (virtuel).*
	Des coûts et des capacités sur les liens du réseau.
	Un ensemble de requêtes entières.
Sortie :	*Un routage (chemin et longueur d'onde) des requêtes.*
Objectif :	*Minimiser* $\alpha \sum$ COÛT DES ARCS $+ (1-\alpha) \sum$ #SRG MOYEN.

Il se trouve que ce problème est dur à résoudre et ce même dans des cas simples.

Complexité, coût des liens nul : $\forall e \in E, p_e = 0$**.** Avec un coût nul sur les liens, le problème de router une requête est connu comme le problème Minimum Color Path (MCP) qui a été montré NP-dur dans [YVJ05]. Le théorème suivant donne également un résultat d'inaproximabilité. Il a été prouvé dans le cas des graphes non orientés mais sa généralisation aux graphes orientés est immédiate.

Théorème 6.2 ([CDP+07]) *Le problème Minimum Color Path est NP-dur et dur à approximer à un facteur* $2^{\log^{1-\delta}|\mathcal{C}|^{\frac{1}{2}}}$ *près, avec* $\delta = (\log\log|\mathcal{C}|^{\frac{1}{2}})^{-\varepsilon}$ *et* $\varepsilon < \frac{1}{2}$.

Complexité, toutes les couleurs ont span 1. Nous avons montré dans [CHPV08] le théorème suivant qui s'applique aux réseaux dont les arcs n'ont pas un coût nul :

Théorème 6.3 *Étant donné un graphe orienté symétrique coloré où toutes les couleurs ont span 1, le problème MCP avec coût sur les arcs (MCP with edge cost, noté MCPwEC) est NP-dur et dur à approximer à un facteur* $2^{\log^{1-\delta}|\mathcal{C}|^{\frac{1}{2}}}$ *près, avec* $\delta = (\log\log|\mathcal{C}|^{\frac{1}{2}})^{-\varepsilon}$ *et* $\varepsilon < \frac{1}{2}$.

6.1.4 Formulations

6.1.4.1 Formulation sommet-arc

Pour résoudre le problème MACF, dans [CHPV08], avec D. Coudert, F. Peix et M-E. Voge, nous avons proposé deux formulations. Une première formulation sommet-arc et une seconde formulation arc-chemin sous forme de génération de colonnes.

Pour la formulation sommet-arc, nous utilisons une variable de flot binaire ϕ_e^{sd} pour tout lien $e \in E_{\text{VN}}$ et toute requête $t_{sd} \in T$. ϕ_e^{sd} vaut 1 si le lien virtuel e est utilisé pour router la requête unitaire t_{sd}, 0 sinon.

Des variables binaires sont utilisées pour les couleurs : χ_{sd}^c, pour tout $t_{sd} \in T$ et $c \in \mathcal{C}$. χ_{sd}^c vaut 1 si la requête unitaire t_{sd} est routée sur un arc appartenant au groupe de risque représenté par la couleur c.

$$\min \frac{1}{|T|} \sum_{t_{sd}} \left(\alpha \sum_{c \in \mathcal{C}} \chi_{sd}^c + (1-\alpha) \sum_{e \in E} p_e \phi_e^{sd} \right)$$

$$\sum_{e \in \Gamma^+(v)} z_e^k - \sum_{e \in \Gamma^-(v)} z_e^k = \begin{cases} 1 & \text{if } v = s_k \\ -1 & \text{si } v = t_k \\ 0 & \text{sinon} \end{cases} \quad \forall k \in K \tag{6.1}$$

$$\sum_{k \in K} z_e^k \leq u_e \qquad \forall e \in L_{\text{VN}} \tag{6.2}$$

$$z_e^k \leq \chi_k^{c(e)} \qquad \forall k, \forall e \in L_{\text{VN}} \tag{6.3}$$

Les contraintes (6.1) et (6.2) sont les contraintes classiques de conservation du flot et de capacité des liens du réseau. La contrainte (6.3) assure que chaque fois qu'une requête utilise un lien e, la variable de couleur correspondant à la couleur de ce lien vaut 1 ce qui indique que la requête utilise cette couleur. Il y a $|T|(|E_{\text{VN}}| + |\mathcal{C}|)$ variables et $|T|(|V_{\text{VN}}| + |E_{\text{VN}}|) + |E_{\text{VN}}|$ contraintes.

6.1.4.2 Formulation arc-chemin

ϕ_P^{sd} est une variable binaire qui vaut 1 si la requête $t_{sd} \in T$ est routée sur un chemin $P \in \mathcal{P}$, et 0 sinon. Le nombre de chemins permettant de router une requête peut être exponentiel, il se peut qu'il y ait un nombre exponentiel de variables de ce type. De même que dans la formulation sommet-arc, des variables binaires χ_{sd}^c indiquent si un chemin qui est utilisé par une requête utilise un arc de couleur c ou pas.

Objectif :

$$\min \frac{1}{|T|} \sum_{t_{sd}} \left(\alpha \sum_{c \in C} \chi_{sd}^{c(e)} + (1-\alpha) \sum_{e \in E_{\mathrm{VN}}} \sum_{P \in \mathcal{P}^{sd} | e \in P} p_e \phi_P^{sd} \right)$$

Contraintes :

$$\sum_{P \in \mathcal{P}^{sd}} \phi_P^{sd} = 1 \qquad \forall t_{sd} \in T \qquad (\sigma_{sd}) \tag{6.4}$$

$$\sum_{t_{sd}} \sum_{P \in \mathcal{P}^{sd} | e \in P} \phi_P^{sd} \leq u_e \quad \forall e \in E_{\mathrm{VN}} \qquad (\omega_e) \tag{6.5}$$

$$\sum_{P \in \mathcal{P}^{sd} | e \in P} \phi_P^{sd} \leq \chi_{sd}^{c(e)} \quad \forall t_{sd} \in T, \forall e \in E_{\mathrm{VN}} \quad (\pi_e^{sd}) \tag{6.6}$$

Les contraintes (6.4) assurent que toutes les requêtes sont bien satisfaites ; σ_{sd} sont les variables duales associées. La contrainte de capacité (6.5) sur chaque lien implique qu'il ne peut pas y avoir plus de chemins utilisant un même lien e que la capacité de ce lien le permet ; ω_e sont les variables duales associées. De même que dans la formulation sommet-arc, les contraintes (6.6) indiquent les couleurs utilisées pour servir chaque requêtes, les variables duales associées sont (π_e^{sd}).

La fonction objectif est la même.

Pour résoudre ce programme linéaire, la technique de génération de colonnes est utilisée. Il y a $|T|(|E_{\mathrm{VN}}|+1)+|E_{\mathrm{VN}}|$ contraintes et de manière empirique, au plus $10|T|$ variables sont générées. Cela est bien plus faible que dans la formulation sommet-arc.

6.1.4.3 La génération de colonnes appliquée à MACF

Problème maître. Le problème maître est une restriction de la formulation arc-chemin sur un sous-ensemble des chemins disponibles, c.à.d. que seule une partie des variables $\{\phi_P^{sd}\}_{P \in \mathcal{P}^{sd}}$ est considérée. Les autres variables n'existent tout simplement pas.

Problème auxiliaire. Étant donnée une requête t_{sd}, le problème auxiliaire est un problème de plus court chemin sur le graphe D_{VN} avec comme coût sur les liens $\frac{(1-\alpha)}{|T|} p_e + \omega_e + \pi_e^{sd}$. Le coût réduit d'un chemin $P \in \mathcal{P}^{sd}$ est :

$$\sigma_{sd} - \sum_{e \in P} \left(\frac{(1-\alpha)}{|T|} p_e + \omega + \pi_e^{sd} \right)$$

Un chemin ayant un coût positif permet d'améliorer la solution du problème maître.

Afin de garantir une solution optimale, nous devons résoudre le problème auxiliaire pour chaque requête $t_{sd} \in T$. Étant donné que le coût d'une couleur ($\chi_{sd}^{c(e)}$) n'est pas le même pour chaque requête, le coût assigné aux arcs varie pour chaque requête.

Génération de la solution initiale. Pour générer la solution initiale, un multiflot classique est résolu, sans tenir compte des couleurs.

Solution entière. Dans la solution obtenue, les variables ϕ_P^{sd} et χ_{sd}^c sont fractionnaires. Il est nécessaire de faire du branchement (fixer la valeur d'une variable et résoudre de nouveau le problème de manière récursive) pour trouver une solution entière. Il est intéressant de noter qu'il n'est pas nécessaire de brancher sur les variables χ_{sd}^c car il est possible de les considérer comme fractionnaires. En effet, puisque toutes les requêtes sont unitaires, lorsque les variables de flot sont entières, les contraintes (6.6) forcent les variables χ_{sd}^c à être entières.

6.1.5 Résultats expérimentaux

6.1.5.1 Génération des instances

Pour comparer les deux formulations proposées, nous avons généré des exemples à partir d'un graphe orienté représentant un réseau physique et d'un graphe orienté représentant un réseau virtuel.

Les réseaux physiques ont été conçus à partir de graphes, que nous avons transformés en graphes orientés en remplaçant chaque arête par un circuit de longueur deux. À chaque arc des graphes orientés obtenus est associée une couleur différente (chaque couleur représentant un groupe de risques). Si les réseaux physiques possèdent déjà des groupes de risques, ils peuvent tout aussi bien être utilisés.

Ensuite, un réseau virtuel est généré en plusieurs étapes :

1. Un graphe non orienté sur l'ensemble des sommets du réseau physique est généré en utilisant un générateur de graphes aléatoires implémenté dans MASCOPT. Le graphe aléatoire requis, a un diamètre d'au plus trois, une coupe minimale d'au moins deux et environ trois fois plus d'arêtes que de sommets.
2. Chaque arête est remplacée par un circuit de longueur deux.
3. Des requêtes unitaires sont générées de manière aléatoire.
4. Un *double multiflot* est calculé : les requêtes sont routées sur le graphe orienté généré précédemment qui lui-même est routé simultanément sur le graphe orienté représentant le réseau physique.
5. À chaque arc du graphe orienté généré correspond un ensemble de liens virtuels parallèles : chaque arc du graphe orienté généré est routé sur un ensemble de chemins, chacun donnant un lien virtuel différent. À chaque lien virtuel sont associées les couleurs des ressources physiques qu'il utilise. Ceci donne le réseau virtuel qui est utilisé ainsi que ses groupes de risques.

6. Le routage des requêtes dans le réseau virtuel donne un poids aux liens virtuels. Chaque liens virtuel v_l a un poids poids(v_l) dans le routage ci-dessus. Étant donné un lien physique p_l sur lequel v_l est routé, la capacité locale de v_l est notée : capacité$_{locale}(v_l)$, la proportion de capacité de p_l qui peut être attribuée à v_l si la capacité de p_l est répartie entre les liens virtuels l'utilisant $v'_l \in p_l$ proportionnellement à leur poids :

$$\text{capacité}_{locale}(v_l) = \frac{\text{poid}(v_l)}{\sum_{v'_l \in p_l} \text{poid}(v'_l)} \text{capacité}(p_l)$$

   La capacité donnée à chaque lien virtuel v_l est alors le minimum des capacités locales : capacité$(v_l) = \min_{p_l \ni v_l}(\text{capacité}_{locale}(v_l))$

7. Enfin, chaque arc est remplacé par une chaîne d'arcs, chacun des nouveaux arcs appartenant à un unique groupe de risques [CDP+07].

6.1.5.2 Résultats

Les tests ont été réalisés sur un *Intel* Core 2 2.4 GHz avec 4Mb de L1 cache et 2 Gb de mémoire, le programme a été écrit en utilisant la bibliothèque Mascopt. Nous avons utilisé comme réseaux physiques les réseaux NSFNET (14 nœuds et 21 liens) et BRAZIL (27 nœuds et 70 liens) [NR06]. Le nombre de requêtes était 140 pour NSFNET et 200 pour BRAZIL. Nous avons réalisé dix tests pour chaque instance du problème, demandant à chaque fois une solution entière à au plus 5 % de la solution optimale (sauf pour $\alpha \geq 0.75$, où nous avons demandé 10 %).

Je présente nos résultats sur deux graphiques, un pour BRAZIL et un pour NSFNET. La hauteur des rectangles représente le temps de calcul. Pour la formulation arc-chemin, le rectangle est partagé en trois : le temps passé à résoudre le problème maître, à résoudre le problème auxiliaire et à trouver une solution entière.

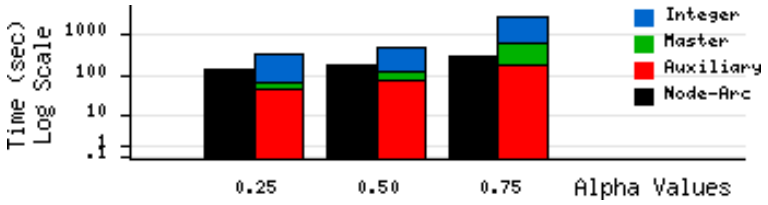


FIG. 6.4 – Comparaison des temps de calcul pour le réseau BRAZIL.

Temps de calcul. De manière surprenante le temps de calcul de la formulation sommet-arc est toujours inférieur à celui de la formulation arc-chemin utilisant la génération de colonnes alors que le nombre de variables est bien plus élevé. Cela peut en partie être expliqué par le fait que pour implémenter la génération de colonnes, nous avons été amenés à utiliser une interface java qui fait appel à CPLEX puis à un Branch

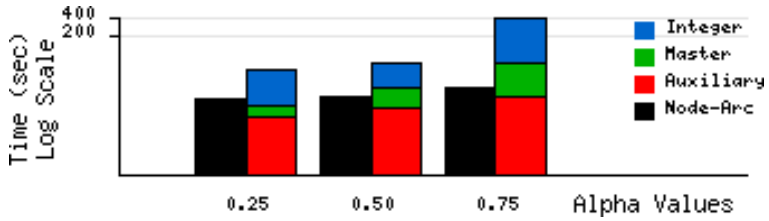


FIG. 6.5 – Comparaison des temps de calcul pour le réseau NSFNET.

and Cut "fait main" car il n'est pas possible de faire les deux simultanément avec Cplex. La formulation sommet-arc peut, quant à elle, être totalement résolue par CPLEX.

Une autre explication peut être que dans la formulation sommet-arc, les contraintes sur les couleurs ne font intervenir que deux variables alors que dans la formulation arc-chemin elles en font intervenir davantage. Les variables de couleurs étant les plus nombreuses cela peut avoir un impact sur le temps de calcul.

De plus, dans le cas de la formulation sommet-arc, à chaque fois que le problème maître est résolu, sont résolus autant de problèmes auxiliaires qu'il y a de requêtes (c.f. 6.1.4.3).

Taille des problèmes résolus. La formulation arc-chemin sous forme de génération de colonnes permet de résoudre des problèmes plus gros que la formulation sommet-arc dans laquelle le nombre de variables est trop grand. Par exemple, lorsque nous avons testé nos programmes en utilisant le réseau BRAZIL comme réseau physique et que l'on a cherché à router 400 requêtes, la formulation arc-chemin a pu donner une solution alors que la formulation sommet-arc n'a pas donné de solution, car elle entrainait un dépassement de mémoire.

Influence des couleurs. Le temps de résolution augmente avec α pour les deux formulations car les variables de couleurs ont une influence de plus en plus grande. Or ce sont des variables binaires et elles sont nombreuses. Il est important de noter que dans la seconde formulation, la qualité de la solution initiale décroit quand α augmente car elle ne prend pas en compte les groupes de risques.

6.1.6 Perspectives

Dans cette section, l'objectif était de maximiser une combinaison linéaire de la fiabilité et du coût du routage d'un ensemble de requêtes unicasts dans un réseau maillé avec groupes de risques. Le problème MACF a été montré NP-dur et dur à approcher. Enfin, une formulation sous forme de génération de colonnes qui permet d'obtenir des solutions optimales ou presque en un temps raisonnable a été présentée.

Il est maintenant intéressant de continuer ce travail et de présenter une formulation pour le routage de multicasts et pour assurer la protection des requêtes en cas de pannes. Concernant la protection des requêtes unicasts, cela a commencé à être étudié

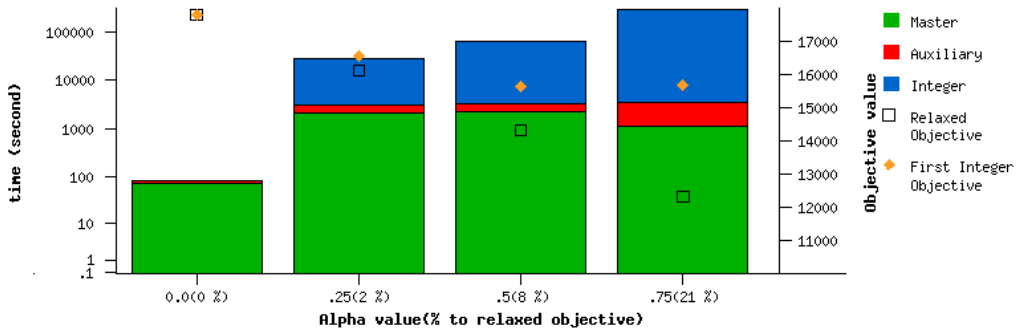


FIG. 6.6 – Évolution du temps de calcul pour le réseau Brésilien.

dans [YVJ05]. Les auteurs y considèrent, comme mode de protection, la protection par chemins en minimisant soit le nombre total de couleurs soit le nombre de couleurs communes. Ce travail mérite d'être continué en considérant d'autres types de protection. Dans la section suivante, je fais un premier pas dans cette direction en étudiant un mode de protection alternatif dans des réseaux sans groupes de risques. J'y présente une formulation mathématique permettant de calculer des protections partagées par segments, pour un ensemble de chemins préétablis.

6.2 Protection par segments

Je présente un nouveau type de protection partagée spécialement conçu pour protéger des routages utilisant la conversion de longueur d'onde dans des réseaux WDM. Son originalité tient du fait qu'elle exploite les équipements déjà installés aux extrémités des segment des chemins du routage à protéger. Contrairement à la protection par lien ou par chemin (dont des formulations mathématiques peuvent être trouvées dans le livre de A.K. Somani [Som06]), la protection par segment a été peu étudiée, elle permet pourtant de réaliser un compromis intéressant entre la capacité utilisée par la protection et le temps nécessaire pour mettre en place le nouveau routage en cas de panne. Je présente deux types de protection par segments différents : la protection par segments partagés de base et la protection par segments partagés avec overlaps. Ce dernier type de protection est à comparer à un autre, introduit par P-H. Ho et H.T. Mouftah dans [HM02], tant en terme d'architecture que de mécanisme de signalisation.

6.2.1 Contexte

Groupage de trafic. Comme la taille d'une requête peut être plus petite que la capacité de transfert offerte par une longueur d'onde, une technique, appelée groupage de trafic, consiste à regrouper plusieurs requêtes sur une même longueur d'onde. Cela peut être réalisé en faisant du multiplexage temporel, comme c'est le cas dans les

réseaux SONET/SDH. Le problème de déterminer les requêtes à assembler et quels chemins et longueurs d'onde leur associer est appelé le **problème de groupage, routage et affectation de longueur d'onde** (**Grooming, Routing and Wavelength Assignment GRWA**), ce problème a largement été étudié, entre autre dans [DR02, ML01, Som06, ZM03, HL04] ou encore dans le chapitre 9 du livre de A.K. Somani [Som06].

ADM. Dans un routage, à chaque endroit où plusieurs requêtes sont groupées, un équipement spécifique est nécessaire, on parle d'Add Drop Multiplexer (ADM). Ceux qui sont considérés dans cette étude convertissent les signaux optiques en signaux électriques pour pouvoir effectuer les opérations nécessaires : associer et dissocier des messages. Les ADM considérés diffèrent des OADM présentés dans la Section 5.1.1.5, mais leur principe de fonctionnement et leur rôle sont similaires.

En plus d'associer et de dissocier des messages, les ADM permettent de changer la longueur d'onde d'un signal, en effet, il suffit de le réémettre le signal sur une longueur d'onde différente de celle sur laquelle il a été reçu.

Routage avec conversion de longueur d'onde. La conversion étant "gratuite" aux nœuds équipés d'ADM, dans cette section, elle y est autorisée. Le Problème 5.1 est donc moins contraint puisqu'un chemin peut utiliser plusieurs longueurs d'onde.

Routage par segments. Étant donné un routage, un **segment de travail** (noté σ_w) est une portion de chemin maximale sur laquelle un ensemble de requêtes sont regroupées et se propagent de manière optique, sans conversion opto-électrique. Un segment de travail est délimité par deux ADM. Comme un ADM est composé de cartes, chacune ayant un port d'entrée et un port de sortie, un segment de travail utilise un port de sortie du premier ADM et un port d'entrée du dernier ADM. Le nombre de cartes nécessaires dans un réseau est un facteur déterminant de son coût [Som06, HDR06]. Or, l'ajout de protection va nécessiter l'installation de cartes supplémentaires. C'est ce nombre de cartes supplémentaires nécessaires, que nous allons utiliser comme indicateur du coût de la protection.

Dans cette section, il est supposé que le chemin de chacune des requêtes est composé de 1 à 3 segments de travail. Cette contrainte permet d'assurer que le temps de transmission d'un paquet n'est pas trop élevé. Cette segmentation va être exploitée par notre mode de protection.

6.2.2 Protection par segments partagés

Le principe de la protection par segments est de protéger un ensemble de liens (appelés segments) simultanément, et non pas tout le chemin simultanément (protection par chemin) ou chaque lien indépendamment (protection par liens). L'ensemble de liens permettant de protéger un segment est appelé **segment de protection** et noté σ_p. Il est question de protection partagée car un segment de protection peut être utilisé pour protéger plusieurs segments de travail du routage, comme illustré sur la Figure 6.7.

L'idée de protection par segments a déjà été utilisée par Bouffard dans [Bou05] ou Ho et Mouftah dans [HM02]. Cependant dans leurs études, ils ne prennent pas en compte la structure du routage déjà établie. L'objectif des deux types de protection que je propose est justement de s'en servir. L'idée commune aux deux types de protection que je présente est en effet de protéger les segments de travail du routage et non pas des segments calculés de manière arbitraire.

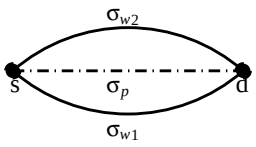


FIG. 6.7 – Protection par segments partagés.

6.2.2.1 Protection par segments partagés de base

Dans ce cas, chaque segment de travail est protégé de bout en bout. Il est aussi possible de protéger plusieurs segments de travail simultanément. Ce type de protection est illustré sur la Figure 6.8.

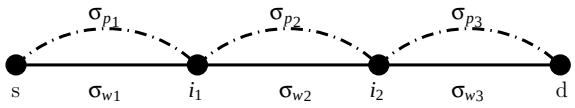


FIG. 6.8 – Protection par segments partagés de base.

Cette protection ne protège pas des pannes aux nœuds équipés d'ADM.

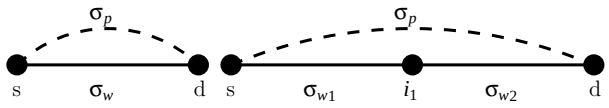
6.2.2.2 Protection par segments partagés avec overlaps

Dans cette protection, chaque segment de travail est protégé avec au moins un autre segment de travail. Cela permet de protéger les nœuds auxquels sont installés les ADM. Les différents cas sont illustrés sur la Figure 6.9.

Cette protection permet de protéger le réseau contre les pannes des liens et des nœuds simultanément.

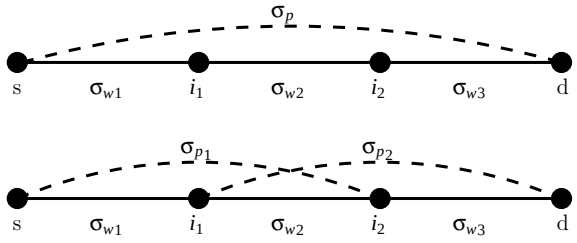
6.2.2.3 Comparaison des deux modes de protection

Suivant le réseau, l'un des deux modes de protection peut être plus cher que l'autre, tant en terme de bande passante utilisée que de nombre d'ADM nécessaires. Cela est illustré par la Figure 6.10 qui est détaillée ci-dessous. Ainsi, à coût égal ou similaire, le deuxième doit être préféré au premier car il protège également contre les pannes des nœuds.



(a) Une requête avec 1 segment de travail.

(b) Une requête avec 2 segments de travail.



(c) Une requête avec 3 segments de travail.

FIG. 6.9 – Possibilités pour une protection par segments partagés avec overlaps.

La Figure 6.10, représente deux exemples de routage par segments pouvant survenir dans un réseau. Le premier exemple, Figure 6.10(a) représente un ensemble de cinq requêtes $\{k_1, k_2, k_3, k_4, k_5\}$:

- k_1, routé de s à i_1 sur un segment,
- k_2, routé de s à i_2 sur deux segments,
- k_3, routé de i_1 à i_2 sur un segment,
- k_4, routé de i_1 à d sur deux segments,
- k_5, routé de i_1 à d sur un segment.

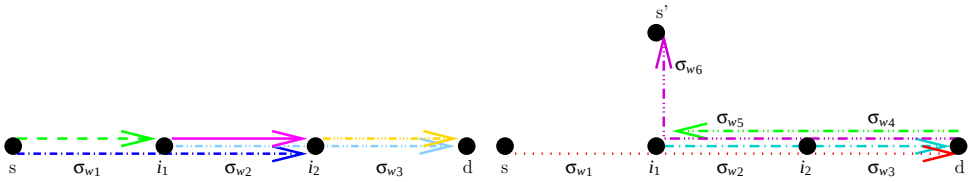
Ces requêtes sont regroupées de la manière suivante :

- k_1 et k_2 sont regroupés de s à i_1 et forment σ_{w1},
- k_2, k_3 et k_4 sont regroupés de i_1 à i_2 et forment σ_{w2},
- k_4 et k_5 sont regroupés de i_2 à d et forment σ_{w3}.

Dans cet exemple, une protection par segments partagés avec overlap nécessite cinq segments de protection et huit cartes d'ADM (deux par nœuds). Cela est plus coûteux qu'une protection par segments partagés de base qui nécessite seulement trois segments de protection et quatre cartes d'ADM (une par nœuds), comme indiqué sur la Figure 6.11.

Le deuxième exemple représente quatre requêtes $\{k_1, k_2, k_3, k_4\}$:

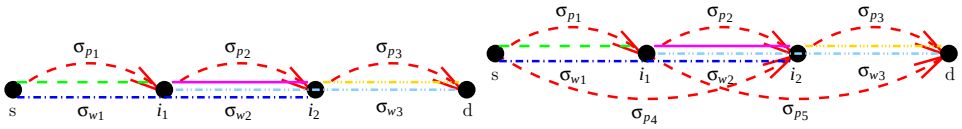
- k_1, routé de s à d
- k_2, routé de d à s'
- k_3, routé de i_1 à d



(a) Un premier ensemble de routage par segment.

(b) Un second ensemble de routage par segment.

FIG. 6.10 – Deux types interactions possibles entre des segments de travail.



(a) Sans overlap : trois segments de protection et quatre cartes d'ADM.

(b) Avec overlap : cinq segments de protection et huit cartes d'ADM.

FIG. 6.11 – Comparaison des deux protections pour l'exemple de la Figure 6.10(a).

– k_4, routé de d à i_1

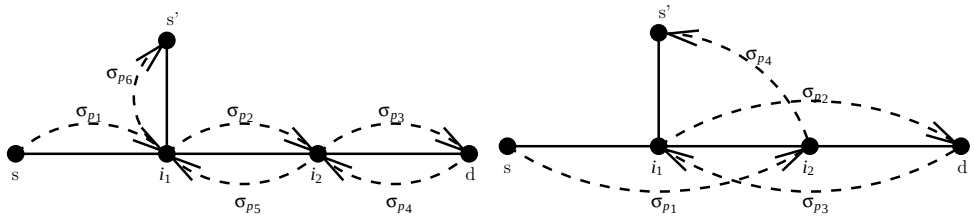
Ces requêtes sont routées en formant des segments de travail de la manière suivante :

– k_1 est routé de s à i_1, ce qui forme σ_{p_1},
– k_1 et k_3 sont regroupés de i_1 à i_2, ce qui forme σ_{p_2},
– k_1 et k_3 sont regroupés de i_2 à d ce qui forme σ_{p_3},
– k_2 et k_4 sont regroupés de d à i_2 ce qui forme σ_{p_4},
– k_2 et k_4 sont regroupés de i_2 à i_1 ce qui forme σ_{p_5},
– k_2 est routés de i_1 à s' ce qui forme σ_{p_6},

Dans cet exemple, une protection par segments partagés avec overlap nécessite cinq cartes d'ADM (une par nœuds), alors qu'une protection par segments partagés de base nécessite sept cartes d'ADM. (deux pour les nœuds i_1 et i_2, une aux autres nœuds). Cela est représenté sur la Figure 6.12.

6.2.3 Formulations mathématiques

Dans l'Annexe D.2, sont présentées des formulations mathématiques pour ces deux problèmes. N. Bhuiyan, de l'université de Concordia, est actuellement en train de les implémenter. L'objectif est de comparer les performances de ces formulations avec des algorithmes déjà existants.



(a) Sans overlap : six segments de protection et sept cartes d'ADM.

(b) Avec overlap : quatre segments de protection et cinq cartes d'ADM.

FIG. 6.12 – Comparaison des deux protections pour l'exemple de la Figure 6.10(b).

6.3 Conclusion et Perspectives

Dans ce chapitre, j'ai présenté des problèmes relatifs à la fiabilité des connexions établies. Ces problèmes sont essentiels pour satisfaire les clients. La première partie concerne les risques de pannes multiples. J'y ai présenté un moyen d'en tenir compte lors du routage. L'efficacité de la formulation mathématique proposée est illustrée par des résultats expérimentaux. Dans la deuxième partie, je présente deux nouveaux types de protections ; Je présente leur mode de fonctionnement et, ensuite, je les compare entre eux. Les modèles mathématiques correspondants sont en cours d'implémentation et vont permettre de comparer ces types de protections à d'autres déjà existants.

L'étape suivante est de "fusionner" les formulations mathématiques des deux sections afin de prendre en compte les risques de pannes multiples lors de la protection de requêtes. Il serait également intéressant d'étendre ces formulations aux multicasts.

Chapitre 7

Reroutage de connexions et Pathwidth

Dans ce chapitre, je présente un invariant de graphe appelé process number qui a été introduit pour modéliser les reconfigurations d'un routage d'un réseau WDM dont les ressources ou les requêtes évoluent dynamiquement. Je présente les liens entre cet invariant et d'autres invariants dont la pathwidth. Ensuite, je présente un algorithme permettant de calculer le process number d'un arbre, ainsi que des avancées sur les liens entre la pathwidth d'un graphe planaire extérieur et celle de son dual (à savoir que pour tout graphe graphe planaire extérieur biconnexe G sans boucles ni arêtes multiples, on a $pw(G^*) \leq pw(G) \leq 2pw(G^*) - 1$) et sur les liens entre la pathwidth d'un graphe planaire et celle son dual (pour tout graphe planaire 3-connexe G, on a $\frac{1}{3}pw(G^*) - 2 \leq pw(G) \leq 3pw(G^*) + 2$). Je finis sur des considérations concernant l'indice d'échappement arête connexe.

L'étude de la reconfiguration d'un routage à l'intérieur d'un réseau WDM a mené à introduire un invariant : le process number, c.f. [CS07]. Dans cette section, je commence par en rappeler la définition ainsi que celles de paramètres qui lui sont reliés comme la pathwidth notamment. Ensuite, je présente les résultats que j'ai obtenus au cours de ma thèse : un algorithme distribué permettant de calculer divers paramètres sur les arbres, un algorithme d'approximation pour calculer la pathwidth de graphes planaires extérieurs et des avancées sur plusieurs conjectures.

7.1 Origine du problème

Le principe du WDM (Wavelength Division Multiplexing), qui est rappelé en Section 5.1.1.5, est de permettre à une même fibre optique de transmettre plusieurs longueurs d'onde, augmentant ainsi sa capacité. Un **routage** dans un réseau WDM est

une affectation à chaque requête d'un chemin et d'une longueur d'onde de telle sorte que deux requêtes dont les chemins partagent une arête aient deux longueurs d'onde différentes (c.f. Problème 5.1). Je me restreins ici au cas où il n'y a pas de conversion de longueur d'onde. Cependant, pour tenir compte de la conversion de longueur d'onde, il suffit de partager une requête en plusieurs sous requêtes. Au cours des ajouts et des suppressions de requêtes dus à l'évolution du trafic, si le routage des requêtes déjà présentes n'est pas modifié, il se peut que l'on se trouve dans des configurations ne permettant plus d'ajouter les nouvelles requêtes alors que les ressources sont suffisantes. Ainsi, il est nécessaire de changer le routage des requêtes présentes dans le réseau de temps en temps. Ce problème est illustré par l'exemple suivant.

Nous considérons une fibre ayant deux longueurs d'onde λ_1 et λ_2, dans laquelle sont effectuées les opérations suivantes : routage de la requête A - B sur λ_1, routage de la requête A - C sur λ_2, routage de la requête B - E sur λ_1, routage de la requête D - E sur λ_2, suppression de la requête B - E et enfin routage de la requête C - E sur λ_1. La succession de ces modifications du trafic est représentée sur la Figure 7.1.

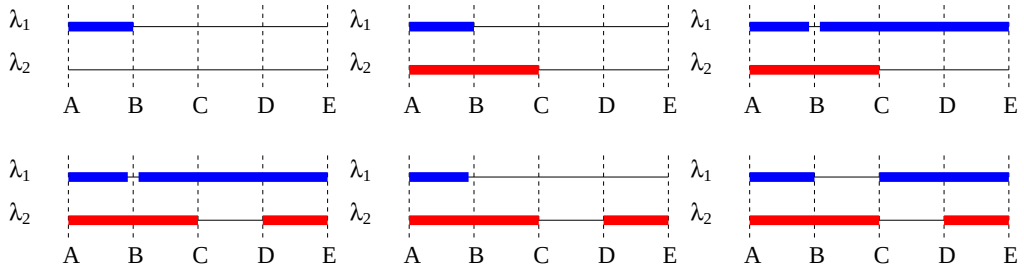


FIG. 7.1 – Exemple d'opérations successives entraînant un blocage.

Le réseau se retrouve dans une configuration telle qu'il n'est pas possible d'ajouter la requête B - D alors que la Figure 7.2 montre qu'un autre routage des requêtes déjà présentes le permet. Il faudrait déplacer la requête C - E sur λ_2 et la requête D - E sur λ_1. Seulement, pour déplacer ces deux requêtes, il faut interrompre l'une des deux car chacune des requêtes veut prendre la place de l'autre. Pour éviter cette interruption, une solution consiste à utiliser une longueur d'onde supplémentaire λ_3 que l'on ne peut utiliser que lors de la reconfiguration. On peut alors déplacer la requête C - E sur λ_3, la requête D - E sur λ_1, la requête C - E sur λ_2 et enfin router la requête B - D sur λ_2. Contrairement à une suppression puis à l'ajout d'une requête, le déplacement d'une requête peut se faire sans interruption de service.

Nous supposons dans la suite que les longueurs d'onde supplémentaires, que l'on appelle **mémoires temporaires** (temporary memory units **TMU**), ne peuvent supporter qu'une seule requête chacune. La question est de savoir quel est le nombre minimum de mémoires temporaires nécessaires pour passer d'un routage à un autre. Si l'on ne dispose pas de mémoires temporaires, il s'agit de savoir quel est le nombre minimum de connexions qu'il va falloir arrêter simultanément.

On modélise le changement d'une configuration C à une configuration C' par un

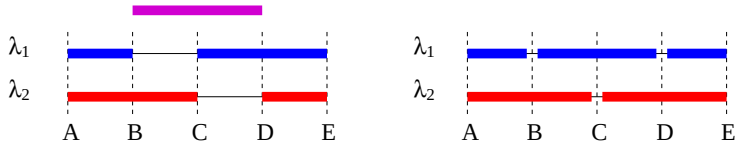


FIG. 7.2 – Illustration du blocage et solution.

graphe orienté des conflits D comme suit : chaque requête est représentée par un sommet et on met un arc d'un sommet représentant une requête r vers un sommet représentant une requête s si dans la configuration C, s utilise une ressource (dans notre cas, une ressource est une paire lien-longueur d'onde) utilisée par r dans C'. La Figure 7.3 représente le graphe des conflits dans le cas de l'exemple de la Figure 7.2

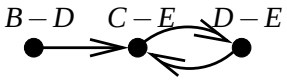


FIG. 7.3 – Exemple de graphe de conflit.

Passer de la configuration C à la configuration C' en utilisant des mémoires temporaires revient à supprimer tous les sommets du graphe orienté auxiliaire en respectant les règles suivantes :

- On peut à tout moment mettre un sommet dans une mémoire temporaire. ce faisant, on supprime tous les arcs entrant de ce sommet.
- On peut supprimer un sommet qui n'a pas d'arc sortant, si le sommet était dans une mémoire temporaire, il la libère.

Une suite d'actions permettant de supprimer tous les sommets d'un graphe orienté D est appelée **stratégie de reroutage**. Le nombre maximum de mémoires temporaires simultanément utilisées par une stratégie de reroutage est appelé **coût de la stratégie de reroutage**, le minimum des coûts des stratégies de reroutage est le **process number** du graphe orienté, on le note $pn(D)$.

Étant donné un graphe (non orienté) G, le process number de G est le process number du graphe orienté obtenu de G en remplaçant chaque arête par un circuit orienté de longueur deux, soit deux arcs, un dans chaque direction. Pour la suite, il est intéressant de remarquer que dans le cas des graphes (simples), cette définition est équivalente à une définition faisant référence à un jeux tour à tour dans lequel un ensemble d'agents souhaitent capturer un fugitif, ou de manière équivalente s'assurer qu'il ne peut se cacher dans aucun sommet.

Le process number d'un graphe est le nombre minimum d'agents nécessaires pour traiter tous les sommets d'un graphe en respectant les règles suivantes :

- Positionner un agent sur un sommet (on dit d'un sommet sur lequel est positionné un agent qu'il est **occupé par un agent**).
- Traiter un sommet **encerclé** (un sommet tel qu'un agent est positionné sur cha-

cun de ses voisins).
- Enlever un **agent inutile** (un agent tel que chaque sommet voisin du sommet sur lequel il est positionné est soit traité soit occupé par un agent) et traiter le sommet sur lequel il était. Cet agent peut ensuite être réutilisé ailleurs.

Cette définition du process number (qui est valable si l'on s'intéresse aux graphes (simples) mais ne l'est pas dans le cas des graphes orientés) permet de mieux illustrer les similitudes entre les stratégies de reroutage et les stratégies sommet définies dans la Section 7.2.4.

Enfin,un graphe ou un graphe orienté qui peut être traité en utilisant k TMU, est dit k-**mémoire**.

7.2 État de l'art

7.2.1 Process number

Les premiers résultats concernant le process number sont dus au travail de P. Quang Cuòng durant son stage de Master encadré par D. Coudert. Ils sont disponibles dans son rapport de stage [Pha04]. Il y montre, entre autres, que calculer le process number d'un graphe orienté est un problème NP-complet. Le process number de certaines classes de graphes a été déterminé dans [CPPS05]. Le théorème suivant y est démontré (voir Section 2.1 pour les définitions des différents graphes) :

Théorème 7.1 ([CPPS05])

- *Le process number d'un graphe biparti complet* $K_{m,n}$ *est* $\min(m,n)$.
- *Le process number d'un arbre binaire complet de hauteur* h *est* $h-1$, *sauf si* $h \in \{1,2\}$ *au quel cas c'est* h.
- *Pour tout* $d \geq 3$, *le process number d'un arbre complet de degré* $d+1$ *de hauteur* h *est* h.
- *Le process number d'une grille de taille* $m \times n$ *est* $\min(m,n)+1$, *sauf si* $n = m = 2$ *au quel cas sont process number vaut* 2.
- *Le process number d'une pyramide de taille* n *est* $\lceil \frac{n}{2} \rceil + 1$, *sauf si* $n \in \{2,3\}$ *au quel cas c'est* $n-1$.
- *N'importe quel graphe planaire extérieur triangulé dont le dual faible est un caterpillar de degré maximum 3 peut être traité avec 3 TMUs.*

Une caractérisation de l'ensemble des graphes orientés 0-, 1-mémoire ainsi qu'une caractérisation partielle des graphes 2-mémoire est également donnée dans [Pha04]. Pham Quang Cuòng y donne un algorithme en $O(n+m)$ pour tester si un graphe est 0-mémoire, un algorithme en $O(n^2)$ pour tester si un graphe est 1-mémoire et un algorithme en $O(n^4)$ pour détecter une partie des graphes 2-mémoire. Par la suite, D. Coudert et J-S. Séréni [CS07] donnent un algorithme linéaire permettant de reconnaître et de traiter les graphes orientés 1-mémoire. Dans le même article, ils donnent une caractérisation par mineurs exclus des graphes 2-mémoire et donnent un algorithme linéaire simple permettant de les reconnaître. Les auteurs de [CS07] donnent

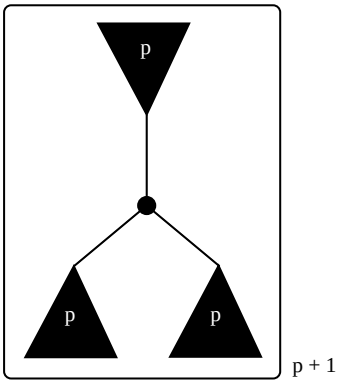


FIG. 7.4 – Un arbre de process number $p+1$ a au moins trois branches de process number p.

également une caractérisation récursive des graphes orientés 2-mémoire et en déduisent un algorithme en $O(n^2(n+m))$ pour les traiter.

Enfin, dans [CPPS05] les auteurs donnent une caractérisation des arbres de process number $p+1$ similaire à celle proposée par Scheffler [Sch90] concernant la pathwidth des arbres (voir Lemme 7.5 ci-dessous).

Lemme 7.2 (Fig. 7.4) *Pour $p \geq 2$, pour tout arbre T, $pn(T) \geq p+1$ ss'il existe un sommet v tel que la forêt $T \setminus \{v\}$ a au moins trois composantes dont le process number est au moins p.*

7.2.2 Vertex separation

L'intérêt de l'étude du process number est accru par le résultat présenté dans la Proposition 7.3, prouvée par Pham Quang Cuòng dans [Pha04], qui relie le process number à la **vertex separation**. La vertex separation est un invariant important d'un graphe ou d'un graphe orienté définit comme suit : étant donné un graphe orienté $D = (V,A)$ et un ensemble de sommets X, le voisinage sortant de X est $\Gamma^+(X) = \{v \in V \setminus X : \exists u \in X : (u,v) \in A\}$. Un **ordonnancement** (layout) L des sommets de D est une bijection entre V et $\{1,\ldots,|V|\}$. La vertex separation de (D,L) est la taille du plus grand voisinage sortant des ensembles $\{L^{-1}(1),\ldots,L^{-1}(i)\}$ pour $1 \leq i \leq n$. La vertex separation de D est la plus petite vertex separation de (D,L) sur tous les agencements L de V, on la note $vs(D)$. La vertex separation s'étend de la même manière que le process number aux graphes non orientés.

Le Théorème 7.3 établit le lien qui existe entre le process number et la vertex separation d'un graphe orienté :

Proposition 7.3 ([Pha04]) *Pour tout graphe orienté D, on a $vs(D) \leq pn(D) \leq vs(D)+1$.*

Dans [CS07], D. Coudert et J-S. Séréni donnent un critère permettant de décider si pour un graphe orienté D on a $pn(D) = vs(D)+1$:

Proposition 7.4 ([CS07]) *Pour tout graphe orienté D, il existe une stratégie de reroutage utilisant $pn(D)$ TMU et telle que tout sommet est placé en TMU ssi $pn(D) = vs(D)+1$.*

Cependant ce critère nécessite de connaître une stratégie de reroutage optimale et de savoir vérifier si elle peut être transformée de telle sorte que tout sommet soit placé en TMU.

7.2.3 Pathwidth

Dans le cas des graphes, Kinnersley a prouvé que la vertex separation d'un graphe est égale à sa **pathwidth** (notée $pw(G)$), qui est un autre paramètre introduit par Robertson et Seymour [RS83]. Pour définir ce paramètre, j'ai besoin d'introduire la **décomposition en chemins** d'un graphe $G = (V,E)$. C'est un ensemble d'ensembles $(X_1,\ldots,X_r)$ de V tels que :

1. $\bigcup_{i=1}^{r} X_i = V$
2. $\forall xy \in E, \exists i \in \{1,\ldots,r\} : \{x,y\} \subset X_i$
3. pour tout $1 \leq i_0 < i_1 < i_2 \leq r$, $X_{i_0} \cap X_{i_2} \subseteq X_{i_1}$

La **largeur** de la décomposition en chemins $(X_1,\ldots,X_r)$ est $\max_{1\leq i\leq r} |X_i| - 1$. La **pathwidth** de G, notée $pw(G)$, est la largeur minimale des décompositions en chemins.

Le calcul de la pathwidth d'un graphe est un domaine de recherche actif dans lequel beaucoup de travail a été fait. Le lecteur intéressé pourra se référer au survey de B. Reed [Ree97].

On notera, entre autres, qu'il existe un algorithme linéaire permettant de calculer la pathwidth d'un arbre [Sch90]. Cet algorithme est basé sur la caractérisation suivante des arbres de pathwidth $p+1$:

Lemme 7.5 ([Sch90]) *Pour tout arbre T, $pw(T) \geq p+1$ s'il existe un sommet v tel que la forêt $T \setminus \{v\}$ ait au moins trois composantes dont la pathwidth est au moins p.*

Ce même algorithme permet d'obtenir une décomposition en chemins optimale en temps $O(n\log(n))$. Ce résultat a été amélioré par K. Skodinis qui propose dans [Sko00], un algorithme permettant d'obtenir un ordonnancement des sommets optimal par rapport à la vertex separation en temps linéaire.

7.2.4 Indice d'échappement sommet

La vertex separation et donc la pathwidth sont également équivalents à l'**indice d'échappement sommet** (node search number en anglais), un autre invariant important introduit par L.M. Kirousis et C.H. Papadimitriou dans [KP86] dans le cadre des jeux de recherche. Les jeux de recherches sont des jeux qui opposent tour à tour un

fugitif à des agents. Le but de ces jeux est de déterminer le nombre minimum d'agents nécessaires pour capturer le fugitif dans un graphe sous certaines conditions de déplacement des agents et du fugitif. Dans le cadre d'une stratégie sommet, le fugitif a le droit, à son tour, de se déplacer d'autant d'arêtes qu'il veut dans le graphe tant qu'il ne passe pas par un sommet occupé par un agent. Ensuite, au tour des agents, l'un d'eux peut faire une des deux actions suivantes :

- se positionner sur un sommet s'il n'est pas déjà positionné sur un autre sommet. Le sommet est dit occupé.
- se retirer d'un sommet sur lequel il est positionné afin de pouvoir se positionner sur un autre sommet lors d'un prochain tour.

Le fugitif est capturé s'il se trouve sur un sommet occupé par un agent ou sur une arête dont les deux extrémités sont occupées par un agent. Étant donné un graphe G, le nombre minimum d'agents nécessaires pour capturer le fugitif (invisible pour les agents tant qu'il n'a pas été capturé) quels que soit ses mouvements est appelé l'indice d'échappement sommet. Il est noté $sn(G)$.

Remarque 7.6 L'indice d'échappement sommet peut aussi être vu comme le nombre d'agents nécessaires pour nettoyer tous les sommets d'un graphe de telle sorte qu' initialement tous les sommets soient contaminés et qu'un sommet non occupé par un agent est recontaminé s'il est adjacent à un sommet contaminé. Un sommet contaminé représente un sommet sur lequel le fugitif peut se trouver.

Remarque 7.7 Selon une remarque de S. Thomassé, si l'on rajoute la règle que le fugitif ne peut pas rester immobile (ce qui peut être simulé par la présence d'un chien aidant les policiers), la nouvelle version de l'indice d'échappement sommet obtenue donne le process number. En effet, pour obtenir la règle que les sommets encerclés sont nettoyés, il suffit que les policiers ne bougent qu'un tour sur deux.

Comme N. Nisse dans [Nis07], on appelle une suite de déplacements des policiers permettant de capturer le fugitif quelque soit ses déplacements, une **stratégie de capture sommet**.

Dans [EST94], J.A. Ellis, I.H. Sudborough et J. Türner montrent que l'indice d'échappement sommet d'un graphe est égal à sa pathwidth plus un : $sn(G) = pw(G)+1$.

De plus on peut noter qu'il existe une suite de $O(n)$ mouvements permettant de capturer le fugitif car **recontaminer** une partie du graphe, c.à.d. laisser la possibilité au fugitif de revenir sur un sommet qui a déjà été occupé par un agent, ne permet pas d'utiliser moins d'agents. Ceci est un résultat de A. S. LaPaugh : [LaP93].

Du fait que la recontamination n'aide pas, on peut aussi déduire que si on autorise un agent à se retirer d'un sommet uniquement s'il est inutile (dans le même sens que pour le process number, c.à.d. si chaque sommet voisin au sommet qu'il occupe est soit occupé par un agent soit l'a déjà été) alors le nombre d'agents nécessaires pour capturer le fugitif n'augmente pas. Dans ce cas, les agents suivent les mêmes règles de déplacements que lors du calcul du process number des graphes simples. La différence réside dans le fait que dans une stratégie sommet, on ne peut pas considérer comme

traité un noeud entouré par des agents. Autrement dit, dans le cas d'une stratégie sommet, tous les sommets doivent être visités par un agent.

Ces dernières années de nombreux travaux ont vu le jour sur différentes variantes des jeux de recherche. Le lecteur intéressé pourra lire le survey récemment écrit par F. Fomin et D. Thilikos [FT08] ainsi que la thèse de N. Nisse [Nis07].

J'évoque deux variantes intéressantes. La première est le cas où le fugitif est visible, dans ce cas le nombre d'agents nécessaires à sa capture donne la treewidth. La deuxième variante est quand les agents peuvent demander un nombre limité de fois la position du fugitif. Cette variante donne toute une famille d'intermédiaire entre la pathwidth et la treewidth.

Par la suite, je me contente de présenter les définitions et les résultats dont j'ai besoin.

7.2.5 Indice d'échappement arête

Une **stratégie arête** est une variante d'une stratégie sommet dans laquelle les policiers, en plus des déplacements autorisés dans le cadre des stratégies sommet, peuvent se déplacer le long d'une arête. Dans cette variante, le fugitif est capturé si un agent est sur le même sommet que lui ou si un agent traverse l'arête dans laquelle il est.

Dans le cadre d'une stratégie arête, on appelle une suite de déplacements permettant de capturer le fugitif une **stratégie de capture arête**. Étant donné un graphe G, le nombre minimum d'agents nécessaires à une stratégie de capture arête dans G est l'**indice d'échappement arête** (edge search number), on le note $es(G)$.

Remarque 7.8 L'indice d'échappement arête peut aussi être vu comme le nombre d'agents nécessaires pour nettoyer tous les sommets et toutes les arêtes d'un graphe de telle sorte qu'initialement tout le graphe est contaminé et qu'un sommet non occupé par un agent est recontaminé s'il est adjacent à un élément (sommet ou arête) contaminé, de même pour les arêtes si un de leur sommet non occupé est adjacent à un élément contaminé. Un élément contaminé représente un élément sur lequel le fugitif peut se trouver.

En utilisant un agent supplémentaire, il est facile de transformer une stratégie de capture arête en une stratégie de capture sommet et vice versa. En effet, un agent qui se déplace le long d'une arête dans une stratégie de capture arête peut être remplacé par deux agents, un à chaque extrémité de l'arête, dans une stratégie de capture sommet. Pour passer d'une stratégie de capture sommet à une stratégie de capture arête, il suffit d'utiliser un agent supplémentaire pour visiter chaque arête quand des agents sont positionnés sur ses extrémités.

Ainsi, on obtient que pour tout graphe G :

$$es(G) - 1 \leq ns(G) \leq es(G) + 1.$$

7.2.6 Indice d'échappement arête connexe

On peut aussi citer la variante de la stratégie arête proposée par P. Fraigniaud et al dans [BFFS02] : la **stratégie arête connexe** (contiguous search number). Dans cette variante, la zone dans laquelle le fugitif ne peut pas se trouver, appelée zone sûre, doit être connexe. Ils proposent également une variante pondérée de ce jeu de recherche. Dans la variante pondérée, le poids d'une arête ou d'un sommet indique le nombre d'agents nécessaires pour nettoyer cette arête ou ce sommet. Enfin un sommet doit être gardé par un nombre d'agents supérieur au poids de chaque arête adjacente afin d'empêcher toute recontamination. Une stratégie permettant de capturer à coup sûr un fugitif, est appelée une **stratégie capture arête connexe**

Le nombre minimum d'agents nécessaires pour capturer un fugitif dans un graphe G est appelé **indice d'échappement arête connexe** de G (contiguous search number) et est noté $cs(G)$.

Dans [BFFS02], les auteurs présentent un algorithme linéaire permettant de calculer l'indice d'échappement arête connexe $cs(T)$ d'un arbre pondéré (T,w). Cet algorithme repose sur le Lemme 7.9. Étant donné un arbre de racine u et un noeud v, on note T_u le sous-arbre maximal connexe de racine u ne contenant pas d'arêtes du chemin de u à v.

Lemme 7.9 *Étant donné un arbre pondéré (T,w) de racine y où w est une fonction de $V \cup E \to \mathbb{N}$, si $y_1,\ldots,y_p$ sont les fils de y tels que $cs(T_{y_i}) \geq cs(T_{y_{i+1}})$ pour $1 \leq i < p$, alors $cs(T) = \max(cs(T_{y_1}), cs(T_{y_2}) + w(y_2))$.*

Dans le cas non pondéré, ce lemme est exact car on peut montrer que, traiter un sous-arbre T_{y_i} partiellement avant d'en traiter un autre n'aide pas. Par contre, dans le cas pondéré, ce lemme est incorrect comme l'illustre l'exemple de la Figure 7.5. L'erreur provient du fait que, dans le cas où les agents sont positionnés sur le sommet y, il est moins coûteux d'envoyer un seul agent garder le sommet y_1 avant de nettoyer l'arbre T_{y_2} plutôt que de laisser 10 agents sur y afin de le garder pendant que l'on nettoie T_{y_2}.

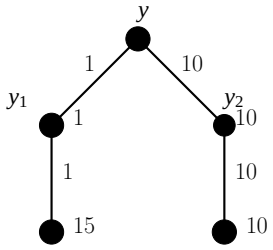


FIG. 7.5 – Contre exemple au Lemme 7.9.

Cet exemple permet de construire un arbre (cf Figure 7.6) pour lequel l'algorithme présenté dans [BFFS02] renvoie une valeur fausse : 16 au lieu de 15.

Dans la Section 7.3.4, je présente de nouveaux résultats concernant cette variante de l'indice d'échappement arête.

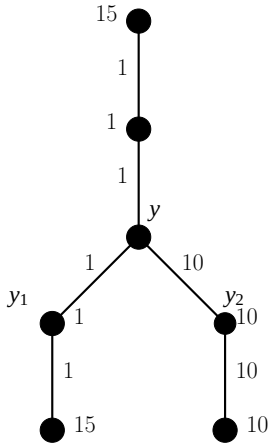


FIG. 7.6 – Exemple d'arbre pondéré trompant l'algorithme de [BFFS02]

7.3 Contributions

Dans cette section, je présente les résultats que j'ai obtenus sur le process number, la pathwidth et l'indice d'échappement arête connexe. La Section 7.3.1 est dédiée à un algorithme permettant de calculer le process number d'un arbre ainsi que d'autres invariants comme la pathwidth et l'indice d'échappement sommet notamment. Ce travail a été fait en collaboration avec D. Coudert et D. Mazauric. Les avancées sur les relations entre la pathwidth d'un graphe planaire extérieur et celle de son dual ont été faites avec D. Coudert et J-S. Serreni ([CHS07]), elles sont présentées dans la Section 7.3.2. Dans la Section 7.3.3, je présente un travail réalisé avec O. Amini et S. Perénes ([AHP]) concernant la pathwidth des graphes planaires. Enfin, dans la dernière section, je présente de nouveaux résultats concernant l'indice d'échappement arête connexe d'un arbre pondéré.

7.3.1 Process number d'un arbre

Nous présentons ici un résultat exposé lors de la conférence Algotel'08 et sous forme de "brief announcement" à DISC 08. Ce résultat a été obtenu avec D. Coudert et D. Mazauric [CHM08a]. Il s'agit d'un algorithme de complexité $O(n\log(n))$ permettant de calculer le process number d'un arbre. De plus, cet algorithme peut être étendu au calcul d'autres paramètres comme l'indice d'échappement sommet et l'indice d'échappement arête. L'algorithme étendu au calcul de l'indice d'échappement sommet ne donne pas un algorithme de complexité minimale puisqu'il existe un algorithme linéaire pour le calculer, celui de P. Scheffler [Sch90]. Cependant, pour atteindre une complexité linéaire, l'algorithme est centralisé et la version distribuée de l'algorithme présentée dans [Sch90] entraîne plus de transmissions de données que l'algorithme que je décris ici.

Dans cette section, mon objectif est de donner le principe de l'algorithme. Une description rigoureuse peut être trouvée dans le rapport de recherche [CHM08b], présent

dans l'Annexe E.1. Ces résultats ont été présentés à Algotel [CHM08a].

Principe de l'algorithme. Notre algorithme est basé sur le Lemme 7.2, et le principe est de détecter les nœuds auxquels se rejoignent trois sous-arbres ou plus, ayant le même process number. C'est un algorithme dynamique qui s'initialise aux feuilles. Ces dernières ne nécessitant aucun agent pour être traitées, elles s'initialisent à 0. Cela signifie que le process number d'un sous-arbre formé par une feuille a process number 0. Ensuite, les feuilles transfèrent leur valeur (0 donc) à leur père.

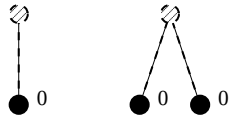


FIG. 7.7 – Initialisation de l'algorithme.

À chaque étape, un nœud v qui a reçu un message de tous ses voisins sauf un $\{v_1,\ldots,v_k\}$ évalue le nombre d'agents nécessaires pour traiter l'arbre formé par lui même et les sous-arbres enracinés aux voisins $\{v_1,\ldots,v_k\}$ qui lui ont envoyé un message. On note $T_v = \{v\} \cup_1^k T_{v_i}$ le sous-arbre enraciné en v dont l'algorithme calcule le process number (c.f. définition de T_v avant le Lemme 7.9).

Si un nœud v a tous ses voisins sauf un qui sont des feuilles, le sous-arbre T_v est une étoile. Or, une étoile a process number un : il suffit de placer un agent sur le nœud central. Ainsi, lors de l'éxécution de l'algorithme sur ce nœud v, v décide qu'il faut un agent pour traiter le sous-arbre T_v (Figure 7.8).

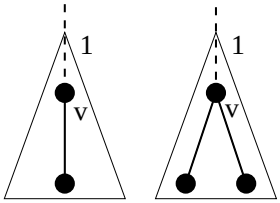


FIG. 7.8 – Une étape de l'algorithme.

Ensuite, un nœud qui reçoit des valeurs 1 et 0 de ses voisins, va déduire que le sous-arbre constitué à process number 1 ou 2 selon des cas particuliers qui constituent l'initialisation de l'algorithme. Ainsi dans l'exemple de la Figure 7.9, T_v a process number 2.

Par la suite, les arbres formés sont plus généraux et le nombre d'agents nécessaires varie mais le principe reste le même. Un nœud v va calculer le process number de T_v et va envoyer le résultat au voisin qui ne lui a pas envoyé de message. Pour ce faire, l'algorithme se base sur le Lemme 7.2. On note p la plus grande valeur que v reçoit. Si v reçoit p une ou deux fois, $pn(T_v)$ vaut p, sinon $pn(T_v)$ vaut $p+1$. Dans l'exemple de

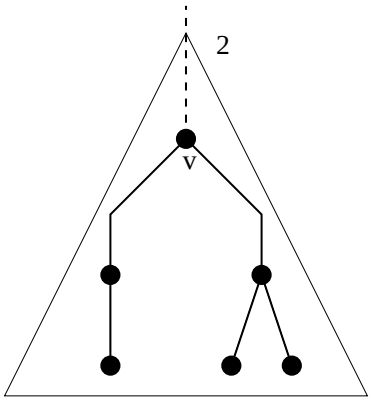


FIG. 7.9 – Le process number de T_v vaut 2.

la Figure 7.10, le nœud v_0 reçoit deux fois la valeur 2 et le process number de T_{v_0} vaut bien 2.

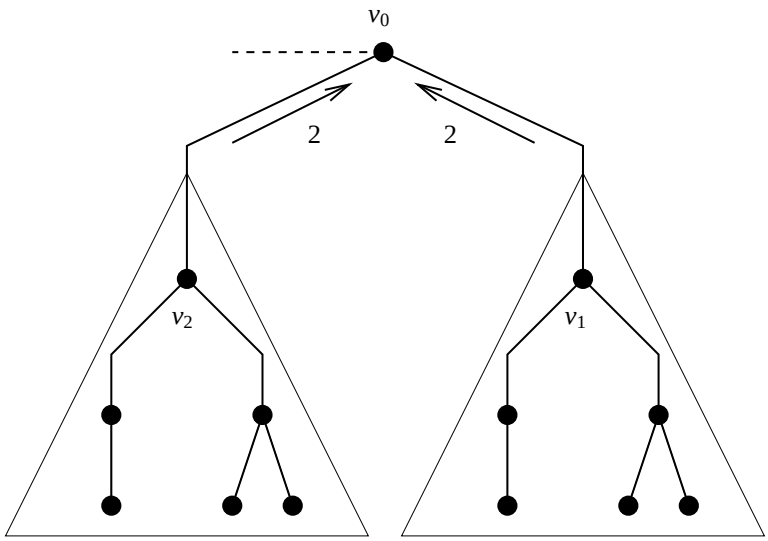


FIG. 7.10 – Le process number de T_{v_0} vaut 2.

Je continue à décrire le déroulement de cet algorithme sur l'arbre T de la Figure 7.11.

Nous avons vu que les nœuds v_0 et v_3 calculent que $pn(T_{v_0})$ et $pn(T_{v_3})$ valent tous les deux 2 (Figures 7.8 et 7.10). v_0 et v_3 envoient donc tous les deux la valeur 2 à u qui calcule T_u. Selon les règles précédentes, comme u reçoit deux fois la valeur 2, il déduit que $pn(T_u)$ (et donc $pn(T)$) vaut 2, or c'est faux : $pn(T) = 3$.

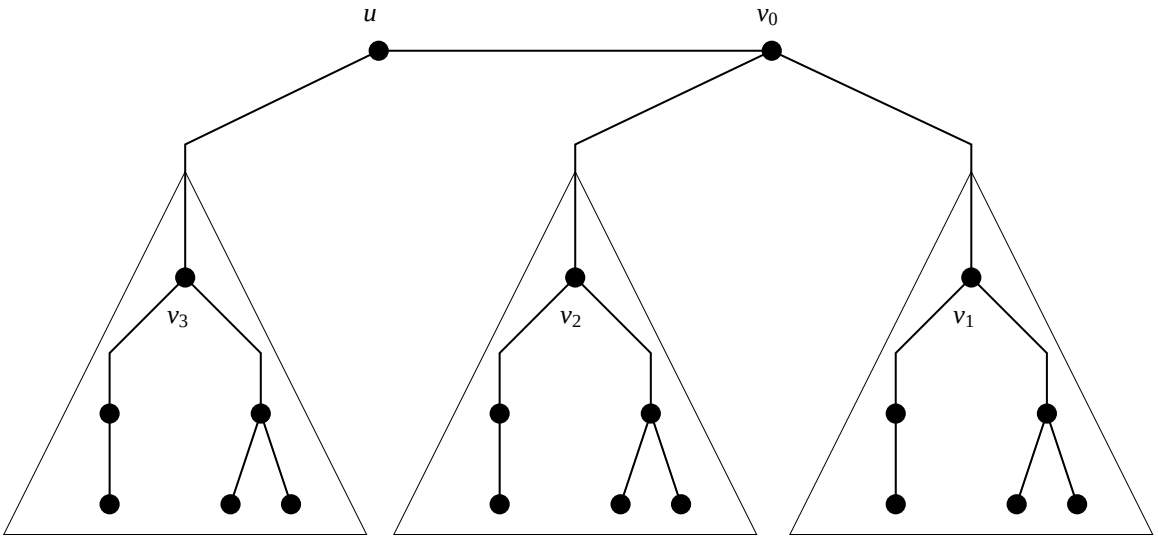


FIG. 7.11 – Le process number de T vaut 3 et non 2.

Origine de l'erreur. L'erreur vient du fait que le nœud u n'a pas détecté que v_0 a trois voisins qui sont racines de sous-arbres de process number 2 : T_{v_1} et T_{v_2} dont il a reçu le process number et T_u dont le process number n'avait pas encore été calculé par l'algorithme au moment où v_0 a fait son calcul.

Pour éviter cette erreur, étant donné un nœud v qui reçoit des informations de tous ses voisins sauf un, on distingue trois cas qui peuvent se produire :

1. parmi les voisins $\{v_1,\ldots,v_k\}$ dont v reçoit un message, seul l'un d'eux est racine d'un sous-arbre de process number maximal : $p = pn(T_{v_1}) > pn(T_{v_i}), 1 < i \leq k$;
2. parmi les voisins $\{v_1,\ldots,v_k\}$ dont v reçoit un message, deux sont racines d'un sous-arbre de process number maximal : $p = pn(T_{v_1}) = pn(T_{v_2}) > pn(T_{v_i}), 2 < i \leq k$;
3. parmi les voisins $\{v_1,\ldots,v_k\}$ dont v reçoit un message, au moins trois sont racines d'un sous-arbre de process number maximal : $p = pn(T_{v_1}) = pn(T_{v_2}) = pn(T_{v_3}) \geq pn(T_{v_i}), 3 < i \leq k$.

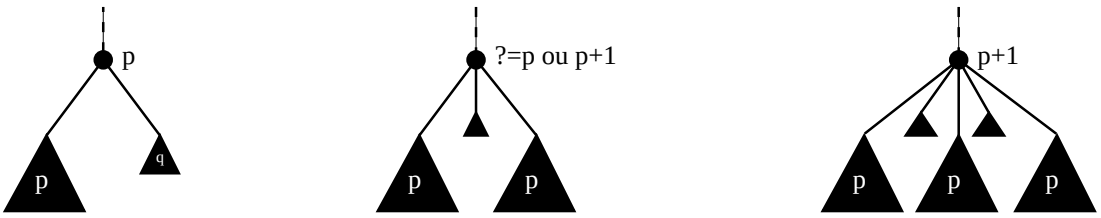


FIG. 7.12 – Trois cas de figure possibles.

Dans le premier cas, d'après le Lemme 7.2, on sait qu'il y a deux sous-arbres qui sont déterminants : le sous-arbre qui n'a pas encore été exploré (qui est celui qui correspond au voisin n'ayant pas envoyé de message) et le sous-arbre enraciné en v_1 (qui est celui avec le plus grand process number (p) parmi ceux qui ont été explorés par l'algorithme). On appelle T_{v_1} le **sous-arbre dominant**. On sait que le process number de tout l'arbre T sera le maximum entre le process number de ces deux sous-arbres. Ainsi le nœud courant v peut envoyer au voisin, dont il n'a pas reçu de message, le process number du sous-arbre dominant : p.

Dans le troisième cas, on sait d'après le Lemme 7.2 que l'arbre T_v a process number $p+1$. Comme pour le cas précédent, le process number de l'arbre T va être le maximun entre le process number de T_v et le process number du sous-arbre qui n'a pas encore été exploré. Le nœud courant v peut donc envoyer $p+1$ au voisin dont il n'a pas reçu de message.

Dans le deuxième cas, c'est un peu plus complexe. En effet, on ne sait pas si deux ou trois sous-arbres ayant le même process number p se rejoignent en v car cela dépend du sous-arbre qui n'a pas encore été exploré. Le process number de l'arbre T peut valoir p (si le sous-arbre qui n'a pas été exploré a un process number plus petit), $p+1$ (si le sous-arbre qui n'a pas été exploré a un process number p) ou le process number du sous-arbre qui n'a pas été exploré s'il est plus grand que p. Pour différencier les trois cas, il est nécessaire d'envoyer plus d'informations que seulement p : il faut signaler que deux branches de process number p se rejoignent en v.

Un nouvel algorithme. La solution que nous avons adoptée revient à envoyer le message "p ou $p+1$?" et à commencer un nouveau compteur à partir du voisin qui n'a pas encore été exploré.

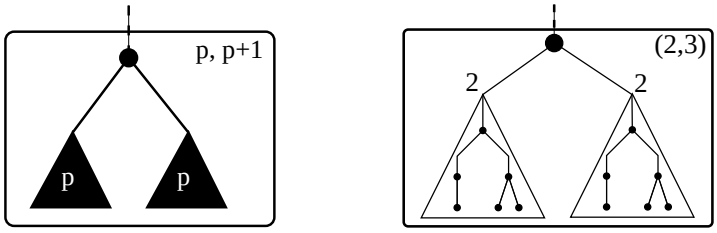


FIG. 7.13 – "p ou $p+1$?"

Seulement, ce problème peut se poser de manière récursive et on peut être amené à avoir plusieurs compteurs en parallèle. Les compteurs indiquent des embranchements de deux types : **instables** s'ils sont issus d'un nœud étant dans le deuxième cas (ils sont alors du type "p ou $p+1$?"), et **stable** s'ils sont issus d'un nœud dans le premier ou le troisième cas (ils sont alors du type p).

Les nœuds instables délimitent des sous-arbres et induisent une décomposition de notre arbre initial. Les sous-arbres enracinés à un nœud instable sont représentés par des rectangles et appelés instables. Les sous-arbres enracinés à un nœud stable sont

représentés par des triangles et appelés stables. Dans [CHM08b], nous montrons qu'il est possible d'avoir une décomposition dans laquelle il y a au plus un sous-arbre stable. L'ensemble des compteurs est stocké sous la forme d'un tableau et d'un entier pour l'éventuel sous-arbre stable. Les Figures 7.14, 7.15 et 7.16 donnent trois exemples différents. Dans notre algorithme, chaque nœud transmet donc un tableau contenant l'ensemble des compteurs. Ces données représentent également la décomposition de T_v en sous-arbres.

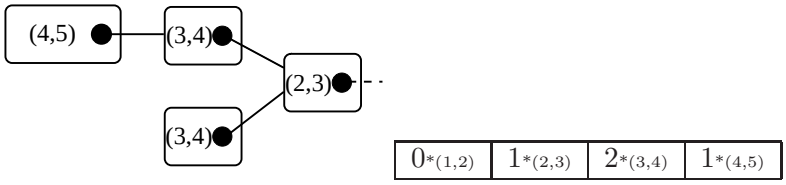


FIG. 7.14 – Premier exemple de décomposition induite par l'algorithme.

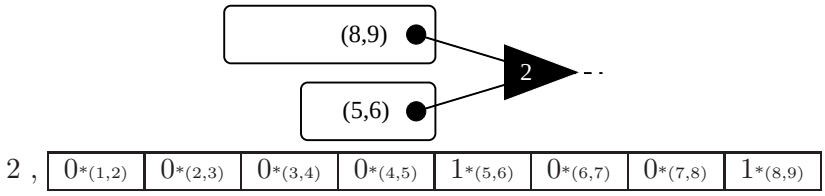


FIG. 7.15 – Deuxième exemple de décomposition induite par l'algorithme.

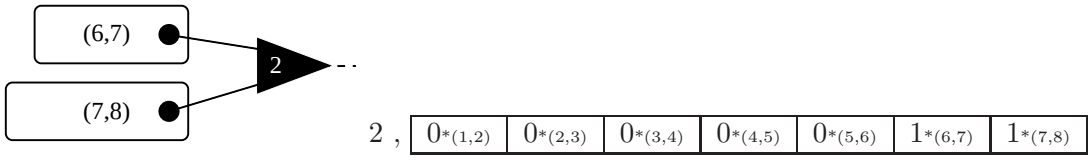


FIG. 7.16 – Troisième exemple de décomposition induite par l'algorithme.

Dans [CHM08b], nous montrons le théorème suivant qui permet de calculer le process number d'un arbre étant donné une décomposition calculée par l'algorithme :

Théorème 7.10 *Etant donné un arbre T et un nœud v, le process number du sous-arbre T_v est égal à la longueur du tableau ℓ calculé par l'algorithme en v si, dans le tableau, la dernière série de 1 est précédée d'un 0. Il vaut $\ell+1$ sinon.*

Pour les trois exemples des Figures 7.14, 7.15 et 7.16 le Théorème 7.10 donne :

Figure 7.14 : $0_{*(1,2)}$ | $1_{*(2,3)}$ | $2_{*(3,4)}$ | $1_{*(4,5)}$
Le process number de l'arbre est 5.

Figure 7.15 : 2 , | $0^{*(1,2)}$ | $0^{*(2,3)}$ | $0^{*(3,4)}$ | $0^{*(4,5)}$ | $1^{*(5,6)}$ | $0^{*(6,7)}$ | $0^{*(7,8)}$ | $1^{*(8,9)}$ |
Le process number de l'arbre est 8.

Figure 7.16 : 2 , | $0^{*(1,2)}$ | $0^{*(2,3)}$ | $0^{*(3,4)}$ | $0^{*(4,5)}$ | $0^{*(5,6)}$ | $1^{*(6,7)}$ | $1^{*(7,8)}$ |
Le process number de l'arbre est 7.

Après avoir décrit les données transmises par chaque nœud, je vais maintenant décrire comment ces données sont traitées. Un nœud v qui reçoit un message de chacun de ses voisins sauf un va :

- traiter les entiers correspondants aux éventuels sous-arbres stables selon les trois cas possibles.
- faire la somme des tableaux reçus case par case.

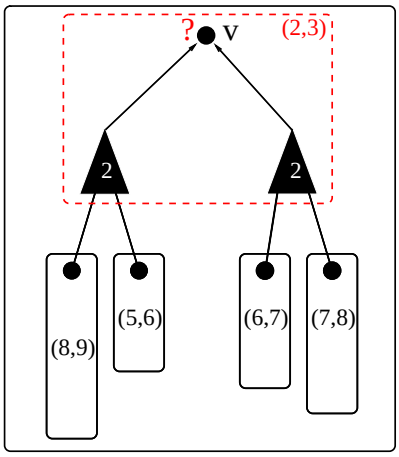


FIG. 7.17 – Premier exemple de calcul à un nœud.

Pour cet exemple, l'algorithme donne :

2 , | $0^{*(1,2)}$ | $0^{*(2,3)}$ | $0^{*(3,4)}$ | $0^{*(4,5)}$ | $1^{*(5,6)}$ | $0^{*(6,7)}$ | $0^{*(7,8)}$ | $1^{*(8,9)}$ |

+

2 , | $0^{*(1,2)}$ | $0^{*(2,3)}$ | $0^{*(3,4)}$ | $0^{*(4,5)}$ | $0^{*(5,6)}$ | $1^{*(6,7)}$ | $1^{*(7,8)}$ |

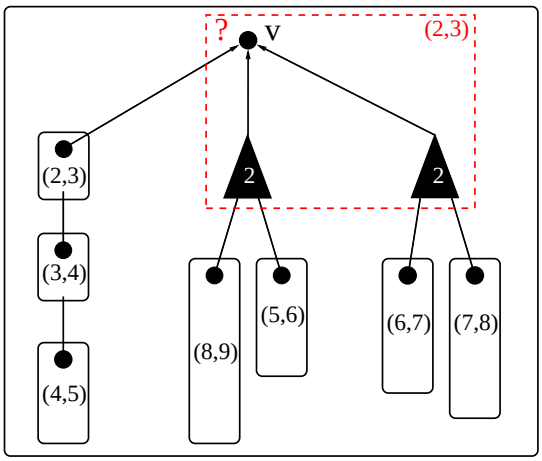
=

$T_{7.18}$: | $0^{*(1,2)}$ | $1^{*(2,3)}$ | $0^{*(3,4)}$ | $0^{*(4,5)}$ | $1^{*(5,6)}$ | $1^{*(6,7)}$ | $1^{*(7,8)}$ | $1^{*(8,9)}$ |

FIG. 7.18 – Premier exemple de calcul à un nœud.

Le processus est illustré par les Figures 7.18 et 7.19. Le Tableau $T_{7.18}$ décrit la décomposition du sous-arbre enraciné en v de la Figure 7.18 qui, d'après le Théorème 7.10 a process number 8.

Le Tableau $T_{7.19}$ décrit la décomposition du sous-arbre enraciné en v de la Figure 7.19. Le Théorème 7.10 dit qu'il a process number 8.



Pour cet exemple, l'algorithme donne :

	$0^{*(1,2)}$	$1^{*(2,3)}$	$1^{*(3,4)}$	$1^{*(4,5)}$				
+								
2 ,	$0^{*(1,2)}$	$0^{*(2,3)}$	$0^{*(3,4)}$	$0^{*(4,5)}$	$1^{*(5,6)}$	$0^{*(6,7)}$	$0^{*(7,8)}$	$1^{*(8,9)}$
+								
2 ,	$0^{*(1,2)}$	$0^{*(2,3)}$	$0^{*(3,4)}$	$0^{*(4,5)}$	$0^{*(5,6)}$	$1^{*(6,7)}$	$1^{*(7,8)}$	
=								
$T_{7.19}$:	$0^{*(1,2)}$	$2^{*(2,3)}$	$1^{*(3,4)}$	$1^{*(4,5)}$	$1^{*(5,6)}$	$1^{*(6,7)}$	$1^{*(7,8)}$	$1^{*(8,9)}$

FIG. 7.19 – Deuxième exemple de calcul à un nœud.

Simplification. Si dans un tableau t calculé, il y a une série de cases : de la case k à la case l, remplies de 1 : $\forall i, k \leq i \leq l, t[i] = 1$ et que la case $k-1$ ne contient pas un 0, alors on peut simplifier la décomposition. Étant donné une série de telles cases telle que l soit maximum, toutes les cases d'indice inférieure ou égale à l sont mises à 0 et l'entier décrivant un embranchement stable est mis à $l+1$. Selon ce principe, le tableau $T_{7.19}$ correspondant à la Figure 7.19 est simplifié à :

9 ,	$0^{*(1,2)}$	$0^{*(2,3)}$	$0^{*(3,4)}$	$0^{*(4,5)}$	$0^{*(5,6)}$	$0^{*(6,7)}$	$0^{*(7,8)}$	$0^{*(8,9)}$	$0^{*(9,10)}$

Cette simplification est justifiée par le Théorème 7.10. Il est montré dans [CHM08b] que ces deux décompositions sont équivalentes. La dernière étant plus simple, notre algorithme la favorise.

Complexité. L'algorithme effectue n étapes (une par nœud) et à chaque fois, il transmet un entier et un tableau de bits. Le nombre d'opérations dépend linéairement de la taille du tableau. La longueur du tableau est au plus le process number qui est borné par $\log_3(n)$. Ainsi le nombre total de bits transmis par l'algorithme est $O(n\log(n))$ tout comme le nombre d'opérations.

Notre algorithme est naturellement distribué et il peut être utilisé pour mettre à jour le process number des composantes d'une forêt lors de l'ajout et de la suppression d'arêtes. Enfin, si l'on code chaque case du tableau transmis par un nœud en utilisant des bits supplémentaires pour indiquer la dernière case, il n'est pas nécessaire de connaître le nombre total de sommets de la forêt.

Les détails concernant ces extensions sont présentés dans [CHM08b]. Nous y montrons également qu'étant donné $k \in \mathcal{N}$, aucun algorithme distribué, qui vérifie qu'un sommet v demande des messages à ses voisins qu'une fois et simultanément, induit, pour certains arbres, un trafic d'au moins $\frac{k-1}{k}n(\log_3 n)$ bits. Ce résultat signifie en particulier que notre algorithme est optimal.

L'algorithme proposé dans [Sch90] dans sa version distribuée peut induire un trafic global de $O(n\log(n)\log(\log(n)))$ bits.

Travail futur. Une question intéressante est de savoir si cet algorithme peut se généraliser à d'autres classes de graphes, notamment aux graphes planaires extérieurs :

Question 7.11 *Existe-t-il un algorithme linéaire permettant de calculer le process number et l'indice d'échappement sommet d'un graphe planaire extérieur ?*

Une autre question intéressante est de savoir si le process number et la pathwidth sont équivalents. En effet, on sait qu'ils diffèrent d'au plus un mais on ne sait pas caractériser en temps polynomial les graphes pour lesquels ces deux paramètres sont égaux.

Question 7.12 *Existe-t-il un algorithme polynomial permettant de caractériser les graphes pour lesquels l'indice d'échappement sommet et le process number sont équivalents ?*

De manière plus générale la question suivante concerne les différents paramètres de recherche que l'on peut définir sur un graphe tels l'indice d'échappement sommet, l'indice d'échappement arête connexe et le process number entre autres.

Question 7.13 *Existe-t-il une classe de graphe pour laquelle calculer un paramètre (indice d'échappement sommet, indice d'échappement arête, process number ...) est polynomial alors que calculer un autre paramètre est NP-dur ?*

7.3.2 Pathwidth des graphes planaires extérieurs

Dans cette section, je résume les résultats que j'ai obtenus concernant la pathwidth des graphes planaires avec D. Coudert et J-S. Séréni. Les résultats ont été publiés dans

Journal of Graph Theory [CHS07], l'article est présent dans l'Annexe E.2. Les définitions de graphe planaire, graphe planaire extérieur, dual et dual faible sont rappelées dans la Section 2.1.

Dans [BF02], H. Bodlaender et F. Fomin ont montré que la pathwidth d'un graphe planaire extérieur biconnexe est au plus deux fois la pathwidth de son dual plus deux. Ils ont ensuite conjecturé l'existence d'une constante c telle que pour tout graphe planaire biconnexe, sa pathwidth est au plus la pathwidth de son dual plus c. Ils ont également conjecturé que c'est vrai pour $c = 1$. Il est intéressant de noter que cette conjecture a pu être motivée par le théorème de D. Lapoire sur la treewidth [Lap99] :

Théorème 7.14 ([Lap99]) *Pour tout graphe planaire G, $tw(G) \leq tw(G^*)$.*

Dans [Fom03], F. Fomin montre que la plus forte des conjectures est vraie pour toute triangulation planaire, mais dans [CHS07], nous montrons que les deux conjectures sont fausses dans le cas général en proposant une famille de graphes planaires extérieurs biconnexes de pathwidth $2p+1$ et dont le dual a pathwidth $p+2$. On montre ensuite de manière algorithmique que cet écart est le plus grand que l'on puisse obtenir. On donne ainsi un algorithme de 2-approximation pour calculer la pathwidth des graphes planaires extérieurs biconnexes. Ces deux résultats sont résumés par les Théorèmes 7.15 et 7.16.

Théorème 7.15 *Pour tout entier $p \geq 1$ et tout entier $k \in \{1, 2, \ldots, p+1\}$, il existe un graphe planaire extérieur biconnexe de pathwidth $p+k$ dont le dual a pathwidth $p+1$.*

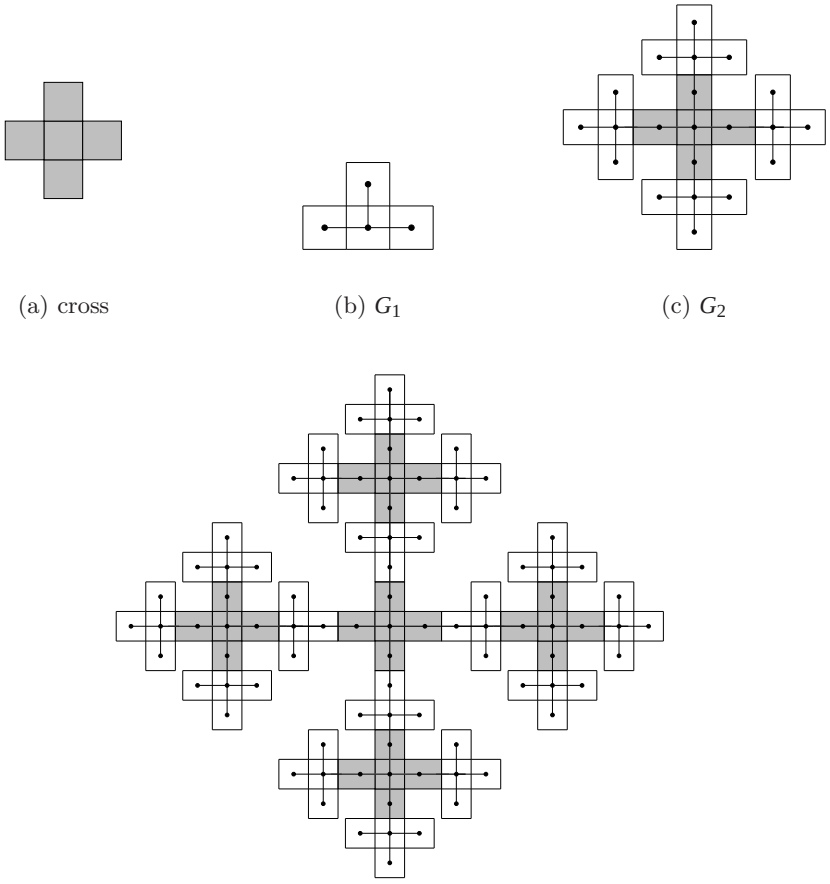
Pour prouver ce théorème on utilise une famille de graphes construite récursivement à l'aide d'une croix (Figure 7.20(a)). La construction est illustrée par les Figures 7.20(b), 7.20(c) et 7.20(d).

Dans [CHS07], nous prouvons que la pathwidth de ces graphes augmente de deux à chaque fois ($pw(G_{i+1}) = pw(G_i) + 2$) alors que celle de leur dual n'augmente que d'un ($pw(G^*_{i+1}) = pw(G^*_i) + 2$). Notre preuve utilise fortement la structure de ces graphes pour montrer que leur pathwidth augmente de deux à chaque étapes. Il serait intéressant de trouver un critère général (tel que celui pour les arbres : Lemme 7.5) décrivant les constructions qui forcent la pathwidth d'un graphe planaire extérieur à augmenter de deux. Un tel critère pourrait être utilisé pour trouver un algorithme exact et efficace pour le calcul de la pathwidth d'un graphe planaire extérieur (Question 7.11).

Le théorème suivant prouve que la famille de graphe que l'on propose maximise l'écart entre la pathwidth du graphe et celle du dual :

Théorème 7.16 *Soit G un graphe planaire extérieur biconnexe sans boucles ni arêtes multiples. On a $pw(G^*) \leq pw(G) \leq 2pw(G^*) - 1$.*

Afin de prouver ce théorème, on a développé un algorithme qui calcule une décomposition en chemins d'un graphe planaire extérieur à partir d'une décomposition en chemins de son dual et on prouve que la largeur de la décomposition obtenue ne peut pas être plus que le double de celle d'origine.



(a) cross (b) G_1 (c) G_2

(d) G_3

FIG. 7.20 – Exemples de graphes satisfaisant $pw(G) = 2pw(G^*) - 1$

On donne également un algorithme exact pour calculer la pathwidth des graphes planaires extérieurs dont le dual faible est un caterpillar.

L'article se termine par le problème suivant qui a été posé comme conjecture par F. Fomin et D. Thilikos dans [FT07] :

Conjecture 7.17 *Y a-t-il une constante c telle que, pour tout graphe planaire 2-connexe G,*

$$\frac{1}{2}pw(G^*) - c \leq pw(G) \leq 2pw(G^*) + c?$$

7.3.3 Pathwidth des graphes planaires

Avec O. Amini et S. Pérennes, nous nous sommes intéressés à la question terminant la section précédente. Les résultats que l'on a obtenus doivent paraître dans SIAM

Journal of Discrete Maths [AHP] et je les présente ici. L'article figure dans l'Annexe E.3.

Dans [FT07], les auteurs prouvent une version plus faible de la Conjecture 7.17. Ils montrent qu'elle est vraie si l'on remplace les facteurs 1/2 et 2 respectivement par 1/6 et 6. Dans [AHP] nous montrons qu'elle est vraie avec les facteurs 1/3 et 3 et qu'elle est également vraie si on se restreint aux graphes planaires quatre connexes.

Théorème 7.18 *Pour tout graphe planaire* 3*-connexe* G*, on a* $\frac{1}{3}pw(G^*)-2 \leq pw(G) \leq 3pw(G^*)+2$.

Théorème 7.19 *Si* G *est un graphe planaire ayant un chemin Hamiltonien, alors* $\frac{1}{2}pw(G)-1 \leq pw(G^*) \leq 2pw(G)+1$. *En particulier cela est vrai si* G *est* 4*-connexe.*

Pour prouver ce théorème nous proposons, comme pour le Théorème 7.16, un algorithme qui transforme une décomposition en chemins du dual d'un graphe planaire en une décomposition en chemins du graphe lui même sans en faire plus que tripler la taille. La technique que l'on propose ne permet malheureusement pas de résoudre la Conjecture 7.17 car il existe des graphes 3-connexes pour lesquels la taille de la décomposition en chemins obtenue par notre algorithme est le triple de la taille de celle de leur dual. Ce théorème englobe presque le Théorème 7.16.

Enfin, d'après une remarque de D. Coudert, un énoncé plus fort proposé dans [FT07] est faux :

Question 7.20 *[FT07] Il existe une constante* c *telle que pour tout graphe planaire 2-connexe* G *de treewidth au moins* m*, on a* $pw(G^*) \leq \frac{m}{m-1}pw(G)+c$.

En effet, si on modifie légèrement la famille de graphes (illustrée par les Figures 7.20(b) à 7.20(d)) de la section précédente en collant sur une face du graphe G_i une grille $i \times i$ (cf Figures 7.21(a) et 7.21(b)) sa pathwidth ne change pas alors que sa treewidth passe de 2 à i.

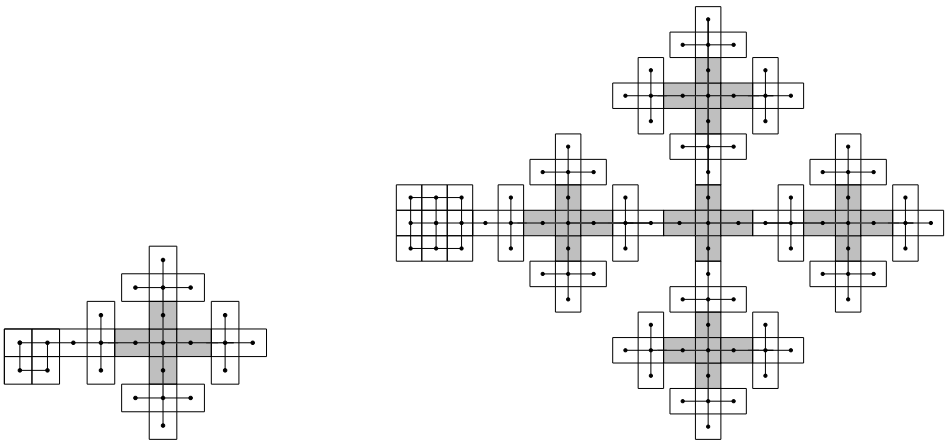
7.3.4 A propos de l'indice d'échappement arête connexe d'un arbre

7.3.4.1 Arbres pondérés et arbres de cliques

D'après un résultat de H. Bodlaender et T. Kloks, on sait que trouver la pathwidth d'un graphe de treewidth borné est polynomial [BK96] (le degré du polynôme dépend de la borne sur la treewidth). Cet algorithme a surtout un intérêt théorique dans la mesure où sa complexité est très élevée. Dans le cas des graphes planaires extérieurs, elle est supérieure à n^{11}, alors que la treewidth vaut 2.

Si un tel résultat est vrai pour l'indice d'échappement arête connexe d'un graphe, calculer l'indice d'échappement arête connexe d'un arbre pondéré serait également polynomial en la somme des poids. En effet, on peut transformer un arbre pondéré en un arbre de cliques ayant le même indice d'échappement arête connexe.

Pour cela il faut d'abord remarquer que si un sommet a un poids inférieur au poids d'une des arêtes qui lui sont incidentes, on peut augmenter son poids sans augmenter l'indice d'échappement arête connexe.



(a) G_2 modifié (b) G_3 modifié

FIG. 7.21 – Les deux premiers exemples modifiés contredisant la Conjecture 7.20

Ensuite, on peut transformer un arbre pondéré (T, w) en un arbre de cliques comme suit : chaque sommet v est remplacé par une clique avec des arêtes doubles C_u dont la taille est le poids du dit sommet v moins un : $w(v) - 1$ (un sommet de poids 2 est remplacé par un sommet avec une boucle). Le poids d'une arête uv est l'arête connectivité entre les deux cliques C_u et C_v remplaçant les sommets u et v. Si un sommet u est relié à deux sommets v_1 et v_2, la transformation est faite de telle sorte à maximiser $\Gamma(C_{v_1}) \cap C_u \cap \Gamma(C_{v_2})$. Cette transformation est illustrée par la Figure 7.22.

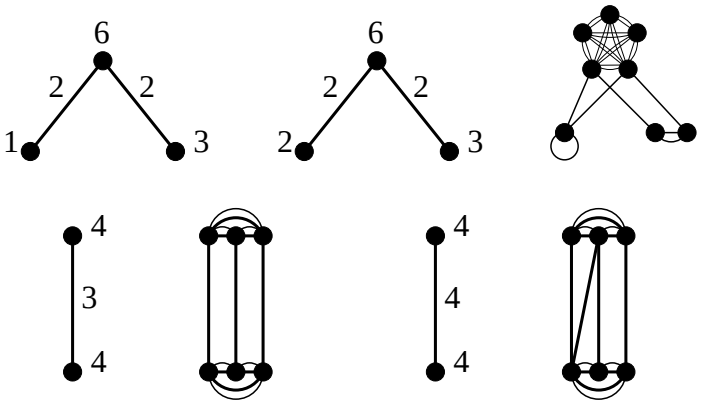


FIG. 7.22 – D'un arbre pondéré à un arbre de cliques.

Remarque 7.21 Le rôle des arêtes doubles dans une clique de taille $k-1$ est de forcer l'utilisation de k agents pour la nettoyer.

Il est facile de voir qu'une stratégie arête connexe sur l'arbre pondéré donne une stratégie arête connexe sur l'arbre de clique correspondant. La Remarque 7.21 permet quant à elle de prouver qu'une stratégie arête connexe dans l'arbre de cliques donne une stratégie arête connexe dans l'arbre pondéré.

Le graphe obtenu a une treewidth bornée par $2w_{\max}$ où $w_{\max}$ est le poids maximum d'un sommet.

Dans la Section 7.3.4.2 je montre qu'il n'est pas possible de le calculer en temps linéaire.

7.3.4.2 Borne inférieure sur la complexité

L'arbre pondéré de la Figure 7.23 montre que l'on ne peut pas trouver une stratégie de capture arête connexe en temps linéaire en n (bien que l'on connaisse l'indice d'échappement connexe), ceci ne dit par contre pas qu'il n'existe pas d'algorithme linéaire en la somme des poids.

Étant donné un entier p assez grand, les $a_1, \ldots, a_k$ représentent k nombres tirés au hasard de somme au plus $p/2-1$. L'indice d'échappement arête connexe est au moins $3p/2+\sum_i^k a_k$. Trouver une stratégie de capture arête connexe optimale nécessite de trier les fils du sommet central dans l'ordre décroissant de leur poids.

La stratégie consiste à d'abord visiter le sommet de poids $3p/2+\sum_{i=1}^k a_i$, puis le sommet central et enfin de placer les agents sur les sommets de poids a_i. Pour ce faire, il faut commencer par le plus gros voisin du sommet central et aller en décroissant, sinon on utilise plus de $3p/2+\sum_i^k a_k$ agents. Une fois les agents ainsi placés, on peut finir de traiter les feuilles.

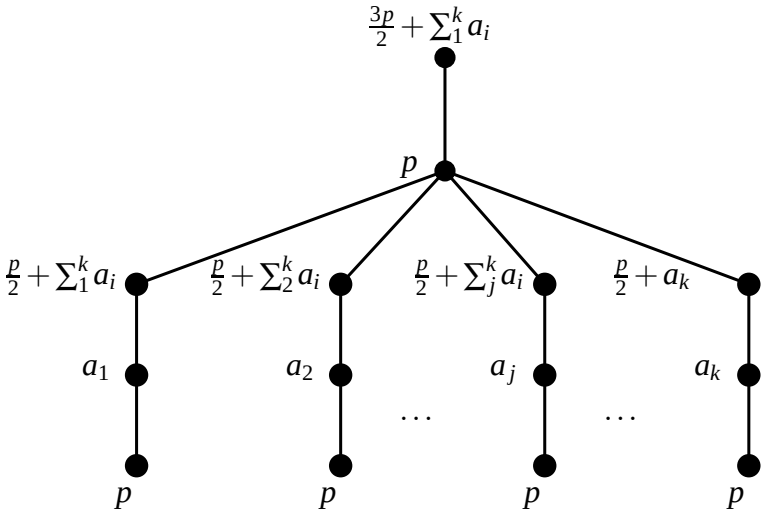


FIG. 7.23 – Un arbre dur à explorer.

7.3.4.3 Version dynamique de l'indice d'échappement arête connexe non pondéré

Dans [BFFS02], L. Barrière, P. Flocchini, P. Fraigniaud, N. Santoro définissent une version dynamique de l'indice d'échappement arête connexe et demandent s'il existe un algorithme polynomial pour le calculer dans les arbres. Dans cette section, je montre que leur algorithme peut être adapté pour le calculer, sa complexité devient alors : $O(n\log(n))$. L'**indice d'échappement arête connexe dynamique** (dynamic contiguous search number) est le nombre minimal d'agents nécessaires pour capturer un fugitif tel qu'à chaque tour des agents, l'un d'eux peut effectuer l'une des deux actions suivantes :

- se positionner sur un sommet s'il n'est pas déjà positionné sur un autre sommet
- se déplacer d'un sommet vers un sommet adjacent

tout en respectant la contrainte que la zone sûre reste connexe.

A son tour, le fugitif peut se déplacer n'importe où dans le graphe tant qu'il ne passe pas par un sommet v sur lequel est positionné au moins autant d'agents qu'il y a de sommets contaminés adjacents à v.

Dans ce cadre là, on peut montrer le lemme suivant :

Lemme 7.22 *Étant donné un arbre T de racine y, si $y_1,\ldots,y_p$ sont les fils de y tels que $cs(T_{y_i}) \geq cs(T_{y_{i+1}})$ pour $1 \leq i < p$, alors $cs(T) = \max_i(cs(T_{y_i}) + i - 1)$.*

Preuve. Comme on est dans le cas non pondéré, une stratégie qui traite partiellement un sous-arbre T_{y_i} avant de traiter un autre sous-arbre T_{y_j} peut être transformée en une stratégie utilisant autant d'agents et qui traite complètement T_{y_i} avant de commencer à traiter T_{y_j}. Il est alors clair qu'une stratégie va utiliser au moins $\max_i(cs(T_{y_i}) + i - 1)$ agents et qu'une stratégie qui traite les sous arbres dans l'ordre T_{y_p} à T_{y_1} utilise ce nombre d'agents. □

Théorème 7.23 *L'indice d'échappement arête connexe dynamique d'un arbre T peut être calculé de manière distribuée par un algorithme de complexité $O(n\log(n))$.*

Dans l'algorithme proposé dans [BFFS02], remplacer l'utilisation du Lemme 7.9 par l'utilisation du Lemme 7.22 nécessite de trier les voisins de chaque sommet au lieu de sélectionner les deux sommets qui sont racines des sous arbres ayant le plus grand indice d'échappement arête connexe. A chaque sommet v, le nombre d'opérations effectuées est alors $d(v)\log(d(v))$ au lieu de $2d(v)$. La complexité globale de l'algorithme est donc bien $O(n\log(n))$ comme annoncé.

7.4 Conclusion et Perspectives

Dans ce chapitre, je me suis intéressé au problème du reroutage, qui est nécessaire dans les réseaux WDM dynamiques. Ce problème peut être modélisé par un problème de traitement des sommets d'un graphe orienté des conflits. J'ai montré les liens entre

ce problème et divers problèmes dont le calcul de la pathwidth. Ensuite j'ai expliqué le fonctionnement d'un algorithme permettant de calculer le process number d'un arbre. Dans la section suivante, j'ai présenté de nouveaux résultats sur les liens entre la pathwidth d'un graphe planaire extérieur et celle de son dual ainsi que sur les liens entre la pathwidth d'un graphe planaire et celle de son dual. Dans la dernière section, je présente une borne inférieure sur la complexité d'un algorithme calculant l'indice d'échappement arête connexe. Je présente également un lemme permettant d'adapter un algorithme qui calcule l'indice d'échappement arête connexe pour calculer l'indice d'échappement arête connexe dynamique.

Il reste de nombreuses questions ouvertes. Par exemple, la conjecture sur les liens entre la pathwidth d'un graphe planaire et celle de son dual n'est pas encore complètement résolue. Il n'existe toujours pas d'algorithme efficace permettant le calcul de la pathwidth d'un graphe planaire extérieur. De manière plus générale, il serait intéressant de pouvoir calculer ou tout du moins approximer de manière efficace le process number des graphes de conflits apparaissant lors des reroutages de connexions. Pour citer un dernier problème, les liens entre les complexités du calcul de différents paramètres restent encore flous.

Chapitre 8

Conclusion

Comprendre et développer des outils combinatoires et algorithmiques pour la conception de réseau et leur administration a été le sujet essentiel de ma thèse. J'ai utilisé des concepts généraux permettant de résoudre divers problèmes apparaissant tout le long du cycle de vie des réseaux. Les outils que j'ai utilisés m'ont permis d'avoir des approches différentes et complémentaires. Mon travail n'est pas spécifique à un seul type de réseaux, au contraire j'en ai considéré plusieurs types dont des réseaux optiques de cœur ou d'accès, des réseaux d'antennes, des réseaux d'accès WiFi, des réseaux en forme de grilles, ou bien encore des réseaux embarqués.

La diversité des réseaux jointe à celle des outils a permis de dégager plusieurs problématiques et d'en traiter un certain nombre. De nombreux problèmes permettant des avancées dans les problématiques que j'ai dégagées peuvent être étudiées à la suite de cette thèse. Je résume ci-dessous mes contributions et quelque questions ouvertes.

- Concernant les réseaux embarqués, le problème, posé par Alcatel Space Industrie, est de concevoir des réseaux capables de router p signaux arrivant par p entrées parmi $p+\lambda$ vers p des $p+k$ sorties. J'ai présenté des techniques permettant d'obtenir des bornes inférieures sur la taille de ces réseaux ainsi que des constructions. Les constructions sont pour la plupart asymptotiquement optimales quand les valeurs de λ et k sont bornées par $\log(n)/15$. J'ai également présenté des résultats sur un nouveau paramètre, qui est une variante de l'expansion, l'α-robustesse permettant de construire des réseaux pour de plus grandes valeurs de k et λ.

  Q : L'étude de l'α-robustesse n'est pas terminée. Afin d'avoir des construction explicites de réseaux, il est nécessaire d'avoir des constructions explicites de graphes ayant une grande α-robustesse. De plus, nous avons vu que l'α-robustesse d'un graphe à n sommets est au plus $\frac{n}{2}$ quand $\alpha > \frac{4}{5}$. Est il possible de généraliser ce résultat et de donner une condition sur α assurant une α-robustesse d'au plus $n/3$ par exemple ?

- Pour le dimensionnement et l'exploitation des réseaux, je me suis ensuite intéressé au problème de routage. Ce problème est important lors de la conception des réseaux afin de dimensionner les liens au mieux, et lors de son exploitation. Dans

ce dernier cas, les solutions doivent pouvoir être calculées rapidement. Il existe de nombreuses versions du problème de routage et autant de manières de l'aborder. J'ai eu trois approches différentes.

1. La première est la conception d'algorithmes, soit optimaux, soit d'approximation. Cela a été le cas pour les problèmes de routage de paquets dans les réseaux en forme de grille, pour lesquels j'ai présenté des algorithmes optimaux dans le cas du routage r-central, de la permutation et du (ℓ,ℓ)-routage, et un algorithme d'approximation pour le (ℓ,k)-routage. J'ai aussi conçu des algorithmes d'approximation pour le problème de routage dans les réseaux d'antennes, problème modélisé par la coloration k-impropre de graphes pondérés. Les coefficients des algorithmes d'approximation sont, pour k de un à cinq, $\frac{25}{13}$, $\frac{12}{7}$, $\frac{18}{13}$, $\frac{80}{63}$ et $\frac{49}{43}$. J'ai également étudié et montré que le problème d'ordonnancement de requêtes à taux constant est NP-dur dans les réseaux d'accès WiFi.

Q : Concernant les algorithmes d'approximation pour le routage dans les réseaux d'antennes, problème modélisé par la coloration k-impropre, il serait intéressant de savoir si des algorithmes ayant de meilleurs rapports d'approximation existent, et également quel est le meilleur rapport d'approximation que l'on peut garantir en temps polynomial.

Q : Pour le problème du routage des requêtes à taux constant, modélisé par la coloration proportionnelle, il n'existe pas encore d'algorithmes d'approximation. Il serait donc intéressant d'en développer ou de trouver d'autres familles de graphes pondérés pour lesquelles calculer l'indice chromatique proportionnel est polynomial.

2. La deuxième approche a été de comparer deux modes de routages différents : le routage unicast et le routage multicast. Pour cela j'ai étudié un problème de coloration -le directed star colouring- sur lequel j'ai fait de nombreuses avancées ; il est entre autre prouvé que $dst(D) \leq 2\Delta^-(D)+1$ et que $dst(D) \leq 2\max(\Delta^+(D),\Delta^-(D))$. Ce problème permet de dire que dans le cas d'un réseau optique équipé de splitters, si le routage est de profondeur bornée et que le nombre de feuilles de chaque arbre de routage est borné par une constante fois la charge maximale du réseau ℓ, il suffit de $\ell+O(\log(\ell))$ longueurs d'onde pour le réaliser alors que si le réseau n'est pas équipé de splitters, il peut falloir jusqu'à $(\Delta+1)\ell$ longueurs d'onde où Δ est le degré maximum du réseau.

Q : Les travaux sur le directed star colouring peuvent être poursuivis. En effet il reste plusieurs conjectures à résoudre. En particulier, est-il vrai que pour tout graphe orienté D, on a $dst(D) \leq 2\Delta^-(D)$?

3. Enfin, la troisième approche du problème de routage a été de concevoir des modèles mathématiques pour le résoudre à l'aide d'un solveur. J'ai utilisé

cette approche pour le problème de routage de multicasts dans les réseaux optiques de cœur et d'accès, pour le problème de routage d'unicasts dans les réseaux avec groupes de risques partagés, et pour le problème de protection par segments.

Q : Quant aux problèmes de routage pour lesquels j'ai conçu des modèles mathématiques dont je vais implémenter les formulations qui ne l'ont pas encore été, ils sont une première étape pour l'étude de problèmes plus complexes dont le routage de multicast dans les réseaux ayant des groupes de risques et la protection dans ces mêmes réseaux, tant des unicasts que des multicasts. Pour ces problèmes, il est difficile de concevoir des algorithmes d'approximation pertinents car les requêtes que l'on considère et les réseaux sont quelconques. Il serait intéressant de savoir s'il existe une structure type dans la topologie des requêtes et dans l'affirmative de l'étudier et d'en proposer un modèle pour pouvoir concevoir des algorithmes d'approximation.

– Pour un réseau dont les requêtes évoluent dynamiquement, plutôt que de recalculer à chaque changement un nouveau routage, il est plus intéressant de se baser sur le routage existant et de le faire évoluer. C'est ce problème -passer d'un routage à un autre- que j'ai étudié dans le Chapitre 7. Ce problème est associé à un paramètre : le process number. J'ai présenté un algorithme qui calcule le process number des arbres. Je présente ensuite un algorithme d'approximation pour les graphes planaires extérieurs qui a un facteur d'approximation de deux. Le process number étant fortement lié à la pathwidth, j'ai également étudié ce paramètre et fait plusieurs avancées sur des conjectures. Il est entre autre prouvé que tout graphe planaire extérieur G biconnexe vérifie $pw(G^*) \leq pw(G) \leq 2pw(G^*)-1$. Pour les graphes planaires, il est prouvé que tout graphe planaire trois connexe G vérifie $pw(G^*)/3-2 \leq pw(G) \leq 3pw(G^*)+2$ et que les graphes planaire quatre connexe G vérifient $pw(G^*)/2-1 \leq pw(G) \leq 2pw(G^*)+1$.

Q : Concernant le process number et la pathwidth, il reste à trouver un algorithme permettant de calculer ces paramètres sur les graphes planaires extérieurs et d'autre classes de graphes. Il serait également intéressant de savoir si pour tous les graphes planaires G, on a $pw(G^*)/2-1 \leq pw(G) \leq 2pw(G^*)+1$. J'ai également présenté plusieurs paramètres -pathwidth, process number, edge search number, contiguous search number- dont on ne sait pas si ils sont équivalents. Est-il possible, connaissant l'un d'eux, de calculer les autres en temps polynomial ?

– Enfin, j'ai étudié un problème théorique permettant de mieux comprendre la structure des graphes orientés. J'ai présenté une nouvelle caractéristique, appelée f-mixing. Il est prouvé que la plus grande valeur de f telle que tout graphe orienté f-mixing contient un circuit orienté de longueur quatre est C_1e^2/n^2 pour une petite constante C_1. Il est également montré que pour tout graphe orienté H biparti, il existe une constante C_1' telle que tout graphe orienté C_1'-mixing ayant suffisamment d'arêtes contient H comme sous-graphe.

Q : À propos des graphes orientés f-mixing, il reste beaucoup de questions ouvertes.

La valeur de $mex(n,e,H)$ n'est connue que quand e est de l'ordre de n^2 ou quand H est une orientation du cycle de longueur quatre. Tous les autres cas restent à résoudre. Une autre question importante est de trouver la complexité du problème de calculer, dans un graphe orienté, deux ensembles A et B tels qu'il n'y a pas d'arcs de B vers A et que le nombre d'arcs de A vers B soit maximum. Tout ce que l'on sait est que ce problème est NP-dur dans le cas des graphes orientés.

Annexe A

Conception de réseaux tolérants aux pannes

A.1 Minimal Selectors and Fault Tolerant Networks

Article accepté pour publication dans Networks.

Minimal Selectors and Fault Tolerant Networks*

Omid Amini[1,2] Frédéric Giroire [2,3] Florian Huc [2]
Stéphane Pérennes [2]

Abstract

In this paper we study a combinatorial optimization problem arising from on-board networks in satellites. In these kinds of networks the entering signals (inputs) should be routed to amplifiers (outputs). The connections are made via expensive switches with four available links. The paths connecting inputs to outputs should be link-disjoint. More formally, we call a $(p, \lambda, k)-$network an undirected graph with $p + \lambda$ inputs, $p + k$ outputs and internal vertices of degree four. A $(p, \lambda, k)-$network is valid if it is tolerant to a restricted number of faults in the network, i.e., if, for any choice of at most λ faulty inputs and k faulty outputs, there exist p edge-disjoint paths from the remaining inputs to the remaining outputs.

Our optimization problem consists of determining $N(p, \lambda, k)$, the minimum number of vertices in a valid $(p, \lambda, k)-$network. We present validity certificates and a quasi-partitioning technique from which we derive lower bounds for $N(p, \lambda, k)$. We also provide constructions, and hence upper bounds, based on expanders. The problem is shown to be sensitive to the order of λ and k. For instance, when λ and k are small compared to p, the question reduces to the avoidance of some forbidden local configurations. For larger values of λ and k, the problem is to find graphs with a good expansion property for small sets. This leads us to introduce a new parameter called α-robustness. We use α-robustness to generalize our constructions to higher order values of k and λ. In many cases, we provide asymptotically tight bounds for $N(p, \lambda, k)$.

Keywords: Fault Tolerant Networks, Switching Networks, Network Design, Concentrators, Routing, Connectivity, Disjoint paths, Expanders.

1 Introduction

Motivation. One of the most fundamental problems in many applications of computer science and graph theory is the design of *efficient networks*. Efficient can have several meanings depending on the context. Properties such as high connectivity, fault tolerance and being algorithmically simple are at the heart of such concerns. In this paper we study a combinatorial optimization problem of this kind. The problem, originally posed by Alcatel Space Industry, comes from

*This work has been partially supported by European project IST FET AEOLUS.

[1]École Polytechnique, Palaiseau, FRANCE, firstname.lastname@polytechnique.org

[2]Mascotte, joint project- INRIA/CNRS-I3S/UNSA- 2004, route des Lucioles - Sophia-Antipolis, FRANCE, firstname.lastname@sophia.inria.fr

[3]This work was done while in the Algorithms project, INRIA Rocquencourt, F-78153 Le Chesnay, France.

the design of efficient on-board networks in satellites (also called Traveling Wave Tube Amplifiers). The satellites under consideration are used for TV and video transmission (like the Eutelsat or Astra series) as well as for private applications. Signals incoming to a telecommunication satellite through ports have to be routed through an on-board network to amplifiers. A first constraint is that the network should be built with switches having four links. But other constraints appear. On the one hand, the amplifiers may fail during the satellite's life time and can not be repaired. On the other hand, as the satellite is rotating, not all the ports are well oriented and hence available. So more amplifiers and ports are needed than the number of signals which have to be routed. Notice that in this context, contrary to the classical networks, we do not consider link and switch failure. As a matter of fact the links are just big tubes and the switches are very reliable rotating mechanical systems.
One can easily construct a network fulfilling these constraints by using a permutation network or two concentrators of any fixed degree. However, to decrease launch costs, it is crucial to minimize the network physical weight, i.e., to minimize the number of switches. Switches are also expensive to build, so it is worth saving even one. Space industries are interested in designing such networks for specific values of the parameters, however the general theory is of interest by itself. Also, due to the increasing launch capacities and the decreasing switch sizes, larger networks will be used in a near future.

Problem. We consider here *networks*, connecting a set of *inputs* to a set of *outputs*. A network can be seen as a graph with a set of inputs and outputs. The vertices of this graph represent the switches. We define a $(p, \lambda, k)-$*network* as a network with $p + \lambda$ inputs and $p + k$ outputs. A $(p, \lambda, k)-$network is said to be *valid* if it can tolerate up to λ faults in inputs and up to k faults in outputs. More formally for any choice of p inputs and p outputs, there exist p edge-disjoint paths linking all the chosen inputs to all the chosen outputs. By symmetry, we may assume in the following that $k \geq \lambda$ and we denote the total number of outputs by $n := p + k$. Fig 1 gives an example of a (12,4,4) valid network with 20 switches (the inputs are represented with arrows, the outputs with squares and the switches with black vertices).

Note that finding minimal networks is a challenging problem and even testing the validity of a given network is hard (see related work). We study the case where the switches of the network have degree four (although the theory can be generalized to any degree) which is of primary interest for the applications. The problem is to find $N(p, \lambda, k)$, the minimum number of switches in a valid $(p, \lambda, k)-$network, and to give constructions of such networks.

Related Work. This study of a new kind of networks should be considered in the spirit of the well studied theory of *superconcentrators*. A $(p+k, p)$-*concentrator* [31] is a directed acyclic graph $G = (V, E)$ with $p + k$ designated input vertices and p designated output vertices ($k \geq 0$), such that for each subset of p input vertices, there exist p edge-disjoint (or, depending on the context, vertex-disjoint) paths from the input vertices to the output vertices. A *selector*, first introduced in [9], is a $(p, 0, k)$-network ($\lambda = 0$). A general theory of selectors can be found in [12], where several results are obtained for small values of k. A selector can be seen as an undirected version of a $(p + k, p)$-concentrator by changing the role of inputs and outputs (and by adding the degree condition). An *n-superconcentrator* [28]

is a directed acyclic graph $G = (V, E)$ with n designated input vertices and n designated output vertices such that, for any set S of $p \leq n$ inputs and any set T of p outputs, there exist p edge-disjoint (or, vertex-disjoint depending on the context) paths connecting S to T. There is a vast theory of superconcentrators (see the seminal work of Pippenger [28] for an introduction, [2, 3, 5, 30] for constructions, and [14] for complexity issues).

In this way, (p, λ, k)-networks generalize the concept of selectors as superconcentrators generalize concentrators. It is also clear that taking a superconcentrator (resp. $(p + k, p)-$concentrator) and forgetting the orientations provides a valid (p, λ, k)-network (resp. selector) for every value of $k = \lambda$ (resp. any k and $\lambda = 0$), once the degree condition is respected. So one can try to use superconcentrators to construct efficient on-board satellite networks, but, not surprisingly, this does not give minimal networks in general. The major difference between the superconcentrators and the general valid networks, beside the degree condition, is that in a valid network the number of inputs and outputs that may become out of service is bounded (in our case, given a failure probability, no more than k failures will occur during the satellite lifetime). Finding a minimum network (a superconcentrator of low density or having a minimum number of vertices) is a challenging problem and is an active area of research. From the algorithmic point of view, one can show that the problem of testing if a graph is a $(p + k, p)$-concentrator or a superconcentrator is coNP-complete. In fact, this is coNP-complete even when restricted to the important special case of graphs of size linear in the number of inputs (see [14]). By the same arguments one can prove that the problem of deciding if a given (p, λ, k)-network is valid or not is coNP-complete.

In [12] the authors consider a variant of selectors where some signals have priority and should be sent to amplifiers offering the best quality of service. In [8] the authors study the case where all the amplifiers are different and where a given input has to be sent to a dedicated output. This problem is related to permutation networks.

In [11, 21], the authors present exact values of $N(p, \lambda, k)$ for small values of k and λ and arbitrary values of p. For instance $N(p, 2, 1) = N(p, 1, 2) = N(p, 2, 2) = p+2$ and also for $k \in \{3, 4\}$ and $0 < \lambda \leq 4,\ N(p, \lambda, k) = \lceil \frac{5n}{4} \rceil$.

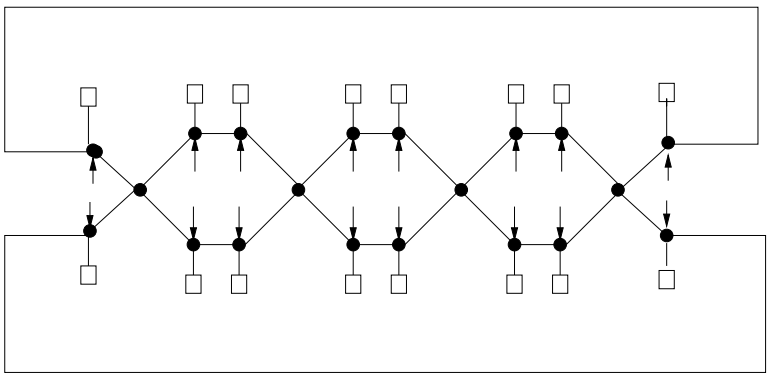


Figure 1: A valid (12,4,4)-network with the minimum number of switches. Switches, vertices of the corresponding graph, are represented by •.

Results. In this paper we are primarily interested in large networks, i.e., where $n = p+k$ tends to infinity and also k is large enough. We present several upper and lower bounds for the minimum number of switches in a valid $(p, \lambda, k)-$network. In many cases these bounds are asymptotically tight. Since forgetting the orientation of superconcentrators provides valid $(p, \lambda, k)-$networks, these results also provide lower bounds on the size of a superconcentrator. To obtain the lower bounds we present a new technique, *quasi-partitioning*. We give here the very first applications of this lemma, but we think it may have several applications in other related problems (for example bisection-width, see [24]).

To obtain upper bounds we propose several constructions. The construction of optimal valid networks will heavily rely on expanders (see Section 4.1). Using these, we are able to construct, for a special class of networks called simplified networks, networks with $2n$ switches as soon as n is large enough and $k \leq c_1 \log n$ for some constant c_1 (Section 4.3). In Section 6 we also give a lower bound of order $2n(1 - \epsilon(k))$ where $\epsilon(k)$ tends to zero when k tends to infinity (but we do not need $k \leq c_1 \log n$). Thus for simplified networks the problem is asymptotically solved for $k \leq c_1 \log n$. For general networks, using bipartite expanders, we obtain an upper bound of $n + \frac{3}{4}n$ when $k \leq c_2 \log n$ for some constant c_2. The lower bound we obtain is $(n + \frac{2}{3}n)(1 - \epsilon(\lambda, k))$, and we conjecture that $n + \frac{3}{4}n$ should be the right value. We also give a construction of selectors (case $\lambda = 0$) of size $n + \frac{n}{2}$, in which case we get also a tight lower bound.

To extend the results to larger values of k, we define a local expansion property of graphs that we call $\alpha-$*robustness* (see [4, 16] for some related notions). Intuitively, the $\alpha-$robustness of a graph G is a local version of the expansion factor. It is the maximum integer r_α such that for small subsets the expansion factor is α but for larger subsets, the minimum number of outgoing edges is required to be at least r_α. In our constructions, we only need our graph to be an expander for small subsets, but for larger subsets all we need is a minimum constant edge-connectivity to the rest of the graph. This notion is also interesting in its own right, and contains several expansion-invariants such as bisection-width and Cheeger's constant as special cases (see [18]). As random graphs are very good expanders, we study the $\alpha-$robustness of random graphs. In this way we generalize Bollobàs' result on the expansion factor of random 4-regular graphs (see [15]) to obtain related results for $\alpha-$robustness of random 4-regular graphs. As far as we know it is the first time in the literature that this new concept of local expanders is defined and investigated. Using this we present a construction of valid $(p, \lambda, k)-$networks with $3n$ switches for $\lambda \leq k \leq \frac{n}{7}$.

2 Preliminaries

In this section, we define more formally the design problem and introduce notation used throughout the paper. We state a *cut criterion* (Proposition 1): this criterion is fundamental because it characterizes the validity of $(p, \lambda, k)-$networks. It is extensively used to prove that networks are valid —in particular for the networks of Section 4. In Section 6 we also use the cut criterion to detect forbidden patterns leading to lower bounds for the number of switches of valid networks. Proofs of lower and upper bounds are simplified by the use of the last notion introduced here, the *associated graph* of a network (see Section 4.2 and Section 6.1).

Notation. Given a function f, we define $f(A) := \sum_{a \in A} f(a)$ for any finite set A. For a subset W of vertices of a graph $G = (V, E)$, let us denote by $\Delta(W)$ the set of edges connecting W and $\overline{W} = V \backslash W$ and by $\Gamma(W)$ the set of vertices of $\overline{W}$ adjacent to a vertex of W. If a set is denoted by an upper case letter, the corresponding lower case letter denotes its cardinality (for example $\delta(W) = |\Delta(W)|$).

$(p, \lambda, k)-$Networks and Valid $(p, \lambda, k)-$Networks. A *$(p, \lambda, k)-$network* is a triple $\mathcal{N} = \{(V, E), i, o\}$ where (V, E) is a graph and i, o are non-negative integer functions defined on V called input and output functions, such that for any $v \in V$, $i(v) + o(v) + deg(v) = 4$. The total number of inputs is $i(V) = \Sigma_{v \in V} i(v) = p + \lambda$, and the total number of outputs is $o(V) = \Sigma_{v \in V} o(v) = p + k$. We can see a network as a graph where all vertices but the leaves have degree four, and in which inputs and outputs are leaves. A *non-faulty output function* is a function o' defined on V such that $o'(v) \leq o(v)$ for any $v \in V$ and $o'(V) = p$. A *used input function* is a function i' defined on V such that $i'(v) \leq i(v)$ for any $v \in V$ and $i'(V) = p$. A (p, λ, k)-network is said *valid* if for any non-faulty output function o' and any used input function i', there are p edge-disjoint paths in G such that each vertex $v \in V$ is the initial vertex of $i'(v)$ paths and the terminal vertex of $o'(v)$ paths.

Design Problem and Simplified Design Problem. Let $N(p, \lambda, k)$ denote the minimum number of switches of a valid (p, λ, k)-network. The *Design Problem* consists of determining $N(p, \lambda, k)$ and constructing a minimum (p, λ, k)-network, or at least a valid (p, λ, k)-network with a number of switches close to the optimal value. We introduce a variation of the problem, the *Simplified Design Problem*, which is to find minimum networks having $p + \lambda$ switches with one input and one output and $k - \lambda$ switches with only one output. Networks of this kind are especially good for practical applications, as they simplify the routing process, minimize path lengths and lower interferences between signals.

Excess, Validity and Cut Criterion. To verify if a network is valid, instead of solving a flow/supply problem for each possible configuration of output failures and of used inputs, it is sufficient to look at an invariant measure of subsets of the network, the *excess*, as expressed in the following proposition.

Proposition 1 (Cut Criterion) *A $(p, \lambda, k)-$network is* valid *if and only if, for any subset of vertices $W \subset V$ the* excess of W*, defined by*

$$\varepsilon(W) := \delta(W) + o(W) - \min(k, o(W)) - \min(i(W), p),$$

satisfies $\varepsilon(W) \geq 0$.

The intuition is that the signals arriving in W (in number at most $\min(i(W), p)$) should be routed either to the valid outputs of W (in number at least $o(W) - \min(k, o(W))$) or to the links going outside (in number $\delta(W)$). The omitted formal proof reduces to a supply/demand flow problem. Note that, for the cut criterion, it is sufficient to consider only connected subsets W with connected complement $\overline{W}$ (this comes from the submodularity of ε).

Associated Graph. Vertices $x \in V$ of degree two with $i(x) = o(x) = 1$ play an important role. We call them *doublons*. A switch that is not a doublon is called an

R-switch. Remark that for $k \geq 3$ no paths has three or more consecutive doublons. Indeed if we consider the set W consisting of these doublons we have $\delta(W) = 2$ and $o(W) = i(W) \geq 3$. The cut criterion would give a contradiction.

Let $\mathcal{N}$ be a $(p, \lambda, k)-$network. We build a *graph* $\mathcal{R}$ *associated with* $\mathcal{N}$. Its vertices are the R-switches of $\mathcal{N}$. Consequently the edges of $\mathcal{R}$ are of three kinds, respectively E_0, E_1 and E_2: the edges of $\mathcal{N}$ between two $R-$switches, the edges corresponding in $\mathcal{N}$ to a path of length two with a doublon in the middle and those corresponding to a path of length three with two doublons in the middle.

3 Summary of Main Results

The aim of this short section is to provide an overview of the results obtained in the upcoming sections.

The design problem is treated in Sections 4 and 5. For relatively small values of k and λ, based on expanders, we provide construction of valid networks for the general design problem, the design problem when $\lambda = 0$ (selectors) and the simplified design problem. More precisely, we prove the following theorems:

Theorem 2 of Section 4 (General Design Problem). Let $n = p + k$, $k \leq \frac{1}{15} \log n$, for n large enough, we have: $N(p, \lambda, k) \leq n + \frac{3}{4}n$

Theorem 3 of Section 4 (Selectors, $\lambda = 0$). Let $n = p + k$, $k \leq \frac{1}{48} \log_2 n$, for n large enough, we have:

$$N(p, 0, k) \leq n + \frac{n}{2}.$$

Theorem 1 of Section 4 (Simplified Design Problem). Let $n = p+k$, $k \leq \frac{1}{6} \log n$, for n large enough, we have: $N(p, \lambda, k) \leq 2n$.

In Section 5, we provide a general framework to extend the above mentioned results to larger values of λ and k. We define a new parameter, which we call *robustness*, which captures the local expansion property of a given graph. We study the robustness of random regular graphs and use these graphs for the design problem. In particular, we prove

Theorem 4 of Section 5. Random 4-regular graphs have 1-robustness at least $\frac{n}{14}$ with high probability, i.e., with probability $1 - \mathcal{O}(\frac{\log n}{n})$.

Corollary 1 of Section 5. For $k \leq \frac{n}{7}$, there exist $(p, \lambda, k)-$valid networks of size $3n$ (remember that $n = p + k$).

The lower bounds, confirming the tightness of the above results in many cases, are provided in Section 6. We first prove a technical *quasi-partition* lemma:

Lemma 4 of Section 6. Let q be a positive integer and let G be a connected graph of order at least $\frac{q}{3}$. Then G admits a q-quasi-partition (for the definition see Section 6).

Using this lemma, we prove the following general theorem:

Theorem 5. In a valid network $\mathcal{N}$ of size N, with $k \leq \frac{n}{2}$, we have

$$N \geq \left(\frac{3}{2}n - (k-\lambda) - 4 - \frac{12}{\sqrt{k}}\right)\left(1 - \frac{9}{2\sqrt{k}} + O\left(\frac{1}{k}\right)\right) + \frac{d}{2}\left(1 + \frac{9}{2\sqrt{k}} + O\left(\frac{1}{k}\right)\right).$$

where $n = p + k$ and d is the number of doublons, i.e., switches with exactly one input and one output.

Corollaries of this theorem are

Theorem 8 of Section 6 (General case). When $\lambda \to \infty$ and $k \to \infty$, any valid network has size at least $n+\frac{2}{3}n+O(\frac{n}{\sqrt{\lambda}})$. In particular, $N(p,\lambda,k) \geq n+\frac{2}{3}n+O(\frac{n}{\sqrt{\lambda}})$.

Theorem 6 of Section 6 (Selectors). In a valid selector $\mathcal{N}$ of size N, when $k \to \infty$ with $k \leq \frac{n}{2}$, we have

$$N \geq n + \frac{n}{2} + \mathcal{O}(\frac{n}{\sqrt{k}}).$$

In other words

$$N(p,0,k) \geq n + \frac{n}{2} + O(\frac{n}{\sqrt{k}}).$$

Theorem 7 of Section 6 (Simplified case). In the simplified case, when $k \to \infty$ with $k \leq \frac{n}{2}$, every valid network has size at least $2n + O(\frac{n}{\sqrt{k}})$, i.e., we have $N(p,\lambda,k) \geq 2n + O(\frac{n}{\sqrt{k}})$ in the simplified case.

4 Upper Bounds: The Design Problem

In this section, we give constructions that rely heavily on expanders. We present here three constructions with $2n$, $n + \frac{3}{4}n$ and $n + \frac{n}{2}$ switches respectively for the simplified design problem, the design problem (any λ) and for the design problem when $\lambda = 0$. All these constructions are valid for $k \leq c \cdot \log n$ (where c is a constant depending only on the expansion factor of 3 and 4−regular graphs).

4.1 Expanders

An expander (see [26, 27] for surveys) is a highly connected sparse graph. They are used in various fields of computer science and mathematics. For example they are valuable in constructions of error-correcting codes with efficient encoding and decoding algorithms, derandomization of random algorithms, and embeddings of finite metric spaces. But they also have applications in areas directly related to the subject of this paper as in the design of explicit superefficient networks and the explicit construction of graphs with large girth (length of the smallest cycle). We present here known results about expanders that are used in proofs of Sections 4.3, 4.4 and 4.5. The formal definition of an expander is as follows: an (n,r,c)-E-expander is a finite r-regular graph $G = (V,E)$ with n vertices such that for any set A of vertices of G with $|A| \leq |V|/2$ we have $\delta(A) \geq c|A|$. Well known examples of expanders are Ramanujan graphs (for more on Ramanujan graphs see [25] and the book [19]). Explicit constructions of Ramanujan graphs are known for r of the form $r = q + 1$, with q a prime power (in particular for $r = 3$ and $r = 4$, which are

of special interest in our case). More precisely there exist explicit constructions of an infinite family $G_i = (V_i, E_i)$ of Ramanujan graphs such that $|V_i| \underset{i\to\infty}{\to} \infty$ with an expansion factor $c \geq 1 - \frac{4(r-1)}{r^2}$. It gives $c \geq \frac{1}{4}$ for 4−regular graphs and $c \geq \frac{1}{9}$ for 3-regular graphs. The girth of the graphs of this family satisfies: $g(G_i) \geq \frac{2}{3}\log_q |V_i|$. There also exists a family $H_i = (W_i, F_i)$ of explicit bipartite Ramanujan graphs of girth $g(H_i) \geq \frac{4}{3}\log_q |W_i|$. Probabilistic arguments show the existence of graphs, for any large order of networks (and not only for the specific values of both families), with the same properties of girth and even better expansion factor: for 4-regular graphs, expanders with $c \geq \frac{11}{25}$ exist (see [15]).

We also need a vertex-expansion version of expanders:

Definition 1 *(V-Expander) An (n, r, d)-V-expander is a finite r-regular graph $G = (V, E)$ with $|V| = n$ ($|V_1| = |V_2| = n$ in case of a bipartite graph and $V = V_1 \cup V_2$) such that for any subset A of vertices ($A \subset V_1$ when G is bipartite), the set of neighbors of $A, \Gamma(A) = \{v \in V | (v, u) \in E \text{ for some } u \in V\}$ satisfies*

$$|\Gamma(A)| \geq |A| + d(1 - |A|/n)|A|.$$

4.2 Cut Criterion for the Associated Graph

To simplify the proofs of validity of Sections 4.3, 4.4 and 4.5, it is sometimes better to work directly on the associated graph $\mathcal{R}$ of the (p, λ, k)−network $\mathcal{N} = ((V, E), i, o)$. More precisely, it means that, when applying the cut criterion on $\mathcal{N}$, it would be sufficient to consider only subsets of $\mathcal{R}$.

We introduce another notion of excess, ε', defined for all $W \subset V$ as $\varepsilon'(W) := \delta(W) + o(W) - \min(k, o(W)) - i(W)$. Note that $\varepsilon'(W) \leq \varepsilon(W)$. Hence, if, for all $W \subset V$, $\varepsilon'(W) \geq 0$, the network is valid. But we have no more an equivalence. The good property of ε' is that, if a switch x is a doublon, $\varepsilon'(W \cup \{x\}) \leq \varepsilon'(W)$. So it is enough to verify the cut criterion for subsets W of $\mathcal{N}$ consisting of a set of R−switches plus all the doublons on the edges of type E_1 and E_2 incident to the R−switches.

4.3 Simplified Design Problem - Upper Bound $2n$

In this subsection, we use the existence of expanders presented in Section 4.1 to construct valid (p, λ, k)−networks with $2n = 2(p + k)$ switches for large n and $k \leq c_1 \log n$ (c_1 depending on the expansion factor of 4−regular expanders, $c_1 = \frac{1}{6}$ when using explicit Ramanujan graphs). Furthermore we will show in Section 6 a lower bound of the same order for the simplified design problem.

Theorem 1 *Let $n = p + k$, $k \leq \frac{1}{6}\log n$, for n large enough, we have: $N(p, \lambda, k) \leq 2n$.*

Proof The results on expanders presented in Section 4.1 state the existence of an $(n, 4, c = \frac{1}{4})$−E-expander, $G = (V, E)$, of girth g, $g \geq \frac{2}{3}\log n$. Let $k \leq c \cdot g$ and $p = n - k$. We claim that in a 4−regular graph there exists a family of vertex-disjoint cycles covering all vertices of G. For now, suppose this is true and call this disjoint union of cycles F. We add n new vertices on each edge of F by subdividing

each edge of F into two edges. On each new vertex, we add an output and input creating a doublon. We now have a $(p, k, k)-$network $\mathcal{N}$ with $2n$ switches.

Let us prove that the constructed network is valid. We use the cut criterion on $\mathcal{R}$, which, in that case, is exactly G. Notice first that the network is symmetric in inputs and outputs and that for any subset $W \subset V$ of vertices, $i(W) = o(W)$. Hence we have

$$\varepsilon'(W) = \delta(W) - \min(k, o(W)).$$

Furthermore, note that when a network is symmetric, it is sufficient to verify the cut criterion only for subsets W with $|W| \leq |V|/2$.

- If $|W| \geq \frac{k}{c}$ then, by the expansion property, there are at least k edges between W and $\overline{W}$ and so $\varepsilon'(W) \geq 0$.
- Otherwise, if $|W| < \frac{k}{c} \leq g$, we have $|W| < g$ and thus W is acyclic. There are at most $|W| - 1$ edges inside W and, as G is 4-regular, we have $\delta(W) \geq 2|W|+2$. Let $e_F(W)$ be the number of edges of F incident to a vertex of W; by construction $o(W) = e_F(W)$. As the cycles of F are disjoint, $e_F(W) \leq 2|W|$. Hence $\delta(W) \geq e_F(W) = o(W)$, that is $\varepsilon'(W) \geq 0$.

The $(p, k, k)-$network is valid. We can derive $(p, \lambda, k)-$networks for any $\lambda \leq k$ by deleting $k - \lambda$ inputs.

The only remaining point now is to prove the claim. We prove it for any $2r$-regular graph. Let $G = (V, E)$ be a $2r-$regular graph. Since G is Eulerian, we can give to G an orientation, such that each vertex has in-degree and out-degree r. Let $D = (V, A)$ be the corresponding digraph. Now consider the following bipartite graph: $H = (V \times \{1\} \cup V \times \{2\}, E')$ where $E' = \{((x, 1), (y, 2))|(x, y) \in A\}$. It is clear that H is a bipartite r-regular graph, and so it admits a perfect matching. Let F be the edges of G given by this perfect matching. It is easy to check that F is a 2-regular graph covering all the vertices, i.e., F is a disjoint union of cycles covering all the vertices.

□

4.4 General Design Problem - Upper Bound $n + \frac{3}{4}n$

In this subsection, we construct general $N(p, \lambda, k)-$networks for large n, $k \leq c_2 \log n$ (where c_2 depends on the expansion factor of $3-$regular expanders). Theorem 2 gives an $n + \frac{3}{4}n$ upper bound for such networks. Constructions are based on 3-regular bipartite expanders.

Definition 2 Two edges are at distance at least d *if any path that contains both of them is of length at least* $d + 2$. A vertex is at distance at least d of an edge *if any path that contains both of them is of length at least* $d + 1$.

Lemma 1 *Let G be a $3-$regular bipartite graph $G = (V_1 \cup V_2, E)$ of large girth ($g = \Theta(\log |V_1|)$) which is a $(2|V_1|, 3, c)$-E-expander and suppose $2k \leq cg$. Let $\mathcal{F}$ be a set of selected edges, such that any two edges of $\mathcal{F}$ are at distance at least 3. The network $\mathcal{N}$ obtained from G by adding a doublon on each edge of $\mathcal{F}$, an input on each vertex of V_1 and an output on each vertex of V_2 is a valid network.*

Proof

We use the cut criterion on the associated graph following Section 4.2. As the construction is symmetric in inputs and outputs, it is sufficient to consider connected subsets $W \in V$ with $|W| \leq \lceil \frac{|V|}{2} \rceil$. The cut criterion is implied by

$$\varepsilon'(W) = \delta(W) + o(W) - \min(o(W), k) - i(W) \geq 0.$$

We now distinguish two cases for W.

- case 1: $|W| \leq \frac{2k}{c} \leq g$.

  As $o(W) - min(o(W), k) \geq 0$, we have

  $$\varepsilon'(W) \geq \delta(W) - i(W).$$

  Hence it is sufficient to prove $\delta(W) \geq i(W)$. As $|W| \leq g$, there are no cycles inside W and therefore there are $|W| - 1$ edges inside; G is $3-$regular, so $\delta(W) = |W| + 2$. Furthermore, $i(W) = |V_1 \cap W| + d$, so we have to prove that $|V_2 \cap W| + 2 \geq d$.

  Consider a doublon D incident to W and its associated edge $e(D) = \{v_1(D), v_2(D)\}$ with $v_1(D) \in V_1$ and $v_2(D) \in V_2$. If $v_2(D) \in W$, associate D with $v_2(D)$. If $v_2(D) \notin W$, then $v_1(D) \in W$. Associate to D a neighboor of $v_1(D) \in W$. As the distance of two edges of $\mathcal{F}$ is at least 3, different doublons have different associate vertices in $V_2 \cap W$. So $|V_2 \cap W| \geq d$.

- case 2: $|W| \geq \frac{2k}{c}$. By definition of ε' we have

  $$\varepsilon' \geq \delta(W) + o(W) - k - i(W).$$

  - If $i(W) - o(W) \leq k$, by the expansion property, we have $\delta(W) \geq 2k \geq i(W) - o(W) + k$.
  - If $i(W) - o(W) \geq k$, since the graph is bipartite, there are at least $3(i(W) - o(W))$ outgoing edges. So $\varepsilon'(W) \geq 2(i(W) - o(W)) - k \geq k > 0$.

□

Theorem 2 *(Construction) Let $n = p + k$, $k \leq \frac{1}{15} \log n$, for n large enough, we have: $N(p, \lambda, k) \leq n + \frac{3}{4}n$.*

Proof Take $H = (A, B, E)$, a bipartite $(2|A|, 4, c')$-E-expander with girth $g = \frac{4}{3} \log n$. Let k, $k \leq c' \cdot g \leq \frac{4}{3} \log n$ (for existence see Section 4.1). Take a complete matching F in the bipartite complement $\overline{H} = (A, B, \overline{E})$ of H, that is, every edge $\{u, v\} \in F$, has u and v in different parts of H, i.e. $u \in A$ and $v \in B$ OR $u \in B$ and $v \in A$, and u, v are not adjacent in H. For each edge $e = \{a, b\}$ of F, $a \in A$ and $b \in B$, replace a and b by three vertices a_1, a_2, a_3 and b_1, b_2, b_3, add edges $\{a_1, b_1\}$, $\{a_1, b_2\}$, $\{a_1, b_3\}$, $\{a_2, b_1\}$ and $\{a_3, b_1\}$. Finally join a_2 (resp b_2) to two neighbors of a (resp b) and a_3 (resp b_3) to the two other neighbors. See Figure 2. We obtain a 3-regular bipartite graph G, which, one can verify, is a $(6|A|, 3, \frac{c}{5}) - E-$expander. Note that by construction the edges of type $\{a_1, b_1\}$, with $\{a, b\} \in F$, form a selected set $\mathcal{F}$ of edges, that are pairwise at distance 3. We can apply Lemma 1 to G with $\mathcal{F}$ as selected set. All together we have $6|A| + |\mathcal{F}| = 7|A|$ switches, $|V_1| + |\mathcal{F} = 4|A|$ inputs, $|V_2| + |\mathcal{F} = 4|A|$ outputs. So, with $n = p + k$, $N(p, k, k) = 7|A| = \frac{7}{4}n$. We can now derive $(p, \lambda, k)-$networks for any $\lambda \leq k$ by deleting $k - \lambda$ inputs. □

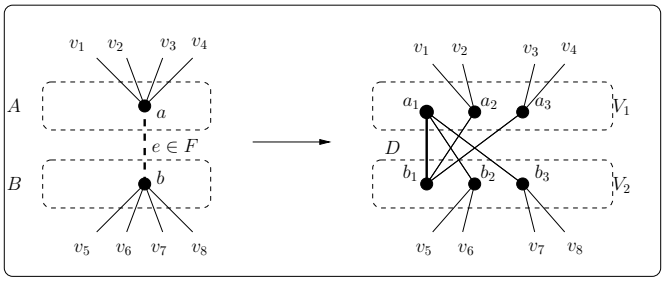


Figure 2: Replacement of a selected couple of a bipartite graph (A, B, E).

4.5 Design Problem: $\lambda = 0$ - Upper Bound $n + \frac{n}{2}$

In this subsection, we study selectors (the case $\lambda = 0$). In Theorem 3 we construct valid $(p, 0, k)-$networks with $n+\frac{n}{2}$ switches for n large and $k \leq c_3 \log n$ (c_3 depends on the vertex-expansion of 3-regular expanders). Such networks are built from bipartite $3-$regular vertex-expanders.

Theorem 3 *Let $n = p + k$, $k \leq \frac{1}{48} \log_2 n$. For n large enough, we have:*

$$N(p, 0, k) \leq n + \frac{n}{2}.$$

Proof Let us take $G = (V_1 \cap V_2, E)$ a bipartite $(n, 3, d = \frac{1}{12}) - V-$expander of large girth $g \geq \frac{4}{3} \log n$ (for existence see Section 4.1). Let $\alpha = \frac{4}{d} = 48$ and k such that $k \leq d \cdot \frac{g}{2} \leq \frac{g}{12}$ and $k.(2^{\alpha k} + 1) \leq n$.

To each vertex of V_1, we connect a vertex with one input and two outputs. Such a switch is said to be of type T. To each vertex of V_2, we connect an input. We choose a subset S of k vertices in V_2 such that the distance between any two of them is at least $\alpha \cdot k$ (see Definition 2). By the choice of k we can choose the vertices in S one by one after removing all the vertices at distance less than αk of already chosen vertices (at each step we remove at most $2^{\alpha k} + 1$ new vertices). We remove the inputs of all vertices of S obtaining a $(p = n - k, 0, k)-$network that we call $\mathcal{N} = (V, E, i, o)$.

We now prove that this network is valid. Let X be a connected subset of V. Let $\overline{X} = V_1 \cup V_2 \backslash X$, $A_1 = X \cap V_1$, $S_{\overline{X}} = S \cap \Gamma(A_1) \cap \overline{X}$. We define $Z(X) = X \cup \Gamma(A_1) \backslash S_{\overline{X}}$ (Figure 3).

$Z = Z(X)$ is connected. We also have $\varepsilon(Z) \leq \varepsilon(X)$, making it sufficient to verify the cut criterion for Z (same principles as in Section 4.2). Let $A_2 = Z \cap V_2$.

We distinguish four cases for $|A_1|$:

- $|A_1| \leq \frac{k}{2}$:
  as $o(Z) = 2|A_1| \leq k$ and $\varepsilon(Z) \geq \delta(Z) - i(Z)$, we have $\delta(Z) \geq 3|A_2| - 3|A_1|$ and $i(Z) \leq |A_2| + |A_1|$. Furthermore there is no cycle in Z so $|A_2| \geq 2|A_1|$. Hence $\varepsilon(Z) \geq 0$.
- $\frac{k}{2} \leq |A_1| \leq \frac{k}{d} \leq \frac{g}{2}$:
  we have $\varepsilon = \delta(Z) + o - i - k$ and $i \leq |A_1| + |A_2|$. So $\varepsilon(Z) \geq \delta(Z) - A_2 + A_1 - k$. As Z contains no cycle $\delta(Z) \geq 3|A_2| - 3|A_1|$ so $\varepsilon \geq 2|A_2| - 2|A_1| - k \geq 2|A_1| - k \geq 0$. We use again that $|A_2| \geq 2|A_1|$.

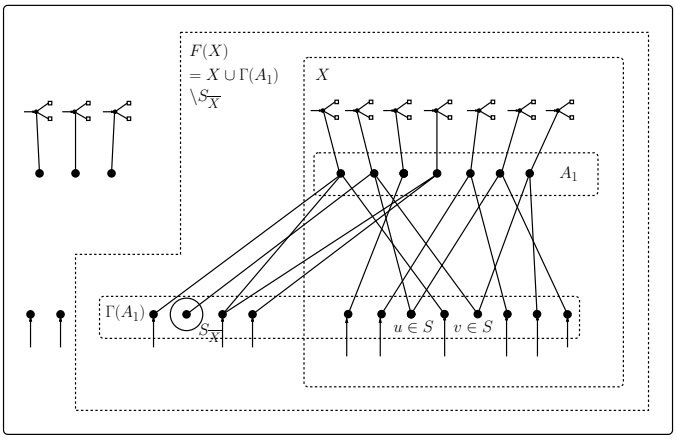


Figure 3: Sketch of proof of Theorem 3.

- $\frac{k}{d} \leq |A_1| \leq n - \frac{k}{d}$:
  in this case, $\varepsilon \geq \delta(Z) - |A_2| + |A_1| - k$ and $\delta(Z) \geq 3(|A_2| - |A_1|) = |A_2| - |A_1| + 2(|A_2| - |A_1|)$, so we have $\varepsilon \geq 2(|A_2| - |A_1|) - k$. Furthermore $|A_2| - |A_1| \geq \Gamma_G(A_1) - k$, because at most k vertices of $\Gamma_G(A_1)$ are in $S \setminus A_2$ (where $\Gamma_G(A_1)$ is the neighborhood of A_1 in G). By the expansion property, we have:

  $$|\Gamma_G(A_1)| \geq |A_1| + d(1 - |A_1|/n)|A_1|.$$

  In this case $d(1 - |A_1|/n) \geq 2k$. So $|\Gamma_G(A_1)| - |A_1| \geq 2k$. So $|A_2| - |A_1| \geq k$. It implies $\varepsilon \geq 0$.

- $n - \frac{k}{d} \leq |A_1|$:

  we can assume that $\overline{Z}$ is connected (see the remarks on Proposition 1). As the vertices of S are at distance at least αk with $\alpha k > \frac{k}{d}$, $\overline{Z}$ contains at most one vertex of S. Hence $i \geq |A_2| - k + 1 + |A_1|$ and $\varepsilon \geq \delta(Z) - |A_2| + |A_1| - 1$. Because of the connectivity of Z and the $3-$regularity $|A_2| > |A_1|$. So $\varepsilon \geq 0$.

□

5 Graph Robustness: A New Approach to the Design of Valid Networks

The construction of valid networks is related to a more general expansion property of graphs, more precisely to what can be seen as a 'bounded expansion property' or expansion property only for sets of bounded size. We call this property *robustness*.

Definition 3 *(α-robustness) The α-robustness, rob_α, of a graph G is the maximum value r, such that, for every subset $X \subset V(G)$ with $|X| \leq \frac{|V|}{2}$, we have $\delta(X) \geq \min(r, \alpha|X|)$.*

For $\alpha = 1$ we simply say robustness. There are two main problems: on the one hand we want to understand the behavior of the robustness for a general graph,

which means giving some upper bounds, for instance showing the existence of subsets which violate the $\alpha-$robustness. This problem covers in particular the classical problem of bisection-width, which has several important applications, see for example [1] and [24]. On the other hand, for many applications, one would like to be able to find graphs of large robustness ($\alpha-$robustness). The first examples of such graphs are expanders of expansion factor c which give graphs with $c-$robustness at least $\frac{n}{2}$ where n is the number of vertices. For d-regular graphs we have:

- if G is d-connected, $rob_d = 1$ and $rob_{d-1} = 2$. $rob_{d-2} = g$ where g is the double girth of the graph (minimum size of a cycle with a chord);
- for $\alpha \leq \frac{1}{2}(d - 2\sqrt{d-1})$, $rob_\alpha \geq \frac{n}{2}$ for Ramanujan graphs;
- for $\alpha > \frac{d}{2} - c\sqrt{d}$, where c is some absolute constant depending only on d, $\alpha-$robustness is strictly smaller than $\frac{n}{2}$ (see Alon [1]);
- for $\alpha > \frac{1}{3} + \epsilon$ the $\alpha-$robustness of a 3-regular graph is strictly smaller than $\frac{n}{2}$, the same is true for $\alpha > \frac{4}{5} + \epsilon$ and for 4-regular graphs (see Monien-Preis [24]).

Remark that, since it is NP-hard to compute the double girth of a graph, computing $rob_2(G)$ is an NP-hard problem. The same is true for any α.

Remark 1 Before going through the study of robustness we would like to stress that the concept of robustness is closely related to the concept of *general selectors*. A *general selector* is a selector without degree conditions. More precisely we have a graph G, such that the set of vertices $V(G)$ is partitioned into three subsets: I the set of vertices with one input, O the set of vertices with one output and S the set of normal vertices with neither input nor output, with $|I| = p + k$ and $|O| = p$. We say that G is a general selector (or is a $\binom{p+k}{p}$-network) if for any subset of I of size p, there exist p edge-disjoint paths, each one joining one different input to one different output.

Selectors also provide a very natural framework for constructing general valid networks, i.e., valid networks without degree constraints. For instance, for any (p, λ, k), we can construct a general valid (p, λ, k)-network as follows: take two selectors, a $\binom{p+k}{p}$-network, A, and a $\binom{p+\lambda}{p}$-network, B. Let G be the graph obtained by taking the disjoint union of A and B, deleting the outputs connected to them, joining the two sets $O(A)$ and $O(B)$ by a matching and replacing the inputs connected to $I(B)$ by outputs. The resulting network is a general valid (p, λ, k)-network in which $I(A)$ is the set of vertices with one input and $I(B)$ is the set of vertices with one output.

In the following, r_α denotes the maximum α-robustness of a 4-regular graph.

5.1 Robustness of random 4-regular graphs

In this section $\mathcal{G}(k, n)$ is the probability space of all graphs on n vertices that are the union of k disjoint Hamiltonian cycles, all graphs occurring with the same probability. By $G_{k,n}$ we denote an element of $\mathcal{G}(k, n)$. This probability space is "equivalent" to the probability space of random $2k$-regular graphs, see [22].

Theorem 4 *Random 4-regular graphs have 1-robustness at least $\frac{n}{14}$ with high probability, i.e. with probability $1 - \mathcal{O}(\frac{\log n}{n})$.*

Corollary 1 *For $k \leq \frac{n}{7}$, there exist $(p, \lambda, k)-$valid networks of size $3n$ (remember that $n = p + k$).*

Proof Suppose that a $4-$regular Hamiltonian graph G of robustness $\frac{n}{7}$ on $2n$ vertices is given. Extract a complete matching from a Hamiltonian cycle and add a doublon to every edge of this matching. It is now straightforward to show that the resulting network is valid. This gives a valid $(p = n - k, k, k)$-network with $3n$ switches. □

Remark 2 The method of the proof of Theorem 4 can be also used to obtain lower bounds for α-robustness of random 4-regular graphs for other values of α. Some numerical results are given below for some special values of α:

- $r_{\frac{11}{25}} = \frac{n}{2}$ (See Bollobàs [15] for another proof of this result).
- $r_{\frac{11}{20}} \geq \frac{n}{2.6},\ r_{\frac{4}{5}} \geq \frac{n}{5.7},\ r_{\frac{2}{3}} \geq \frac{n}{3.6},\ r_{\frac{3}{5}} \geq \frac{n}{3}$.

Proof [of Theorem 4] We first give preliminary results on the unions of k disjoint Hamiltonian cycles. The results for 4-regular graphs are then a direct application with $k = 2$. Let G be a graph of $\mathcal{G}(k, n)$ and S be a subset of V with s vertices. On the cycles $C_1, C_2, \ldots C_k$ constituting G, S can be described as a set of continuous intervals. For $i = 1, \ldots k$, let $m_i(S)$ be the number of intervals that S defines on C_i. Then $\delta(S) = 2\sum_{i=1,2,\ldots k} m_i(S)$.

For a given cyclic permutation π, if $I(s, m)$ denotes the number of sets of s elements defining m segments on the cycle associated to π, we have:

Claim 1

$$I(s,m) = \binom{s-1}{m-1}\binom{n-s}{m} + \binom{n-s-1}{m-1}\binom{s}{m} = \binom{s}{m}\binom{n-s}{m}\left(\frac{m}{s} + \frac{m}{n-s}\right). \tag{1}$$

Proof

The number of ways to write a number t as the ordered sum of p non-negative integers is $cut(t, p) = \binom{t+p-1}{p-1}$. Also, the number of ways to write t as the ordered sum of p positive integers is $cut(t - p, p) = \binom{t-1}{p-1}$.

Starting from the vertex 0 of the cycle, the set S can be encoded by giving two sequences of numbers : the ordered list of the lengths of the segments in S and the ordered list of the lengths of the segments in $\overline{S}$. There are two cases, depending on wether the first interval belongs to S or to $\overline{S}$.

- If 0 belongs to $\overline{S}$, then S is associated with exactly m segments on the path, all having length at least 1, and $\overline{S}$ is associated to $m + 1$ segments, all but the last one having length at least 1 (the last segment can have length 0).

  So $\overline{S}$ can be encoded by dividing $|\overline{S}| - m = n - s - m$ into $m+1$ non-negative integers. There are exactly $cut(n - s - m, m + 1)$ ways of doing it.

  We should also write S as the ordered sum of m non null numbers, there are $cut(s - m, m) = \binom{s-1}{m-1}$ possibilities.

  This gives a contribution of $cut(n-s-m, m+1) \cdot cut(s-m, m) = \binom{s-1}{m-1}\binom{n-s}{m}$ to $I(s, m)$.

- If 0 is in S, the situation is symmetric, with $n-s$ instead of s: we replace $cut(n-s-m, m+1)cut(s-m, m)$ by $cut(s-m, m+1)cut(n-s-m, m) = \binom{s}{m}\binom{n-s-1}{m-1}$.

The total contribution is the sum of the partial contributions, and the claim follows. □

Given a set S with s elements, we denote by $P(s, m)$ the probability that for a random cyclic permutation π, S defines m segments on the cycle associated to π. Since the probability does not depend on S, but only on s, $P(s, m)$ is well defined. We have:

Claim 2

$$P(s,m) = I(s,m)/\binom{n}{s} = \frac{\binom{s}{m}\binom{n-s}{m}(\frac{m}{s}+\frac{m}{n-s})}{\binom{n}{s}}. \quad (2)$$

Proof Consider a bipartite graph whose vertices are the sets of s elements and the $(n-1)!$ cyclic permutations. We put an edge between a set and a cyclic permutation if the set defines m segments on the cycle associated to the permutation. This graph is a "regular bipartite graph", where sets have degree Δ and permutations have degree $I(s, m)$. Counting the edges of the graph, we get $\Delta = \frac{(n-1)!I(s,m)}{\binom{n}{s}}$.

Now, the probability for a set to be adjacent to a permutation is $\Delta/(n-1)!$. So, given a set S with s elements, the probability that S defines exactly m segments on a cycle C is $I(s, m)/\binom{n}{s}$. Then Equation 1 gives the result. □

Claim 3

$$Prob(\delta(S) = 2x) = \sum_{k_1,\ldots,k_k | k_1+k_2+\ldots+k_k=x} \prod I(s,k_i)/\binom{n}{s}^k. \quad (3)$$

Proof The result follows from the three points below:

- the probability that S defines k_i segments on a random cycle C_i is $I(s, k_i)$;
- the cycles constituting $G_{k,n}$ are independent
- $\delta(S) = 2\sum_{1,2,\ldots,k} k_i$.

□

We say that a set is *bad* if $\delta(S) < min(s, n-s, r)$. By the above equations we can estimate the probability of having bad sets. The following two lemmas complete the proof of our Theorem. □

Lemma 2 (Robustness for small sets) *Any set with size $s \leq \frac{n}{14}$ has border at least s with probability $1 - \mathcal{O}(\frac{\log n}{n})$.*

Proof

The number of cycles is two for 4-regular graphs. Let $W(n, s)$ be the probability that there is a set of size $s \leq r = \frac{n}{14}$ with border less than s:

$$W(n,s) \leq \binom{n}{s} \sum_{r \leq s/2} \sum_{m_1+m_2=r} I(s,m_1)I(s,m_2)/\binom{n}{s}^2.$$

Since $I(s,m_1)I(s,m_2) \leq I(s,\frac{m_1+m_2}{2})$, we have

$$W(n,s) = \sum_{r\leq s/2} \sum_{m_1+m_2=r} I(s,m_1)I(s,m_2)/\binom{n}{s} \leq \sum_{r\leq s/2} r(I(s,r/2))^2/\binom{n}{s}.$$

Now, since $r \leq s/2$, we have $I(s,r/2) \leq I(s,s/4)$ and $W(n,s) \leq \frac{s^2}{8}\left(I(s,s/4)\right)^2/\binom{n}{s}$. Replacing $I(s,s/4)$ by its value we get:

$$W(n,s) \leq \frac{s^2}{8}\binom{s}{s/4}^2\binom{n-s}{s/4}^2/\binom{n}{s}.$$

For the following we need the following rather sharp form of *Stirling's formula* proved by Robbins(1955):

$$n! = \left(\frac{n}{e}\right)^n \sqrt{2\pi n}e^{\alpha_n},$$

where $\frac{1}{12n+1} < \alpha_n < \frac{1}{12n}$.
Putting this formula in the above inequality we obtain:

$$W(n,s) \leq \frac{s^{3s}(n-s)^{3n-3s}}{(\frac{s}{4})^s(\frac{3s}{4})^{\frac{3s}{2}}(n-\frac{5s}{4})^{2n-\frac{5s}{2}}\ n^n}\ \beta,$$

where $\beta = \frac{16\ (n-s)\sqrt{s(n-s)}}{3\pi\sqrt{2\pi n}\ (4n-5s)}.e^{\gamma}$ with

$$\gamma = 3\alpha_s + 3\alpha_{n-s} - 4\alpha_{\frac{s}{4}} - 2\alpha_{\frac{3s}{4}} - 2\alpha_{n-\frac{5s}{4}} - \alpha_n,$$

α_i being a real number in the interval $[\frac{1}{12i+1}, \frac{1}{12i}]$. Let

$$X(n,s) := \frac{s^{3s}(n-s)^{3n-3s}}{(\frac{s}{4})^s(\frac{3s}{4})^{\frac{3s}{2}}(n-\frac{5s}{4})^{2n-\frac{5s}{2}}\ n^n}\ \beta.$$

Suppose $n = a \cdot s$. Then we have:

$$X(n,s) = \left(\frac{(a-1)^{3(a-1)}}{\frac{1}{4}.(\frac{3}{4})^{\frac{3}{2}}.(a-\frac{5}{4})^{2a-\frac{5}{2}}.a^a}\right)^s \beta.$$

If we note

$$g(a) := \frac{(a-1)^{3(a-1)}}{\frac{1}{4}.(\frac{3}{4})^{\frac{3}{2}}.(a-\frac{5}{4})^{2a-\frac{5}{2}}.a^a},$$

we have

$$X(n,s) = g(a)^s.\beta.$$

Remember that

$$\beta = \frac{16\ (n-s)\sqrt{s(n-s)}}{3\pi\sqrt{2\pi n}\ (4n-5s)}.e^{\gamma}.$$

It is now easy to see that g is decreasing. As $g(13.80\cdots) = 1$, we have $g(14) < 1$. Hence

- For $s \in [-\frac{4\log n}{\log(g(14))}, \frac{n}{14}]$ we have

$$X(n,s) \leq C.\frac{1}{n^4}\beta = O(\frac{1}{n^2}),$$

for some constant C. Hence for some other constant C,

$$W(n,s) \leq C.\frac{1}{n^2}.$$

- For $s \leq -\frac{4\log n}{\log(g(14))}$ we have $a \geq C.\frac{n}{\log n}$ (for some constant C). Hence

$$X(n,s) \leq C.(n/s)^{-\frac{s}{2}}.\beta.$$

It is clear that for $s \geq 4$, we have $X(n,s) \leq C.\frac{1}{n}$ for some constant C.

- It is easy to show that $W(n,s) = 0$ for $s = 1,2,3$, as the edge connectivity of a random 4-regular graph is 4 with very high probability.

Then we have

$$\sum_{s\leq\frac{n}{14}} W(n,s) \leq \sum_{-\frac{4\log n}{\log(g(14))}\leq s\leq\frac{n}{14}} X(n,s) + \sum_{s\leq-\frac{4\log n}{\log(g(14))}} X(n,s)$$
$$< C.n.\frac{1}{n^2} + \frac{C.\log(n)}{n} \;=\; \mathcal{O}(\frac{\log n}{n}).$$

□

Lemma 3 (Robustness for large sets) *For n large enough, any set of size $s \in [\frac{n}{14}, \frac{13n}{14}]$ has border at least $\frac{n}{14}$ with probability $1 - \mathcal{O}(\frac{1}{n})$.*

Proof

A set larger than $\frac{n}{14}$ is *bad* when its border is smaller than $\frac{n}{14}$. Let $W'(n,s)$ be the probability that there exists a set of size $s \geq r = \frac{n}{14}$ with border less than $r = \frac{n}{14}$. We have

$$W'(n,s) \leq \sum_{i\leq\frac{n}{28}} \sum_{k_1+k_2=i} I(s,k_1)I(s,k_2)/\binom{n}{s}.$$

Hence $W'(n,s) \leq \frac{(\frac{n}{14})^2}{8}\left(I(s,\frac{n}{56})\right)^2/\binom{n}{s}$.

Let define $\delta(a)$ for $a \in [\frac{1}{14}, 1-\frac{1}{14}]$ as

$$\delta(a) = \frac{\left(\binom{a.n}{\frac{n}{56}}\binom{(1-a)n}{\frac{n}{56}}\right)^2}{\binom{n}{a.n}}.$$

Then

$$W'(n,an) \leq \frac{1}{8}\left(\frac{n}{14}\right)^2\delta(a).$$

Again, using methods similar to the ones of last section, we obtain $W'(n,a) \leq h(a)^n\beta'$, where β' is a rational function on n and a, and h is defined below:

$$h(a) = \frac{a^{3a}\;(1-a)^{3-3a}}{(\frac{1}{56})^{\frac{1}{14}}\;(a-\frac{1}{56})^{2a-\frac{1}{28}}\;(1-a-\frac{1}{56})^{2-2a-\frac{1}{28}}}\beta'.$$

Since $h(\frac{1}{14}) < 1$, with h decreasing,

$$W'(n, an) \leq h(\frac{1}{14})^n \beta'.$$

As β' is rational, we have for n large enough $W'(n, an) < \frac{1}{2n^2}$, and so $W'(n, s) = \mathcal{O}(\frac{1}{n})$. So

$$\sum_{s \in [\frac{n}{14}\ ,\ \frac{13n}{14}]} W'(n, s) = \mathcal{O}(\frac{1}{n}),$$

and the lemma is proved. □

6 Lower Bounds

In this section we distinguish switches according to their number of inputs and outputs. Figure 4 defines the different set of switches: S, S_i, S_o, V_i, V_o, D and T. For example, a switch v is in D if $i(v) = o(v) = 1$. Remember that we called such a switch a doublon and that all switches that are not a doublon are called *R-switch*. Remark that direct applications of the cut criterion show that no other types of switches are possible and that, as soon as $\lambda \geq 1$, switches of kind T are forbidden. Recall the convention that a lower case letter in the notations indicates the cardinality of the set denoted by the corresponding upper case letter.

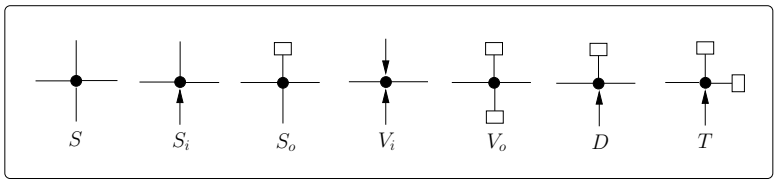


Figure 4: Kinds of switches.

Fundamental equations are linking the different kinds of switches: Equation 4 (*switch partition equation*) which counts the number N of switches of the network, Equation 5 (*input equation*) which counts the number of inputs and Equation 6 (*output equation*) which counts the number of outputs.

$$N = s + s_i + s_o + d + v_i + v_o + t \quad (4)$$

$$p + \lambda = s_i + 2v_i + d + t \quad (5)$$

$$p + k = s_o + 2v_o + d + 2t \quad (6)$$

In Subsection 6.1, we prove a fundamental preliminary theorem, Theorem 5. Its main point is that $N(p, \lambda, k) \geq \frac{3}{2}n + \frac{d}{2} - \varepsilon(n)$ (where d is the number of doublons and where $\varepsilon(n)$ goes to zero when n goes to infinity). Direct applications of this theorem give lower bounds for general and simplified networks (Theorems 6 and 7). We succeed in obtaining a better bound of $n + \frac{2}{3}n - \varepsilon(n)$ for two cases: when λ goes to infinity (Theorem 8) and when some kinds of switches are not allowed in the networks (Theorem 9). Finally (Theorem 10), we succeed to obtain a tight bound $n + \frac{3}{4}n - \varepsilon(n)$ for another family of networks using majority arguments.

6.1 Preliminary Theorem

The idea to find and prove lower bounds here is first to obtain local information on possible switch configurations. The methodology is to use the cut criterion (Proposition 1) to infer equations linking the numbers of switches of different kinds. Then, we use the existence of a quasi-partition (Lemma 4) to obtain bounds for the whole network.

Definition 4 (q-quasi-partition) *Let $G = (V, E)$ be a graph and q be a positive integer. A* q-quasi-partition *of G is a family $Q = \{A_1, A_2, \ldots, A_m\}$ of subsets of V, such that :*

(i) for every $1 \leq i \leq m$, the subgraph $G[A_i]$ induced by A_i is connected;

(ii) for every $1 \leq i \leq m$, $\frac{q}{3} \leq |A_i| \leq q$;

(iii) $V = \bigcup_{i=1}^{m} A_i$ and $\sum_{i=1}^{m} |A_i| \leq |V| + |\{A_i; |A_i| > \frac{2q}{3}\}| + 1$.

Lemma 4 ([20]) *Let q be a positive integer and G be a connected graph of order at least $\frac{q}{3}$. Then G admits a q-quasi-partition.*

Remark 3

- If G has several connected components, each of size at least $\frac{q}{2}$, applying the lemma to each component and using the additivity of both sides of Equation (iii) gives us a $q-$quasi-partition of G.
- Let Q be a quasi-partition of G as in Lemma 4. Let $t = |\{A_i, |A_i| \geq \frac{2q}{3}\}|$ and $v = |V|$. Then we have: $m \leq \frac{3(v+t+1)}{q}$ and $t \leq \frac{v+1}{\frac{2q}{3}-1}$.

Proof The lemma as stated in [20] is slightly incorrect. Hence we restate it here and give a new proof. Since every connected graph G contains a spanning tree, it is sufficient to prove the result for trees. We prove this by induction on the size of the trees. Let T be a tree. If T contains less than q elements, then the result trivially holds. So suppose that $|T| \geq q$. An edge e of T is called a *q-balanced separator*, if removing e separates the tree into connected components of sizes at least q. Let E_q be the subgraph consiting of all q-balanced edges. If there is no q-balanced edge in the tree, a *q-separator vertex* exists, that is a vertex v such that all the connected components of $T \setminus \{v\}$ have less than q vertices. Let us take l to be a leaf of E_q if some q-balanced edges exist or a q-separator vertex if $E_q = \emptyset$. $T \setminus (E_q \cup \{l\})$ has r connected components, say $C_1, \ldots, C_r$, such that $|C_1| \geq |C_2| \geq \cdots \geq |C_r|$. By the choice of l, and our hypothesis on the size of T ($|T| > q$), we have $|C_i| < q$ and $\sum_{i=1}^{r} |C_i| \geq q$. The following cases can appear.

- $|C_1| \geq \frac{q}{3}$ and $T \setminus C_1$ has at least $\frac{q}{3}$ vertices. Then by induction it has a q-quasi partition $\{A_1, \ldots, A_{m-1}\}$. Adding $A_m = V(C_1)$ to this family provides a q-quasi partition for T.
- $|C_1| < \frac{q}{3}$. We have $r \geq 3$. Let s be the largest integer such that $\sum_{i \leq s} |C_i| \leq q$. As $|C_{s+1}| \leq |C_1| < \frac{q}{3}$, we have $s \geq 3$ and $\sum_{i \leq r} |C_i| \geq \frac{2q}{3}$ (by maximality of s).

  So we can suppose that $\sum_{i \leq r} |C_i| \geq \frac{2q}{3}$. If the connected tree $T \setminus \bigcup_{i \leq s} C_i$ has more than $\frac{q}{3}$ elements, for example if $E_q \neq \emptyset$, by induction it admits a q-quasi partition $\{A_1, \ldots, A_{m-1}\}$. Adding $A_m = \bigcup_{i \leq s} C_i \cup \{l\}$ to this family provides a q-quasi partition for T.

- T has less than $q + \frac{q}{3}$ vertices, E_q is empty and we can find a vertex l such that each $|C_i|$ has less than $\frac{q}{3}$ vertices. Let t be the smallest integer such that $\sum_{i\leq t} |C_i| \geq \frac{q}{3}$. We claim that $T \setminus \bigcup_{i\leq t} C_i$ has at most q and at least $\frac{q}{3}$ vertices. The first one is true because T has at most $q+\frac{q}{3}$ vertices and $\bigcup_{i\leq t} C_i$ has at least $\frac{q}{3}$ vertices. The second one is true, because of the minimality of t. Indeed, $\bigcup_{i\leq t-1} C_i$ has at most $\frac{q}{3}$ vertices, C_t has at most $\frac{q}{3}$ vertices, and so because $|T| \geq q+1$, $T \setminus \bigcup_{i\leq t} C_i$ has at least $\frac{q}{3}$ vertices. It is now easy to check that $\bigcup_{i\leq t} C_i \cup \{l\}$ and $T \setminus \bigcup_{i\leq t} C_i$ form a q-quasi partition of T.

□

For the proof of Theorem 5 we need to define large and small $H-$components of $\mathcal{R}$, the associated graph of $\mathcal{N}$.

Definition 5 [***H*-component, large and small *H*-components, adjacent *H*-components**] *We consider a (p, λ, k)-network and its associated graph $\mathcal{R}$ (see Section 2 for definition). We take H the subgraph of $\mathcal{R}$ which contains only the edges of type E_0. An* H-component *of $\mathcal{R}$ is a connected component of H. An H-component is said* large *(respectively* small*) if it has more than (resp. strictly less than) q switches, with q the greatest integer satisfying $2(q+(2q+2)q)+2 \leq k-1$. Remark that $q \sim \frac{\sqrt{k}}{2}$. Two H-components C_1 and C_2 are said* adjacent *if there exists an edge of $\mathcal{R}$ with one $R-$switch in C_1 and the other in C_2.*

Proposition 2 *1. A small H-component has no outgoing edge of kind E_2.*

2. A small H-component has no input inside.

3. Two small H-components are not adjacent.

Proof Let C be a small H-component. We apply the cut criterion of Section 4.2 to the set $\tilde{C}$ of $\mathcal{N}$ obtained from C by adding in $\mathcal{N}$ the doublons of the edges of type E_1 and E_2 incident to $R-$switches of C.

As $k \leq p$ and $|C| \leq q \approx \frac{\sqrt{k}}{2}$, we have $o(C) \leq \frac{k}{2} \leq p$. Hence the cut criterion reduces to

$$\delta(\tilde{C}) \geq i(\tilde{C}).$$

Let e_1 (res. e_2) be the number of outgoing edges of kind E_1 (resp. E_2) incident to $R-$switches of C.

- By definition of H and of the small components, $\delta(\tilde{C}) = e_1 + e_2$. We have $i(\tilde{C}) \geq e_1 + 2e_2 + i(C)$. So by the cut criterion, $e_2 = 0$ proving 1) and $i(C) = 0$ proving 2).

- Let C' be an other small $H-$component. If C and C' are joined by $f \geq 1$ edges, then let $W = \tilde{C} \cup \tilde{C'}$, we have $i(W) \geq e_1 + e'_1 - f$ and $\delta(W) \leq e_1 + e'_1 - 2f$ so $\delta(W) < i(W)$, giving a contradiction.

□

Theorem 5 (Preliminary Theorem) *In a valid network $\mathcal{N}$ with $k \leq \frac{n}{2}$, we have*

$$N \geq \left(\frac{3}{2}n - (k-\lambda) - 4 - \frac{12}{\sqrt{k}}\right)\left(1 - \frac{9}{2\sqrt{k}} + O\left(\frac{1}{k}\right)\right) + \frac{d}{2}\left(1 + \frac{9}{2\sqrt{k}} + O\left(\frac{1}{k}\right)\right),$$

where $n = p + k$.

Proof According to Lemma 4 and Remark 3, the union of the large H-components of $\mathcal{R}$ admits a q-quasi-partition $Q = \{A_1, \ldots, A_m\}$. So each A_j is connected and of size $\frac{q}{2} \leq |A_j| \leq q$. We say that an edge with doublons (i.e. an edge of type E_1 or E_2) is of type:

- R if it is between an A_j and a small $H-$component,
- M if it is between two distinct A_j and A_k,
- N if it is inside an A_j.

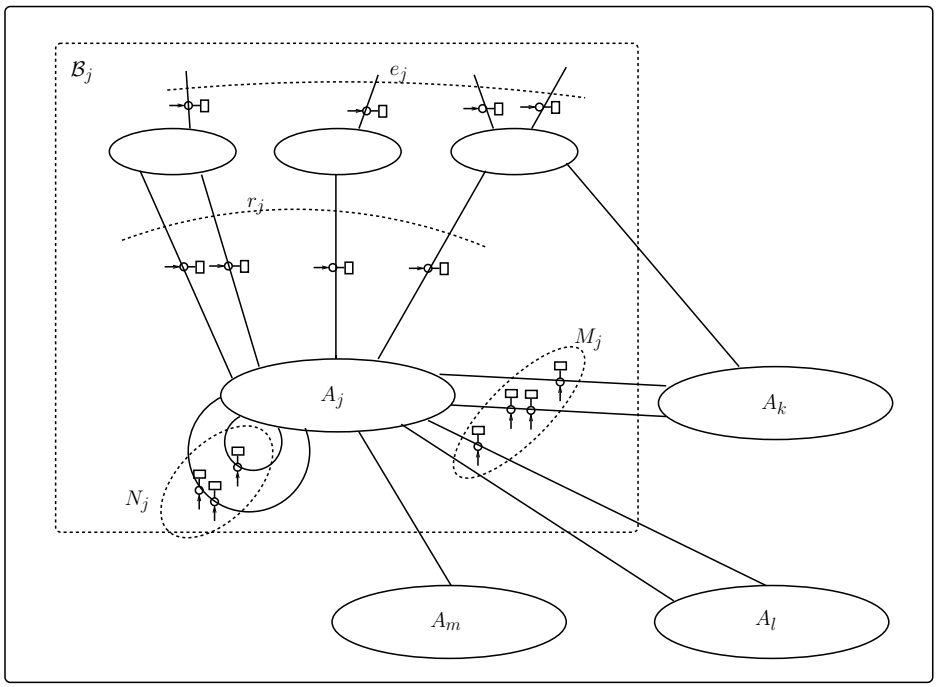


Figure 5: Sketch of proof of Theorem 5

We introduce the sets $\mathcal{B}_j$ consisting of A_j union all the small $H-$components adjacent to it. We define new quantities as shown on Figure 5:

- r_j is the number of edges between A_j and its small H-components.
- e_j is the number of outgoing edges of $\mathcal{B}_j$ incident to one of its small components.

- M_j is the number of doublons on the edges of type M.
- N_j is the number of doublons on the edges of type N.

As two small components can not be adjacent, remark that all r_j and e_j edges are of type R.

Let us now apply the cut criterion to $\mathcal{B}_j$. $\delta(\mathcal{B}_j) + o(\mathcal{B}_j) - \min(k, o(\mathcal{B}_j)) - \min(i(\mathcal{B}_j), p) \geq 0$. As $k \leq \frac{n}{2}$, we have $i(\mathcal{B}_j) \leq p$. We first show that $o(\mathcal{B}_j) < k$. Indeed, if $o(\mathcal{B}_j) \geq k$, the cut criterion reduces to $\delta(\mathcal{B}_j) \geq k$. Furthermore A_j is connected and of size less than q, so it has at most $2q+2$ outgoing edges and the number of small $H-$components of $\mathcal{B}_j$ is at most $2q+2$. As the size of a small $H-$component is less than q, the number of vertices in $\mathcal{B}_j$ is at most $q + (2q+2)q$. Hence the number of outgoing edges $\delta(\mathcal{B}_j)$ is at most $2(q + (2q+2)q) + 2$. By our choice of q it gives $k \leq \delta(\mathcal{B}_j) \leq 2(q + (2q+2)q) + 2 \leq k - 1$, a contradiction. Consequently $o(\mathcal{B}_j) < k$.

The cut criterion is now equivalent to $\delta(\mathcal{B}_j) \geq i(\mathcal{B}_j)$. Noting δ' the number of outgoing edges of $\mathcal{B}_j$ incident to A_j, we have $\delta(\mathcal{B}_j) = \delta'(\mathcal{B}_j) + e_j$. Using the definitions of switch kinds (Figure 4), we have

$$\delta' = 4|A_j| - 2e(A_j) - r_j - 2N_j - s_i - s_o - 2v_i - 2v_o - 3t.$$

Since A_j is connected, $e(A_j) \geq |A_j| - 1$, we get

$$\delta' \leq 2|A_j| + 2 - r_j - 2N_j - s_i - s_o - 2v_i - 2v_o - 3t.$$

For the number of inputs in $\mathcal{B}_j$ we have

$$i(\mathcal{B}_j) = e_j + r_j + M_j + N_j + s_i + 2v_i + t.$$

The cut criterion (Proposition 1) then implies:

$$2|A_j| + 2 \geq 2s_i + s_o + 4v_i + 2v_o + 4t + 3N_j + 2r_j + M_j. \tag{7}$$

The total number of doublons is d. All doublons are of type R, M or N. The doublons in M are counted for two different A_j and A_k. So $\sum_{j=1}^{m} M_j + 3N_j + 2r_j \geq 2d$. Hence taking the sum of Equation 7 over all j we obtain

$$2\sum_{j=1}^{m} |A_j| + 2m \geq 2s_i + s_o + 4v_i + 2v_o + 4t + 2d.$$

The input and output equations (Equations 5 and 6) give $2s_i + 4v_i + 2t + 2d = 2n - 2(k - \lambda)$ and $s_o + 2v_o + 2t = n - d$. Hence

$$2\sum_{j=1}^{m} |A_j| + 2m \geq 3n - d - 2(k - \lambda). \tag{8}$$

The family $\{A_j\}$ forms a q-quasi-partition of $\mathcal{R}$. Let $t := |\{A_j, |A_j| \geq \frac{2q}{3}\}|$ and v the number of vertices of $\mathcal{R}$. Then, by Remark 3, we have $m \leq \frac{3(v+t+1)}{q}$ and $t \leq \frac{v+1}{\frac{2q}{3}-1}$. By definition of a quasi-partition: $\sum_{i=j}^{m} |A_j| \leq v + |\{A_j, |A_j| \geq \frac{2q}{3}\}| = v + t + 1$.

Putting all these equations into Equation 8 gives:

$$2v\left(1 + \frac{1}{\frac{2q}{3} - 1}\right) \geq \frac{q}{q+3}[(3n - d - 2(k - \lambda)) - \frac{2(q+3)}{q} - \frac{6}{q} - 2].$$

$$2v \geq \frac{q}{q+3}\frac{2q-3}{2q}[3n-d-2(k-\lambda)-\frac{2(q+3)}{q}-\frac{6}{q}-2]$$
$$= [3n-d-2(k-\lambda)-\frac{2(q+3)}{q}-\frac{6}{q}-2]\left(1-\frac{9}{2q+6}\right).$$

Using $N \geq v+d$, we obtain

$$N \geq \left(\frac{3}{2}n-(k-\lambda)-4-\frac{12}{q}\right)\left(1-\frac{9}{2q+6}\right)+\frac{d}{2}\left(1+\frac{9}{2q+6}\right).$$

Finally,

$$N \geq \left(\frac{3}{2}n-(k-\lambda)-4-\frac{12}{\sqrt{k}}\right)\left(1-\frac{9}{2\sqrt{k}}+O\left(\frac{1}{k}\right)\right)$$
$$+\frac{d}{2}\left(1+\frac{9}{2\sqrt{k}}+O\left(\frac{1}{k}\right)\right).$$

□

6.2 Lower Bounds

Here we apply Theorem 5 to prove lower bounds for the number of switches of minimal valid networks. Let us remind the reader that $l \leq k$ is assumed for symmetry reasons and that $n := p+k$.

Theorem 6 (Selector: $\lambda = 0$) *In a valid network $\mathcal{N}$, when $k \to \infty$ with $k \leq \frac{n}{2}$, we have*

$$N \geq n+\frac{n}{2}+\mathcal{O}(\frac{n}{\sqrt{k}}).$$

In other words,

$$N(p,0,k) \geq n+\frac{n}{2}+O(\frac{n}{\sqrt{k}}).$$

In particular, we obtain a tight bound for networks with $\lambda = 0$ (see Theorem 3).

Proof The proof follows directly from Theorem 5. □

Theorem 7 (Simplified case) *In the simplified case, when $k \to \infty$ with $k \leq \frac{n}{2}$, we have $N(p,\lambda,k) \geq 2n+O(\frac{n}{\sqrt{k}})$.*

In particular, we obtain a tight bound for the simplified case (see Theorem 1).

Proof First recall that a direct application of the cut criterion shows that as soon as $\lambda \geq 1$, T is empty (see definition of switch types in Figure 4, p. 18). This means that in this proof and the following we will not consider switches of T. Now the proof follows from Theorem 5 and from $d = n-(k-\lambda)$ in the simplified case. □

Theorem 8 (General case) *When $\lambda \to \infty$ and $k \to \infty$, $N(p,\lambda,k) \geq n+\frac{2}{3}n+O(\frac{n}{\sqrt{\lambda}})$.*

Proof We first give a bound for the number of switches of types V_i and V_o using the following remark:

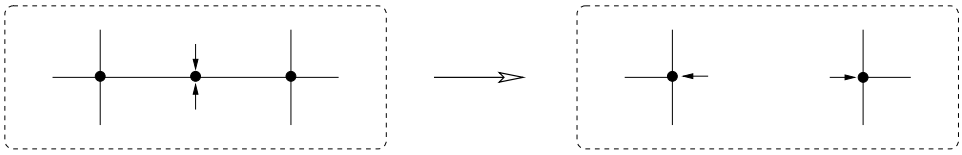


Figure 6: When $\lambda = 0$, switches of kind V_i may be removed.

Lemma 5 *When $\lambda \to \infty$ and $k \to \infty$,*

$$\begin{cases} v_i & \leq & N - \frac{3}{2}n - \frac{d}{2} - \frac{k-\lambda}{2} + \lambda O(\frac{n}{\sqrt{k}}), \\ v_o & \leq & N - \frac{3}{2}(n-k+\lambda) - \frac{d}{2} + k - \frac{\lambda - k}{2} + O(\frac{n}{\sqrt{\lambda}}). \end{cases}$$

Proof Imagine we have a valid $(p, \lambda, k)-$network with N switches and v_i switches in V_i. We obtain a valid $(p, 0, k)-$network after removing any λ inputs. This new network has at least $v_i - \lambda$ switches of kind V_i. By Remark 4 we may remove these switches and obtain a valid $(p, 0, k)-$network with $N - v_i + \lambda$ switches. Theorem 5 gives

$$N - v_i + \lambda \geq \frac{3}{2}n + \frac{d}{2} + \frac{k-\lambda}{2} + O(\frac{n}{\sqrt{k}}).$$

So the result holds. Symmetry (in the sense of swapping inputs and outputs gives a valid $(p, k, \lambda)-$network) gives the second equation. □

The switch partition equation and Equations 5 and 6 give here

$$N = 2n - (k - \lambda) - v_i - v_o - d + s.$$

Lemma 5 gives

$$\begin{aligned} N &\geq 2n - (k-\lambda) - d + s - 2N + 3n + d \\ &\quad + O(\frac{n}{\sqrt{k}}) + O(\frac{n}{\sqrt{\lambda}}) - \frac{3}{2}(k-\lambda) - \lambda - k. \\ N &\geq \frac{5}{3}n + O(\frac{n}{\sqrt{k}}) + O(\frac{n}{\sqrt{\lambda}}) - \frac{5}{2}k + \frac{\lambda}{2}. \end{aligned}$$

□

Theorem 9 *When $\lambda \geq 1$ and no switches of kinds V_i and V_o are allowed, when $k \to \infty$ with $k \leq \frac{n}{2}$, we have $N(p, \lambda, k) \geq n + \frac{2}{3}n + O(\frac{n}{\sqrt{k}})$.*

Proof Let us first remark the following:

Remark 4 Notice that when $\lambda = 0$, switches of type V_i are not present in a minimal valid $(p, \lambda, k)-$network. As shown in Figure 6, they may be removed to form a new valid $(p, \lambda, k)-$network with v_i less switches.

Proof Direct by the cut criterion. □

When $v_i = v_o = 0$, the input equation (Equation 5) becomes $n = d + s_i$ and the output equation (Equation 6) becomes $n = d + s_o$. So $s_i = s_o$ and the switch partition equation (Equation 4) gives

$$N = 2n - d. \tag{9}$$

Theorem 5 gives

$$\begin{aligned} 2n - d \geq & \ \frac{3}{2}n + \frac{d}{2} + O(\frac{n}{\sqrt{k}}) \\ \frac{n}{3} \geq & \ d + O(\frac{n}{\sqrt{k}}). \end{aligned}$$

Equation 9 gives

$$N \geq \ \frac{5}{3}n + O(\frac{n}{\sqrt{k}}).$$

□

To sum up, in the general case, we proved a lower bound of type $n + 2n/3$. Let us remind the reader that we obtained an upper bound with $n + 3n/4$ switches (Theorem 2). We conjecture that the later is the right answer to the problem. We are able to prove this with some extra conditions on the network using majority arguments:

Theorem 10 *Let $\mathcal{N}$ be a network of N switches with the associated graph $\mathcal{R} = (V = A \cup B, E)$ such that $\mathcal{R}$ is bipartite, all vertices of A have exactly one output and all vertices of B exactly one input in the original network.*

$$N \geq n + \frac{3}{4}n + O(\frac{n}{\sqrt{k}}).$$

Proof We are given a ternary bipartite graph $R = (V = A \cup B, E)$ such that all vertices of A have an output and all vertices of B have an input. The vertices of B are partitioned in two sets B_1 and B_0. A vertex of B_1 is adjacent to an edge of type E_1 or E_2, while the vertices of B_0 are not.

For a subset $X \subset A$ we use the following notations : $B_i(X) = \Gamma(X) \cap B_i, i = 0, 1$ and $b_i = |B_i|$.

Let X be a subset of A such that $F(X) = X \cup \Gamma(X)$ is connected and $6|X| \leq k$. X has less than k outputs. To fullfill the cut criterion for small subsets (less than k outputs), we need

$$b_0(X) \geq b_1(X) - 3 + 3c_{F(X)}, \tag{10}$$

where $c_{F(X)}$ is the feedback edge set of the subgraph induced by $F(X)$.

The goal is to prove that this can happen only if $b_0 + O(\frac{1}{\sqrt{k}}) \geq 2b_1$.

Let Y be a set of y vertices of B_0 such that $Y \cup \Gamma(Y)$ is connected. We have $|\Gamma(Y)| = 2y + 1 - c_{F(Y)}$. There are $3y + 3 - 3c$ edges comming out of $F(Y)$, say αy toward vertices of type B_0 and $3y + 3 - 3c - \alpha y$ toward vertices of type B_1. In G we consider the connected subgraph induced by $Z = F(Y) \cup (\Gamma(F(Y)) \cap B_1)$. We have $(6 - \alpha)y + 3 - 3c_{F(Y)}$ edges inside Z, so the number of vertices in $\Gamma(F(Y)) \cap B_1$ is

$(6-\alpha)y+3-3c_{F(Y)}-y-(2y+1-c_{F(Y)})-c_Z=(3-\alpha)y+2-2c_{F(Y)}-c_Z$. If we take $X=\Gamma(Y)$, Equation 10 gives

$$\begin{array}{rcl}(\alpha+1)y & \geq & b_0(X)\geq b_1(X)-3+3c_{F(X)}\\ & \geq & (3-\alpha)y+2-2c_{F(Y)}-c_Z-3+3c_{F(X)}\\ & \geq & (3-\alpha)y-1-2c_{F(Y)}+2c_{F(X)}\end{array}$$

where in the last inequality we use the fact that $c_Z\leq c_{F(X)}$. As $c_{F(Y)}\leq c_{F(X)}$, we have:

$$\alpha\geq 1-\frac{1}{2y}.$$

We consider all the connected components of the graph induced by $B_0\cup A$ and we take a q-quasi-partition of the big components for some q, $q=O(k)$ such that all components are of the form $F(Y)$ for some Y subset of B_0. Now we count the edges going to B_0 and B_1.

For one component D of the quasi-partition with y vertices of B_0, we find at least $(3+\alpha)y\geq 4y-\frac{1}{2}$ edges toward B_0 and at most $2y+\frac{7}{2}$ edges toward B_1 with one extremity in D.

Globally, if m is the number of components and up to some small number of recounting (Quasi-partition arguments) we get $4|A|-\frac{m}{2}=3b_0$ edges toward B_0 and $2A+\frac{7m}{2}=3b_1$ edges toward B_1.

Hence $b_0\geq 2b_1+O(\frac{1}{\sqrt{k}})$. □

7 Conclusion

In this paper we proposed constructions of valid $(p,\lambda,k)-$networks and gave lower bounds on their size. The design problem appears to be driven by two constraints: a local one in which small patterns are forbidden and another one which is related to some global expansion property of the network. This led us to define an expansion parameter of a graph: the $\alpha-$robustness. This parameter is a generalization of the usual edge-expansion. Using graphs of $2-$robustness equal to $\Theta(\log n)$ we constructed almost optimal simplified networks. Similarly when $k\leq\frac{n}{7}$, using graphs of large $1-$robustness we proposed good simplified networks. Despite many answers for small values of k ($k\leq\frac{n}{7}$), little is known when k is larger.
To obtain our lower bounds, we presented a powerful technique: quasi-partitioning. We think that this technique can potentially have other applications to derive lower bounds for problems of the same kind.

Some interesting open questions remain

- For a fixed α, find explicit constructions of nice α-robust graphs, graphs having large $\alpha-$robustness.

- Is there a very explicit construction such as the "Zig-Zag product"of [29] and "lift"of [13] providing nice robust graphs? Does the Zig-Zag product preserve robustness?

- What is the maximum bisection-width of a 4-regular graph? In [24] the maximal bisection width of 4-regular graphs on n vertices is shown to be at most $\frac{2n}{5}$. This leads to an $\alpha-$robustness $<\frac{n}{2}$ when $\alpha>\frac{4}{5}$. Is it possible to

obtain the same kinds of results for larger values of α ensuring a maximal robustness of size at most $\frac{n}{3}$, for example?

- What is the minimum number of switches or edges in an *n-superselector*? (See [23, 30] for some constructions.)

8 Acknowledgments

We are grateful to the anonymous referees for their valuable remarks which helped us to improve the presentation of our results. We especially thank Networks' Editor-in-Chief Douglas Shier for his helpful comments on earlier versions of this paper. We also want to acknowledge Jean-Claude Bermond and Frédéric Havet for their interest in our work.

References

[1] N. Alon, On the edge-expansion of graphs, Combin, Probability Comput 11 (1993), 1–10.

[2] N. Alon and M. Capalbo, Smaller explicit superconcentrators, Proc 14th Ann Symp Discret Algorithms ACM-SIAM SODA, 2003, pp. 340–346.

[3] N. Alon, Z. Galil, and V.D. Milman, Better expanders and superconcentrators, J Algorithms 8 (1987), 337–347.

[4] N. Alon, P. Hamburger, and A.V. Kostochka, Regular honest graphs, isoperimetric numbers, and bisection of weighted graphs, Eur J Combin 20 (1999), 469–481.

[5] N. Alon and V.D. Milman, Eigenvalues, expanders and superconcentrators, Proc 25th Ann Symp Fondations Comput Sci (FOCS), Singer Island, Florida, 1984, pp. 320–322.

[6] L.A. Bassalygo, Asymptotically optimal switching circuits, Problemy Pederachi Informatsii 17 (1981), 81–88.

[7] B. Beauquier and E. Darrot, Arbitrary size Waksman networks, Première rencontres francophones sur les aspects algorithmiques de télécommunication (Algotel), 1999, pp. 95–100.

[8] B. Beauquier and E. Darrot, Arbitrary size Waksman networks and their vulnerability, Parallel Process Lett 3-4 (2002), 287–296.

[9] J.C. Bermond, E. Darrot, and O. Delmas, Design of fault tolerant on-board networks in satellites, Networks 40 (2002), 202–207.

[10] J.C. Bermond, O. Delmas, F. Havet, M. Montassier, and S. Perennes, Réseaux de télécommunications minimaux embarqués tolérants aux pannes, Cinquième rencontres francophones sur les aspects algorithmiques de télécommunication (Algotel), 2003, pp. 27–32.

[11] J.C. Bermond, F. Giroire, and S. Pérennes, Design of minimal fault tolerant on-board networks : Practical constructions, 14th Int Colloq Structural Informat Commun Complexity (SIROCCO), Lecture Notes in Comput Sci, Springer, 2007, Number 4474 pp. 261–273.

[12] J.C. Bermond, F. Havet, and C.D. Tóth, Fault tolerant on-board networks with priorities, Networks 47 (2006), 9–25.

[13] Y. Bilu and N. Linial, Lifts, discrepancy and nearly optimal spectral gaps, Combinatorica 26 (2006), 495–519.

[14] M. Blum, R. Karp, C. Papadimitriou, O. Vornberger, and M. Yannakakis, The complexity of testing whether a graph is a superconcentrator, Informat Process Lett 13 (1981), 164–167.

[15] B. Bollobás, The isoperimetric number of random regular graphs, Eur J Combin 9, 3 (1988), 241–244.

[16] M. Capalbo, O. Reingold, S. Vadhan, and A. Wigderson, Randomness conductors and constant-degree lossless expanders, 34th ACM Symp Theory Comput (STOCS), 2002, pp. 659–668.

[17] F.R.K. Chung, On concentrators, superconcentrators, generalizers and nonblocking networks, Bell System Tech J 58 (1979), 1765–1777.

[18] F.R.K. Chung, Spectral Graph Theory, American Mathematical Society, 1997.

[19] G. Davidoff, P. Sarnak, and A. Valette, Elementary number theory, group theory, and Ramanujan graphs, Cambridge University Press, 2003.

[20] O. Delmas, F. Havet, M. Montassier, and S. Pérennes, Design of fault tolerant on-board networks, INRIA Res Report 5866, submitted (2006).

[21] F. Giroire, Réseaux, algorithmique et analyse combinatoire de grands ensembles, Ph.D. Thesis, Université Paris VI, November 2006.

[22] C. Greenhill, J.H. Kim, and N.C. Wormald, Hamiltonian decompositions of random bipartite regular graphs, J Combinatorial Theory Ser B 90 (2004), 195–222.

[23] F. Havet, Repartitors, selectors and superselectors, J Interconnection Networks 7 (2006), 391–415.

[24] B. Monien and R. Preis, Upper bounds on the bisection width of 3-and 4-regular graphs, J Discr Algorithms 4 (2006), 475–498.

[25] M. Morgenstern, Existence and explicit constructions of $q + 1$ regular Ramanujan graphs for every prime power q, J Combinatorial Theory Ser B 62 (1994), 44–62.

[26] M. Murty, Ramanujan graphs, J Ramanujan Math Soc 18 (2003), 33–52.

[27] H.Q. Ngo and D. Du, *Notes on the complexity of switching networks*, Advances in Switching Networks, Kluwer Academic Publishers, Dordrecht, The Netherlands, (2001), pp. 307–367.

[28] N. Pippenger, Superconcentrators, SIAM J Comput 6 (1977), 298–304.

[29] O. Reingold, S. Vadhan, and A. Wigderson, Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors, Ann Math 155 (2002), 157–187.

[30] U. Schöning, Smaller superconcentrators of density 28, Informat Process Lett 98 (2006), 127–129.

[31] L.G. Valiant, On non-linear lower bounds in computational complexity, 7th ACM Symp Theory Comput (STOCS), 1975, pp. 45–53.

Annexe B

Algorithmes de routage

B.1 (ℓ, k)-Routing on Plane Grids

Article soumis à Journal of Interconnectio Networks.

B.2 On Multiple Improper Colorings

Article en préparation.

B.3 The Proportional Colouring Problem : Optimising Buffers in Radio Mesh Networks

Article soumis à un numéro spécial de Discrete Applied Mathematics.

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

(ℓ, k)-*Routage dans les Grilles Planaires*

Omid Amini — Florian Huc — Ignasi Sau — Janez Žerovnik

N° 6480

February 2008

Thème COM

ISSN 0249-6399 ISRN INRIA/RR--6480--FR+ENG

INRIA
SOPHIA ANTIPOLIS

(ℓ, k)-Routage dans les Grilles Planaires *

Omid Amini †, Florian Huc ‡, Ignasi Sau ‡ §, Janez Žerovnik ¶ ‖

Thème COM — Systèmes communicants
Projet MASCOTTE

Rapport de recherche n° 6480 — February 2008 — 33 pages

Résumé : Le problème de routage de paquets joue un rôle très important dans les réseaux de télécommunication. Il consiste à envoyer des données entre les sommets d'un réseau en un temps raisonable. Dans le (ℓ, k)-routage, chaque sommet peut envoyer au plus ℓ paquets et en recevoir k. Le routage de permutation correspond au cas $\ell = k = 1$. Dans le routage r-central, tous les noeuds à distance au plus r d'un sommet fixé envoient un paquet à ce sommet. Dans cet article nous étudions les problèmes précédents dans les grilles planaires. On utilise le modèle *store-and-forward* et Δ-port. Nous considérons les réseaux half et full-duplex. Les résultats principaux sont :

1. un algorithme *tight* pour le problème de routage de permutation dans les grilles hexagonales full-duplex, et dans les grilles hexagonales et triangulaires half-duplex.
2. un algorithme *tight* pour le routage r-central dans les grilles triangulaires et hexagonales.
3. un algorithme *tight* pour le (ℓ, ℓ)-routage dans les grilles carrées, triangulaires et hexagonales.
4. de bon algorithme d'approximation (en terme de temps de calcul) pour le (ℓ, k)-routage dans les grilles carrées, triangulaires et hexagonales. On donne aussi de nouvelles bornes inférieures sur le temps d'éxécution d'un algorithme qui utilise le routage par le plus court chemin.

* This work has been partially supported by European project IST FET AEOLUS, PACA region of France, Ministerio de Educación y Ciencia of Spain, European Regional Development Fund under project TEC2005-03575, Catalan Research Council under project 2005SGR00256, and COST action 293 GRAAL, and has been done in the context of the CRC Corso with France Telecom.

† Max-Planck-Institut für Informatik, Saarbrücken, Germany
‡ Mascotte Project - I3S (CNRS/UNSA) and INRIA - Sophia-Antipolis, France
§ Graph Theory and Combinatorics group, MA4, UPC, Barcelona, Spain
¶ University of Ljubljana Faculty of Mathematics and Physics (IMFM), jubljana, Slovenia
‖ University of Maribor, Maribor, Slovenia

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Tous ces algorithmes sont complètements distribués.

Mots-clés : Routage de paquets, algorithme distribué, (ℓ, k)-routage, grilles planaires, routage de permutation, plus court chemin, algorithme sans mémoire

(ℓ, k)-Routing on Plane Grids **

Abstract: The packet routing problem plays an essential role in communication networks. It involves how to transfer data from some origins to some destinations within a reasonable amount of time. In the (ℓ, k)-routing problem, each node can send at most ℓ packets and receive at most k packets. Permutation routing is the particular case $\ell = k = 1$. In the r-central routing problem, all nodes at distance at most r from a fixed node v want to send a packet to v.

In this article we study the permutation routing, the r-central routing and the general (ℓ, k)-routing problems on plane grids, that is square grids, triangular grids and hexagonal grids. We use the *store-and-forward* Δ-port model, and we consider both full and half-duplex networks. The main contributions are the following:

1. Tight permutation routing algorithms on full-duplex hexagonal grids, and half duplex triangular and hexagonal grids.
2. Tight r-central routing algorithms on triangular and hexagonal grids.
3. Tight (k, k)-routing algorithms on square, triangular and hexagonal grids.
4. Good approximation algorithms (in terms of running time) for (ℓ, k)-routing on square, triangular and hexagonal grids, together with new lower bounds on the running time of any algorithm using shortest path routing.

All these algorithms are completely distributed, i.e. can be implemented independently at each node. Finally, we also formulate the (ℓ, k)-routing problem as a WEIGHTED EDGE COLORING problem on bipartite graphs.

Key-words: Packet routing, distributed algorithm, (ℓ, k)-routing, plane grids, permutation routing, shortest path, oblivious algorithm

1 Introduction

In telecommunication networks, it is essential to be able to route communications as quickly as possible. In this context, the *packet routing* problem plays a capital role. In this problem we are given a network and a set of packets to be routed through the nodes and the edges of the network graph. A packet is characterized by an origin and a destination node, and typically an edge can be used by no more than one packet at the same time. The objective is to find an algorithm to compute a schedule to route all packets which minimizes the total delivery time. This problem has been widely studied in the literature under many different assumptions. In 1988, Leighton, Maggs and Rao proved in their seminal article [31, 29] the existence of a schedule for routing any set of packets with edge-simple paths on a general network, in optimal time of $\mathcal{O}(C + D)$ steps. Here C is the congestion (maximum number of paths sharing an edge) and D the dilation (length of a longest path) and it is assumed that the paths are given a priori. The proof of [29] used Lovász Local Lemma and was non constructive. This result was further improved in [28] where the same authors gave an explicit algorithm, using the Beck's constructive version of the Local Lemma.

These algorithms to compute the optimal schedule are centralized. Then in [38] Ostrovsky and Rabani gave a distributed randomized algorithm running in $\mathcal{O}(C+D+\log^{1+\epsilon}(n))$ steps. We give a more detailed overview in Section 1.1.

Although these results are asymptotically tight, they deal with a general network, and in many cases it is possible to design more efficient algorithms by looking at specific packet configurations or network topologies. For instance, is it natural to bound the maximum number of messages that a node can send or receive. We focus on this point in Section 1.2, where we will formally define the problem under study in this paper.

On the other hand, the network under study plays a major role on the quality and the simplicity of the solution. For example, in a radio wireless environment, cellular networks are usually modeled by a *hexagonal grid* where nodes represent base stations. The cells of the hexagonal grids have good diameter to area ratio and still have a simple structure. If centers of neighboring cells are connected, the resulting graph is called a *triangular grid*. Notice that hexagonal grids are subgraphs of the triangular grid. We will talk about such networks in Section 1.3. In this paper we focus on the study of the (ℓ, k)-routing problem in convex subgraphs, i.e. that contain all shortest paths between all pairs of nodes of the square, triangular and hexagonal grid.

1.1 General Results on Packet Routing

In this section we provide a fast overview of the state-of-the-art of the general packet routing problem, in both the off-line and on-line settings in Sections 1.1.2 and 1.1.3 respectively, focusing mostly on the latter. We begin by recalling three classical lower bounds for the packet routing problem.

1.1.1 Classical lower bounds

In the packet routing problem, there are three classical types of lower bounds for the running time of any algorithm :

1. **Distance bound** : the longest distance over the paths of all packets (usually called *dilation*, denoted D) constitutes a lower bound on the number of steps required to route all the packets.
2. **Congestion bound** : the *congestion* of an edge of the network is defined as the number of paths using this edge. Then, the greatest congestion over all the edges of the network (denoted C) is also a lower bound on the number of steps, since at each step an edge can be used by at most one packet.
3. **Bisection bound** : Let $G = (V, E)$ be the graph which models the network, and $F \subseteq E$ a cut-set disconnecting G into two components G_1 and G_2. Let m be the number of packets with origin in G_1 and destination in G_2. Then, the number of routing steps used by any algorithm will be at least $\left\lceil \frac{m}{|F|} \right\rceil$.

1.1.2 Off-line routing

Given a set of packets to be sent through a network, a *path system* is defined as the union of the paths that each packet must follow. For a general network and any set of n demands, we have seen in Section 1.1.1 that the dilation and the congestion provide two lower bounds for the routing time. This proves that the *dilation* + *congestion* of a paths system used for the routing procedure is a lower bound of twice the routing time. In a celebrated paper, Leighton, Maggs and Rao proved the following theorem :

Theorem 1.1 ([31]) *For any set of requests and a path system for these requests, there is an* **off-line** *routing protocol that needs $\mathcal{O}(C + D)$ steps to route all the requests, where C is the congestion and D is the dilation of the path system.*

In addition, in [49] the authors show that, given the set of packets to be sent, it is possible to find in polynomial time a path system with $C + D$ within a factor 4 of the optimum. Thus, Theorem 1.1 can be announced in a more general way :

Theorem 1.2 ([49]) *For any set of requests, there is an* **off-line** *routing protocol that needs $\mathcal{O}(C + D)$ steps to route all the requests, where $C + D$ is the minimum congestion + dilation over all the possible path systems.*

Furthermore, this routing protocol uses fixed buffer size, i.e. the queue size at all nodes is bounded by a constant at each step. Nevertheless, it is important to notice that a huge constant may be hidden inside the $\mathcal{O}$ notation. As we have said in the introduction, this result was further improved in [28] where the same authors gave an explicit algorithm. These algorithms to compute the optimal schedule are centralized. In a distributed algorithm nodes must make their decisions independently, based on the packets they see, without the use of

a centralized scheduler. Then in [38] Ostrovsky and Rabani gave a distributed randomized algorithm running in $\mathcal{O}(C + D + \log^{1+\epsilon}(n))$ steps.

We refer to Scheideler's thesis [45] for a complete compilation of general packet routing algorithms.

1.1.3 On-line routing

In the on-line setting, the oldest on-line protocol that deviates only by a factor logarithmic in n from the best possible runtime $\mathcal{O}(C + D)$ for arbitrary path-collections is the protocol presented by Leighton, Maggs and Rao in the same paper [31], running in $\mathcal{O}(C + D \log(Dn))$ steps with high probability. The authors call the algorithm on-line, rather than distributed. This schedule assumes that the paths are given a priori, hence it does not focus on the problem of choosing the paths to route the packets.

The results of [1] provide a routing algorithm that is $\log n$ competitive with respect to the congestion. In other words, it is worse than an optimal off-line algorithm only by a factor $\log n$. In this setting the demands arrive one by one and the algorithm routes calls based on the current congestion on the various links in the network, so this can be achieved only via centralized control and serializing the routing requests. In [3] the authors gave a distributed algorithm that repeatedly scans the network so as to choose the routes. This algorithm requires shared variables on the edges of the network and hence is hard to implement. Note that the two on-line algorithms above depend on the demands and are therefore *adaptive*. Recall that an *oblivious* routing strategy is specified by a path system $\mathcal{P}$ and a function w assigning a weight to every path in $\mathcal{P}$. This function w has the property that for every source-destination pair (s, t), the system of flow paths $\mathcal{P}_{s,t}$ for (s, t) fulfills $\sum_{q \in \mathcal{P}_{s,t}} w(q) = 1$. One can think of this function as a frequency distribution among several paths going from an origin s to a destination t. In *adaptive* routing, however, the path taken by a packet may also depend on other packets or events taking place in the network during its travel. Remark that every oblivious routing strategy is obviously on-line and distributed.

The first paper to perform a worst case theoretical analysis on oblivious routing is the paper of Valiant and Brebner [54], who considered routing on specific network topologies such as the hypercube. They give a randomized oblivious routing algorithm. Borodin and Hopcroft [6] and subsequently [22] have shown that deterministic oblivious routing algorithms cannot approximate well the minimal load on any non-trivial network.

In a recent paper, Räcke [40] gave the construction of a polylog competitive oblivious routing algorithm for general undirected networks. It seems truly surprising that one can come close to minimal congestion without any information on the current load in the network. This result has been improved in [4]. Lower bounds on the competitive ratio of oblivious routing have been studied for various types of networks. For example, for the d-dimensional mesh, Maggs et al. [35] gave the $\omega(\frac{C_*}{d}(\log n))$ lower bound on the competitive ratio of an oblivious algorithm on the mesh, where C_* is the optimal congestion.

So far, the oblivious algorithms studied in the literature have focused on minimizing the congestion while ignoring the dilation. In fact, the quality of the paths should be determined by the congestion C, and the dilation D. An open question is whether C and D can be

controlled simultaneously. An appropriate parameter to capture how good is the dilation of a path system is the *stretch*, defined as the maximum over all packets of the ratio between the length of the path taken by the routing protocol and the length of a shortest path from source to destination. In a recent work, Bush et al. [8] considered again the case of the d-dimensional mesh. They presented an on-line algorithm for which C and D are both within $\mathcal{O}(d^2)$ of the potential optimal, i.e. $D = \mathcal{O}(d^2 D_*)$ and $C = \mathcal{O}(dC_* \log(n))$, where D_* is the optimal dilation (remark that by [35], it is impossible to have a factor better than $\omega(\frac{C_*}{d}(\log n))$).

There is a simple counter-example network that shows that in general the two metrics (*dilation* and *congestion*) are orthogonal to each other : take an adjacent pair of nodes u, v and $\Theta(\sqrt{n})$ disjoint paths of length $\Theta(\sqrt{n})$ between u and v. For packets traveling from u to v, any routing algorithm that minimizes congestion has to use all the paths, however, in this way some packets follow long paths, giving high stretch. Nevertheless, in grids [8], and in some special kind of geometric networks [7] the congestion is within a poly-logarithmic factor from optimal and stretch is constant (d the dimension). As mentioned before an interesting open problem is to find other classes of networks where the congestion and stretch are minimized simultaneously [2]. Possible candidates for such networks could be for example bounded-growth networks, or networks whose nodes are uniformly distributed in closed polygons, which describe interesting cases of wireless networks.

The recent paper of Maggs [34] surveys a collection of theoretical results that relate the congestion and dilation of the paths taken by a set of packets in a network to the time required for their delivery.

1.2 Routing Problems

The initial and final positioning of the packets has a direct influence on the time needed for their routing. Considering static packet configuration, the most studied constraints refer to the maximum number of packets that a node can send and receive. Due to their practical importance, some of these problems have specific names :

1. **Permutation routing** : each node is the origin and the destination of at most one packet. To measure the routing capability of an interconnection network, the partial permutation routing (PPR) problem is usually used as the metric.
2. (ℓ, k)**-routing :** each node is the origin of at most ℓ packets and destination of at most k packets. Permutation routing corresponds to the case $\ell = k = 1$ of (ℓ, k)-routing. Another important particular case is the $(1, k)$**-routing**, in which each node can send at most one packet and receive at most k packets.
3. $(1, any)$**-routing** : each node is origin of at most one packet but there are no constraints on the number of packets that a node can receive.
4. r**-central routing** : all nodes at distance at most r of a central node send one message to this central node.

In all these problems we are given an initial packet configuration, and the objective is to route all packets to their respective destinations minimizing the total routing time, under the constraint that each edge can be used by at most one packet at the same time.

Besides of the constraints about the initial and final positions of the packets, there also exist different routing models at the intermediate nodes of the network. For instance, in the *hot potato model* no packet can be stored at the nodes of the network, whereas in the *store-and-forward* at each step a packet can either stay at a node or move to an adjacent node. Another widely used model is the *wormhole* routing.

On the other hand, one can consider constraints on the number of incident edges that each node of the network can use to send or receive packets at the same time. In the Δ-port model [16], each node can send or receive packets through all its incident edges at the same time.

In this article we study the store-and-forward Δ-port model. In addition, we suppose that cohabitation of multiple packets at the same node is allowed. I.e. a queue is required for each outgoing edge at each node.

The nature of the links of the network is another factor that influences the routing efficiency. The type of links is usually one of the following : full-duplex or half-duplex. In the full-duplex case there are two links between two adjacent nodes, one in each direction. Hence two packets can transit, one in each direction, simultaneously. In the half-duplex case only one packet can transit between two nodes, either in one direction of the edge or in the other. In this paper we focus on both half and full-duplex links.

1.3 Topologies

We now give a brief summary of various cases of (ℓ, k)-routing and $(1, any)$-routing that have been studied for several specific topologies. More precisely, in Section 1.3.1 we list the most important results for some networks which have attracted a great interest in the literature, like hypercubes and circulant graphs. Then we move to plane grids in Section 1.3.2. It is well known that there exist only three possible tessellations of the plane into regular polygons [55] : squares, triangles and hexagons. These graphs are those which we study in this article.

1.3.1 Different network topologies

In [20] the authors studied the permutation routing problem in low-dimensional hypercubes ($d \leq 12$). They gave optimal or good in the worst case oblivious algorithms, i.e. algorithms in which the path used by a packet is entirely determined by its origin and its destination. An other network widely studied in the literature is the two dimensional mesh with row and column buses. This network can also be diversified according to the capacities of the buses. In [51] Suel gave a deterministic algorithm to solve the permutation routing problem in such networks. It gives a schedule using at most $n + o(n)$ steps and a queue of size 2, where the queue is the maximum number of packets that have to be stored at an intermediate node. He also proposed a deterministic algorithm for r-dimensional arrays with

buses working in $(2-\frac{1}{r})n+o(n)$ steps and still using queues of size 2. In [27], the authors studied the (ℓ,ℓ)-routing problem in the mesh grid with two diagonals and gave, for $\ell \geq 9$ a deterministic algorithm using $\frac{2\ell n}{9}+(\ell n^{2/3})$ steps.

In [19], the authors introduced an algorithm called *big foot* algorithm. The idea of this algorithm is to identify two types of links and to move towards the destination using first the links of the first type and then those of the second type. The algorithms we develop will use such a strategy. They give an optimal centralized algorithm for the permutation routing problem in full-duplex 2-circulant graphs and double-loop networks (i.e. oriented 2-circulant graphs).

Another network of great practical importance is the *double-loop* network : a network modeled by a graph with vertex set $v=\{v_0,\ldots,v_{n-1}\}$ such that there are two integers h_1 and h_2 such that $E=\{v_iv_{i\pm h_1},v_iv_{i\pm h_2}\}$. The permutation routing problem in this network is studied in [14]. The authors gave an algorithm for the permutation routing problem which in mean uses 1.12ℓ steps (the mean being empirically measured). In [15] the authors described an optimal centralized permutation routing algorithm in k-circulant graphs ($k\geq 2$), and in [42] an optimal distributed permutation routing in 2-circulant graphs was obtained.

The problem has been also studied for packets arriving dynamically. In [18], the author gave an optimal online schedule for the linear array. He also gave a 2-approximation for rings and show that, using shortest path routing, no better approximation algorithm exists. In [21], the authors studied Cube Connected Cycles : $CCC(n,2^n)$ (hypercubes of dimension n where each node is replaced by a cycle of length n). They gave an algorithm working in $\mathcal{O}(n^2)$ with $\mathcal{O}(1)$ buffers for the online partial permutation routing (PPR).

1.3.2 Plane grids

Maybe the most studied networks in the literature are the two dimensional grids (or plane grids), and among them in particular the square grid has deserved special attention. Let us briefly overview what has been previously done on (ℓ,k)-routing in plane grids.

In [32] the first optimal permutation routing (with running time $2n-2$), and queues of size 1008 appears. The queue size is reduced in [41] to 112 and further in [48] to 81. Furthermore, in [48] the authors provide another algorithm running in near-optimal time $2n+\mathcal{O}(1)$ steps with a maximum queue size of only 12. [36] gives an *asymptotically* optimal algorithm for $(1,k)$-routing on plane grids, with queues of small constant size. They introduced for the first time the $(1,k)$-routing and the $(1,any)$-routing problems. This result was further improved in [47], where the authors gave near-optimal deterministic algorithm running in $\sqrt{k}\frac{n}{2}+\mathcal{O}(n)$ steps. They gave another algorithm, slightly worse in terms of number of steps, but with queues of size only 3. They also studied the general problem of (ℓ,k)-routing in square grids. They proposed lower bounds and near-optimal randomized and deterministic algorithms. They finally extended them to higher dimensional meshes. They performed (ℓ,ℓ)-routing in $\mathcal{O}(\ell n)$ steps, the lower bound being $\Omega(\sqrt{\ell kn})$ for (ℓ,k)-routing. Finally, in [39], the authors gave deterministic and randomized algorithms for (ℓ,k)-routing in square grids, with constant queue size. The running time is $\mathcal{O}(\sqrt{\ell kn})$ steps, which is optimal according

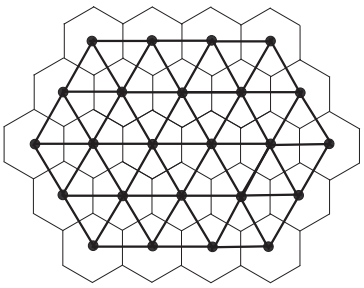


FIG. 1 – Hexagonal network (△) and hexagonal tessellation (⬡)

to the bound of [47]. This work closed a gap in the literature, since optimal algorithms were only known for $\ell = 1$ and $\ell = k$.

Nodes in a hexagonal network are placed at the vertices of a regular triangular tessellation, so that each node has up to six neighbors. In other words, a hexagonal network is a finite subgraph of the triangular grids. These networks have been studied in a variety of contexts, specially in wireless and interconnection networks. The most known application may be to model cellular networks with hexagonal networks where nodes are base stations. But these networks have been also applied in chemistry to model benzenoid hydrocarbons [52, 25], in image processing and computer graphics [26].

In a radiocommunication wireless environment [37], the interconnection network among base stations constitutes a hexagonal network, i.e. a triangular grid, as it is shown in FIG. 1.

Tessellation of the plane with hexagons may be considered as the most natural because cells have optimal diameter to area ratio. Hexagonal networks are finite subgraphs of the triangular grid. The triangular grid can also be obtained from the basic 4-mesh by adding NE to SW edges, which is called a 6-mesh in [53]. Here we study convex subgraphs, i.e. that contain all shortest paths between all pairs of nodes, of the square, triangular and hexagonal grid. Summarizing, to the best of our knowledge the only optimal algorithms concerning (ℓ, k)-routing on plane grids (according to the lower bound of [47]) have been found on square grids, but modulo a constant factor [39]. On triangular and hexagonal grids, the best results are randomized algorithms with good performance [46].

1.4 Our Contribution

In this paper we study the permutation routing, r-central and (ℓ, k)-routing problems on plane grids, that is square grids, triangular grids and hexagonal grids. We use the *store-and-forward* Δ-port model, and we consider both full and half-duplex networks.

We have seen in Section 1.3.2 that the only plane grid for which there existed an optimal (ℓ, k)-routing is the square grid. In addition, what is important is that the results of these

articles concerning (ℓ, k)-routing in plane grids are optimal modulo a constant factor. In this paper we improve these results by giving tight algorithms including the constant, in the cases of square, triangular and hexagonal grids. It is important to stress that all the algorithms presented in this paper are **distributed** (except the one given in Appendix B), i.e. can be implemented independently at each node. Our algorithms only use shortest paths, therefore they achieve minimum stretch. In addition, the algorithms are oblivious, so they can be used in an on-line scenario, unless the performance guarantees that we prove apply only to the off-line case. The main new results of this article are the following :

1. First tight (also including the constant factor) permutation routing algorithms in full-duplex hexagonal grids, and half duplex triangular and hexagonal grids.
2. First tight (also including the constant factor) r-central routing algorithms in triangular and hexagonal grids.
3. First tight (also including the constant factor) (k, k)-routing algorithms in square, triangular and hexagonal grids.
4. Good approximation algorithms for (ℓ, k)-routing in square, triangular and hexagonal grids.

This paper is structured as follows. In Section 2 we study the permutation routing problem. Although permutation routing had already been solved for square grids, we begin in Section 2.1 by illustrating our algorithm for such grids. Then in Section 2.2 we give a tight permutation routing algorithm for half-duplex triangular grids, using the optimal algorithm of [44]. In Section 2.3 we provide a tight permutation routing algorithm for full-duplex hexagonal grids and a tight permutation routing algorithm for half-duplex hexagonal grids. In Section 3 we focus on $(1, any)$-routing, giving an optimal r-central routing algorithms for the three types of grids. We finally move in Section 4 to the general (ℓ, k)-routing problem. We provide a distributed algorithm for (ℓ, k)-routing in any grid, using the ideas of the optimal algorithm for permutation routing. We also prove lower bounds for the worst-case running time of any algorithm using shortest path routing. In addition, these lower bounds allow us to prove that our algorithm turns out to be tight when $\ell = k$, yielding in this way a tight (k, k)-routing algorithm in square, triangular and hexagonal grids. We also propose in Appendix B an approach to (ℓ, k)-routing in terms of a graph coloring problem : the Weighted Bipartite Edge Coloring. We give a centralized algorithm using this reduction.

2 Permutation Routing

As we have already said in Section 1, in the permutation routing problem, each processor is the origin of at most one packet and the destination of no more than one packet. The goal is to minimize the number of time steps required to route all packets to their respective destinations. It corresponds to the case $\ell = k = 1$ of the general (ℓ, k)-routing problem. This problem has been studied in a wide diversity of scenarios, such as Mobile Ad Hoc Networks [23], Cube-Connected Cycle (CCC) Networks [21], Wireless and Radio Networks [10], All-Optical Networks [33] and Reconfigurable Meshes [9].

In a grid with full-duplex links an edge can be crossed simultaneously by two messages, one in each direction. Equivalently, each edge between two nodes u and v is made of two independent arcs $\{uv\}$ and $\{vu\}$, as illustrated in Fig. 2a.

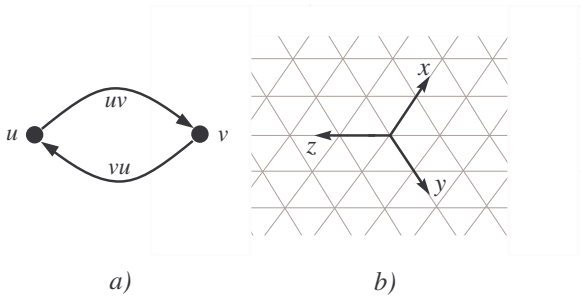


FIG. 2 – **a)** Each edge consists of two independent links. **b)** Axis used in a triangular grid

Remark 2.1 *If the network is half-duplex, it is easy to construct a 2-approximation algorithm from an optimal algorithm for the full-duplex case by introducing* odd-even *steps, as explained for example in [14].*

2.1 Square grid

Many communication networks are represented by graphs satisfying the following property : for any pair of nodes u and v, the edges of a shortest path from u to v can be partitioned into k disjoint classes according to a well-defined criterium. For instance, on a triangular grid the edges of a shortest path can be partitioned into *positive* and *negative* ones [44]. Similarly, on a k-circulant graph the edges can be partitioned into k classes according to their length.

In graphs that satisfy this property there exists a natural routing algorithm : route all packets along one class of edges after another. For hexagonal networks this algorithm turns out to be optimal [44]. Optimality for 2-circulant graphs is proved using a static approach in [19], and recently using a dynamic distributed algorithm in [42]. In [19] the authors introduce the notion of *big-foot* algorithms because their algorithm routes packets first along *long hops* and then along *short hops* in a 2-circulant graph.

On the square grid, the *big-foot* algorithm consists of two phases, moving each packet first horizontally and then vertically. In this way a packet may wait only during the second phase. Using that all destinations are distinct, the optimality for square grid is easy to prove. Summarizing, it can be proved that

Theorem 2.1 *There is a translation invariant oblivious optimal permutation routing algorithm for full-duplex networks that are convex subgraphs of the infinite square grid.*

2.1.1 Regarding the queue size

Of course, this is not the first optimal permutation routing result on square grids, as the classical $x - y$ routing (first route packets through the horizontal axis, and then through the vertical axis) has been used for a long time. Thus, another more challenging issue is to reduce the queue size, as we have already discussed in Section 1.3.2. Leighton describes in [30] a simple off-line algorithm for solving any permutation routing problem in $3n - 3$ steps on a $n \times n$ square grid, using queues of size one. Since the diameter of a $n \times n$ square grid is $2n-2$, this algorithm provides a $\frac{3}{2}$-approximation. The main drawback is that this algorithm is off-line and centralized. In contrast, our oblivious distributed algorithm is optimal in terms of running time, but it is easy to see that on a $n \times n$ square grid, the queue size can be $\frac{n-1}{2}$. Up to date, the best algorithm running in optimal time to route permutation routing instances on square grids is the algorithm of Sibeyn et al. [48], using queues of size 81. So far, there is no algorithm that guarantees optimal running time with queues of size 1, and it is unlikely that such an algorithm exists.

Remark 2.2 *The same observation regarding the unbounded queue size applies to all the algorithms described in this article. However, our aim is to match the optimal running time, rather than minimizing the queue size. Additionally, it turns out that some appropriate modifications of the permutation routing algorithms that we provide for plane grids allow us to find oblivious algorithms which route any permutation within a factor 3 of the optimal running time, and using queues of size 1 (in fact, we can say something stronger : we just need memory to keep 1 message at each node). We do not describe these modifications in this article.*

2.2 Triangular grid

We use the addressing scheme introduced in [37] and used also in [44] : we represent any address on a basis consisting of three unitary vectors $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$ on the directions of three axis x, y, z with a 120 degree angle among them, intersecting on an arbitrary (but fixed) node O . This node is the origin and is given the address $\boldsymbol{O} = (0, 0, 0)$. This basis is represented in FIG. 2b. Thus, we can assume that each node $P \in V$ is labeled with an address $\boldsymbol{P} = (P_1, P_2, P_3)$ expressed in this basis $\{\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}\}$ with respect to the origin O. At the beginning, each node S knows the address of the destination node D of the message placed initially at S, and computes the relative address $\overrightarrow{SD} = \boldsymbol{D} - \boldsymbol{S}$ of the message. Note that this relative address does not depend on the choice of the origin node O. This relative address is the only information that is added in the heading of the message to be transmitted, constituting in this way the packet to be sent through the network.

Using that $\boldsymbol{i} + \boldsymbol{j} + \boldsymbol{k} = 0$, it is easy to see that if (a, b, c) and (a', b', c') are the relative addresses of two packets, then $(a, b, c) = (a', b', c')$ if and only if there exists $d \in \mathbb{Z}$ such that $a' = a + d$, $b' = b + d$, and $c' = c + d$.

We say that an address $\overrightarrow{SD} = (a, b, c)$ is of the *shortest path form* if there is a path from node S to node D, consisting of a units of vector $\boldsymbol{i}$, b units of vector $\boldsymbol{j}$ and c units of vector $\boldsymbol{k}$, and this path has the shortest length.

Theorem 2.2 ([37]) *An address (a, b, c) is of the shortest path form if and only if at least one component is zero, and any two components do not have the same sign.*

Corollary 2.1 ([37]) *Any address has a unique shortest path form.*

Thus, each address $\overrightarrow{SD}$ written in the shortest path form has at most two non-zero components, and they have different sign. In fact, it is easy to find the shortest path form using the next result.

Theorem 2.3 ([37]) *If $\overrightarrow{SD} = a\mathbf{i} + b\mathbf{j} + c\mathbf{k}$, then*

$$|\overrightarrow{SD}| = \min(|a - c| + |b - c|, |a - b| + |b - c|, |a - b| + |a - c|).$$

Permutation routing on full-duplex triangular grids has been solved recently [44] attaining the distance lower bound of $\ell_{\max}$ routing steps, where $\ell_{\max}$ is the maximum length over the shortest paths of all packets to be sent through the network.

As said in Remark 2.1, ff the network is half-duplex, one can construct a 2-approximation algorithm from an optimal algorithm for the full-duplex case by introducing *odd-even* steps. Thus, using this algorithm we obtain an upper bound of $2\ell_{\max}$ for half-duplex triangular grids.

Let us show with an example that this naïve algorithm is tight. That is, we shall give an instance requiring at least $2\ell_{\max}$ running steps, implying that no better algorithm for a general instance exists. Indeed, consider a set of nodes distributed along a line on the triangular grid. We fix $\ell_{\max}$ and an edge e on this line, and put $\ell_{\max}$ packets at each side of e along the line, at distance at most $\ell_{\max} - 1$ from an end-vertex of e. For each packet, each destination is placed at the other side of e with respect to its origin, at distance exactly $\ell_{\max}$ from the origin. It is easy to check that the congestion of e (that is, the number of shortest paths containing e) is $2\ell_{\max}$, and thus any algorithm using shortest path routing cannot perform in less than $2\ell_{\max}$ steps. On the other hand, $\ell_{\max}$ is a lower bound for any distance, yielding that the approximation ratio of our algorithm is at most 2.

Previous observations allow us to state the next result :

Theorem 2.4 *There exists a tight permutation routing algorithm for half-duplex triangular grids performing in at most $2\ell_{\max}$ steps, where $\ell_{\max}$ is the maximum length over the shortest paths of all packets to be sent. This algorithm is a 2-approximation algorithm for a general instance.*

2.3 Hexagonal grid

In a hexagonal grid one can define three types of *zigzag chains* [50], represented with thick lines in FIG. 3. Similarly to the triangular grid, in the hexagonal grid any shortest path between two nodes uses at most two types of zigzag chains [50]. Let us now give a lower bound for the running time of any algorithm. Consider the edge labeled as e in FIG. 3, and the two chains containing it (those shaping an X). Fix $\ell_{\max}$ and e, and put one message on all nodes placed at both chains at distance at most $\ell_{\max} - 1$ from an endvertex of e. As in the case of the triangular grid, choose the destinations to be placed on the other side of e along the same zigzag chain than the originating node, at distance exactly $\ell_{\max}$ from it. It is clear that all the shortest paths contain e. It is also easy to check that the congestion of e is in this case $4\ell_{\max} - 4$, constituted of symmetric loads $2\ell_{\max} - 2$ in each direction of e. Thus, $2\ell_{\max} - 2$ establishes a lower bound for the running time of any algorithm in the full-duplex case, whereas $4\ell_{\max} - 4$ is a lower bound for the half-duplex case, under the assumption of shortest path routing.

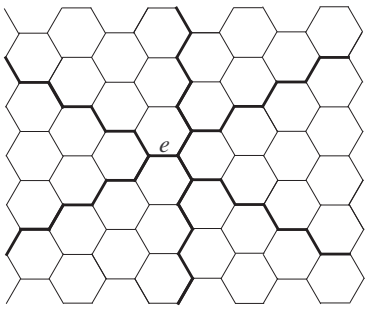


FIG. 3 – Zigzag chains in a hexagonal grid

Let us now describe a routing algorithm which reaches this bound. We have three types of edges according to the angle that they form with any fixed edge. Each edge belongs to exactly two different chains, and conversely each chain is made of two types of edges. Moreover, in an infinite hexagonal grid any 2 chains of different type intersect exactly on one edge.

Given a pair of origin and destination nodes S and D, it is possible to express the relative address $D - S$ counting the number of steps used by a shortest paths on each type of chain. In this way we obtain an address $D - S = (a, b, c)$ on a generating system made of unitary vectors following the directions of the three types of chains (it is not a basis in the strict sense, since these vectors are not linearly independent on the plane. However, we will call it so). Choose this basis so that the three vectors form angles of 120 degrees among them. As it happens on the triangular grid [37, 43], there are at most two non-zero components (see [50]), and in that case they must have different sign. Nevertheless, now the address is not

unique, since an edge placed at the *bent* (that is, a change from a type of chain to another) of a shortest path is part of both types of chains. Anyway, this ambiguity is not a problem in the algorithm that propose, as we will see below.

Suppose first that edges are bidirectional or, said otherwise, full-duplex. Roughly, the idea is to use the optimal algorithm for triangular grids described in [43], and adapt it to hexagonal grids. For that purpose we label the three types of zigzag chains c_1, c_2, c_3, and the three types of edges e_1, e_2, e_3. Without loss of generality, we label them in such a way that c_1 uses edges of type e_2 and e_3, c_2 uses e_1 and e_3, and c_3 uses e_1 and e_2 (see FIG. 4).

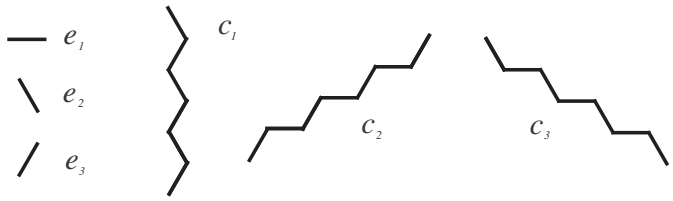


FIG. 4 – 3 types of chains and edges in a hexagonal grid

For each type of edge, we define two phases according to the type of chain that uses this type of edge. This defines two global phases, namely : during Phase 1, c_1 uses e_2, c_2 uses e_3, and c_3 uses e_1. Conversely, during Phase 2 c_1 uses e_3, c_2 uses e_1, and c_3 uses e_2.

We suppose that at each node packets are grouped into distinct queues according to the next edge (according to the rules of the algorithm) along its shortest path. Given the relative addresses $D - S$ in the form (a, b, c), the algorithm can be described as follows.

At each node u of the network :

1) During the first step, move all packets along the direction of their negative component. If a packet's address has only a positive component, move it along this direction.

2) From now on, change alternatively between Phase 1 and Phase 2. At each step (the same for both phases) :

- **a)** If there are packets with negative components, send them immediately along the direction of this component.
- **b)** If not, for each outgoing edge order the packets in decreasing number of remaining steps, and send the first packet of each queue.

3) If at some point, all the packets in u have remaining distance one, send them immediately.

Let us analyze the correctness and optimality of this algorithm.

In **1)** all packets can move, since initially there is at most one packet at each node. In **2a)**, there can only be one packet with negative component at each outgoing edge [43]. In **2b)** *the* packet with maximum remaining length at each outgoing edge is unique, since all these

packets are moving along their last direction (their negative component is already finished, otherwise they would be in **2a)**) and each node is the destination of at most one packet. Hence, using this algorithm every 2 steps (one of Phase 1 and one of Phase 2) the maximum remaining distance over all packets decreases by one. Moreover, during the first step all packets decrease their remaining distance by one. Because of this, after the $(2\ell_{\max} - 3)th$ step the maximum remaining distance has decreased at least $1 + \frac{2\ell_{\max}-4}{2} = \ell_{\max} - 1$ times, hence the maximum remaining distance is 1, and we are in **3)**. Since all destinations are different, all packets can reach simultaneously their destinations. Thus, the total running time is at most $2\ell_{\max} - 3 + 1 = 2\ell_{\max} - 2$, meeting the *worst case* lower bound. Again, $\ell_{\max}$ is a lower bound for any instance, hence the algorithm constitutes a 2-approximation for a general instance.

Theorem 2.5 *There exists a tight permutation routing algorithm for full-duplex hexagonal grids performing in $2\ell_{\max} - 2$ steps, where $\ell_{\max}$ is the maximum length over the shortest paths of all packets to be sent.*

Remark 2.3 *The optimality stated in Theorem 2.5 is true only under the assumption of shortest path routing. This means that for certain traffic instances the total deliver time may be shorter if some packets do not go through their shortest path. To illustrate this phenomenon, consider the example of* FIG. *5. Node labeled i wants to send a message to node labeled i', for $i = 1, \ldots, 8$. We have that $\ell_{\max} = 5$, and thus our algorithm performs in $2\ell_{\max} - 2 = 8$ steps. It is clear that all shortest paths use edge e, and its congestion bottlenecks the running time. Suppose now that we route the messages originating at even nodes through the path defined by the edges $\{abcd\}$, instead of $\{fe\}$, and keep the shortest path routing for messages originating at odd nodes. One can check that with this routing only 7 steps are required.*

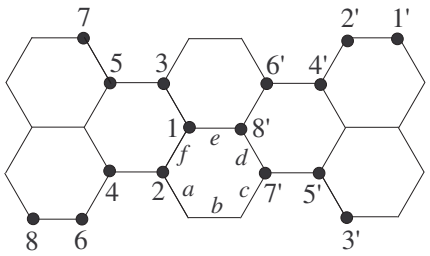


FIG. 5 – Shortest path is not always the best choice

In the half-duplex case, just introduce again odd-even steps in both phases. Thus, we have Phase 1-even, Phase 1-odd, Phase 2-even, and Phase 2-odd, which take place sequentially. Now, **1)** consists obviously of two steps (even/odd). Using this algorithm, every 4 steps the

maximum remaining distance decreases by one. In addition, during the first 2 steps and during the last 2 steps all packets decrease their remaining distance by one. Thus, the total running time is at most twice the time of the full-duplex case, that is $2(2\ell_{\max}-2) = 4\ell_{\max}-4$ steps, meeting again the lower bound for the running time of any routing algorithm using shortest path routing. Again, this algorithm constitutes a 4-approximation for a general instance.

Theorem 2.6 *There exists an tight permutation routing algorithm for half-duplex hexagonal grids performing in $4\ell_{\max} - 4$ steps, where $\ell_{\max}$ is the maximum length over the shortest paths of all packets to be sent.*

Remark 2.4 *As explained in Appendix A, there exists an embedding of the triangular grid into the hexagonal grid with load, dilation, and congestion 2. Using this embedding, any algorithm performing on k steps on the triangular gird performs on $2k$ steps on the hexagonal grid. Using this fact, we obtain a permutation routing algorithm on full-duplex hexagonal grids performing on $2\ell_{\max}$ steps. Note that the optimal result given in Theorem 2.5 is slightly better.*

The same applies to half-duplex hexagonal networks, with a running time of $4\ell_{\max}$ using the embedding, in comparison to $4\ell_{\max} - 4$ steps given by Theorem 2.6.

3 $(1, any)$-Routing

In this case the routing model is the following : each packet has at most one packet to send, but there are no constraints on the destination. That is, in the worst case all packets can be sent to one node. This special case where all packets want to send a message to the same node in often called *gathering* in the literature [5]. Notice that this routing model is conceptually different from the $(1, k)$-routing, where the maximum number of packets that a node can receive is fixed *a priori*.

Square grid Assume first that edges are bidirectional. The modifications for the half-duplex case are similar to those explained in the previous section.

We will focus on the case where all packets surrounding a given vertex want to send a packet to that vertex. We call this situation *central* routing, and if we want to specify that all nodes at distance at most r from the center want to send a packet, we note it as *r-central* routing. Note that this situation is realistic in many practical applications, since the central vertex can play the role of a router or a gateway in a local network.

Lemma 3.1 (Lower Bound) *The number of steps required in a r-central routing is at least $\binom{r+1}{2}$.*

Proof: Let us use the bisection bound [17] to prove the result. It is easy to count the number of points at distance at most r from the center, which is $4\binom{r+1}{2}$. Now consider the cut consisting of the four edges outgoing from the central vertex. All packets must traverse one

of these edges to arrive to the central vertex. This cut gives the bisection bound of $4\binom{r+1}{2}/4$ routing steps. □

Let us now describe an algorithm meeting the lower bound.

Proposition 3.1 *There exists an optimal r-central routing algorithm on square grids performing in $\binom{r+1}{2}$ routing steps.*

Proof: Express each node address in terms of the relative address with respect to the central vertex. In this way each node is given a label (a, b). Then, for each packet placed in a node with label (a, b) our routing algorithm performs the following :

- If $ab = 0$, send the packet along the direction of the non-zero component.
- If $ab > 0$, send the packet along the vertical axis.
- If $ab < 0$, send the packet along the horizontal axis.

Queues are managed so that the packets having greater remaining distance have priority.

This routing divides the square grid into 4 subregions surrounding the central vertex, as shown in FIG. 6. The type of routing performed in each subregion is symbolized by an arrow.

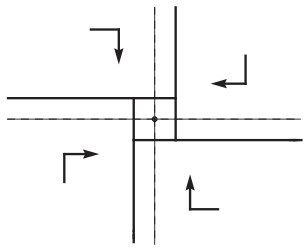


FIG. 6 – Division of the grid in the proof of Proposition 3.1

Let us now compute the running time in the r-central case. It is obvious that using this algorithm all packets are sent to the 4 axis outgoing from the central vertex. The congestion of the edge in the axis containing the central vertex along each line is $1+2+3+\ldots+r = \binom{r+1}{2}$. Since at each step one packet reaches its destination along each line, we conclude that $\binom{r+1}{2}$ is the total running time of the algorithm. □

Triangular grid The same idea of the square grid applies to the triangular grid. In this case, the number of nodes at distance at most r is $6\binom{r+1}{2}$. The cut is made of 6 edges. Dividing the plane onto 6 subregions gives again an optimal algorithm performing in $\binom{r+1}{2}$ steps.

Hexagonal grid The same idea gives an optimal routing in the r-central case. In this case the degree of each vertex is 3, and then it is easy to check (maybe a drawing using FIG. 3 can help) that there are $3\binom{r+1}{2}$ nodes at distance at most d that may want to send a message to the central vertex, and the cut has size 3. As expected, the running time is again $\binom{r+1}{2}$.

4 (ℓ, k)-Routing

Recall that in the general (ℓ, k)-routing problem each node can send at most ℓ packets and receive at most k packets. We propose a distributed approximation algorithm using the ideas of the algorithms that we have developed for the permutation routing problem. We also provide lowers bounds for the running time of any algorithm using shortest path routing, that allow us to prove that our algorithm is tight when $\ell = k$, on any grid.

Remark 4.1 *We also propose in Appendix B an approach to find a solution of the (ℓ, k)-routing problem, on any grid, using the problem of* WEIGHTED EDGE COLORING *in a bipartite graph. Nevertheless, the algorithm obtained using this approach is centralized.*

We start by describing the results for full-duplex triangular grids. The results can be completely adapted to square grids, but we focus on the other grids since there were few results in the literature. We also show how to adapt the results to hexagonal grids and to the half-duplex version. In this section we denote $c := \left\lceil \frac{\max\{\ell,k\}}{\min\{\ell,k\}} \right\rceil = \lceil \max\{\frac{\ell}{k}, \frac{k}{\ell}\} \rceil$. Note that $c \geq 1$. Lemma 4.1 and Lemma 4.2 provide two lower bounds for the running time of any algorithm using shortest paths.

Lemma 4.1 (First lower bound) *The worst-case running time of any algorithm for (ℓ, k)-routing on full-duplex triangular grids using shortest path routing satisfies*

$$\textit{Running time} \geq \min\{\ell, k\} \cdot \ell_{\max}$$

Proof: Consider a set of $\ell_{\max}$ nodes placed along a line, placed consecutively at one side of a distinguished edge e. Each node wants to send $\min\{\ell, k\}$ messages to the nodes placed at the other side of e along the line, at distance $\ell_{\max}$ from it. Then the congestion of e is $\min\{\ell, k\} \cdot \ell_{\max}$, giving the bound. □

Definition 4.1 *Given a vertex v, we call the rectangle of side (a, b) starting at v the set $R^v_{a,b} = \{v + \alpha\mathbf{i} + \beta\mathbf{j}, 0 \leq \alpha < a, 0 \leq \beta < b\}$ We call such a rectangle a square if $a = b$. Notice that in the triangular grid the node set is generated by $\{\mathbf{i}, \mathbf{j}, \mathbf{k}\}$, where $\mathbf{k} = -\mathbf{i} - \mathbf{j}$, as we have explained in Section 2.2.*

Using standard graph terminology, given a graph $G = (V, E)$ and a subset $S \subseteq V$, the set $\Gamma(S)$ denotes the (open) neighborhood in G of the vertices in S. The following theorem can be found, for example, in [13].

Theorem 4.1 (Corollary of Hall's theorem [13]) *Let $G=(V,E)$ be a bipartite graph, with $V=X\cup Y$. If for all subsets A of X, $|\Gamma(A)|\geq c|A|$, then for each $x\in X$, there exists $S_x\subset Y$ such that $|S_x|=c$, and $\forall x,x'\in X$, $S_x\cap S_{x'}=\emptyset$ and $S_x\subset\Gamma(x)$.*

We use this theorem to prove the following lower bound.

Lemma 4.2 (Second lower bound) *The worst-case running time of any algorithm for (ℓ,k)-routing on full-duplex triangular grids using shortest path routing satisfies*

$$\textit{Running time}\geq\left\lceil\frac{\max\{\ell,k\}}{4}\cdot\left\lfloor\frac{\ell_{\max}+1}{\sqrt{c+1}}\right\rfloor\right\rceil\ ,$$

where $c=\left\lceil\frac{\max\{\ell,k\}}{\min\{\ell,k\}}\right\rceil$.

Proof: Suppose without loss of generality that $\ell\geq k$, otherwise replace ℓ by $\max\{\ell,k\}$. Let v be a vertex, and consider the square $R^v_{d,d}$, with $d:=\left\lfloor\frac{\ell_{\max}+1}{\sqrt{c+1}}\right\rfloor$. We claim that all nodes inside this square can send ℓ messages such that all destination nodes are in the destination set $D=R^v_{d+\ell_{\max},d+\ell_{\max}}\setminus R^v_{d,d}$. Let S be the subgrid generated by positive linear combinations of the vectors $\mathbf{i}$ and $\mathbf{j}$. More precisely, $S:=\{v+\alpha\boldsymbol{i}+\beta\boldsymbol{j},\alpha\geq 0,\beta\geq 0\}$. FIG. 2b gives a graphical illustration.

To prove this, we consider a bipartite graph H on vertex set $R^v_{d,d}\cup D$, with an edge between a vertex of $R^v_{d,d}$ and a vertex of D if they are at distance at most $\ell_{\max}$ in S. To apply Theorem 4.1, we have to show that any subset of vertices $A\subset R^v_{d,d}$ has at least $c|A|$ neighbors in H. Theorem 4.1 will then ensure the existence of a feasible repartition of the messages from vertices of $R^v_{d,d}$ to those of D such that they all travel a distance at most $\ell_{\max}$.

Given $A\subset R^v_{d,d}$, let us call $D_A:=\{u\in D\ :\mathrm{dist}_S(A,u)\leq\ell_{\max}\}$, where $\mathrm{dist}_S(A,u)$ means the minimum distance in S from any vertex of A to the vertex u. For any $A\subset R^v_{d,d}$, we need to show that

$$|D_A|\geq c|A| \tag{1}$$

Without loss of generality we suppose that A is maximal, in the sense that there is no set A' strictly containing A with $D_A=D_{A'}$. Instead of considering all possible sets A, we will show below that we can restrict ourselves to rectangles. Hence given a set A, we denote by R_A the smallest rectangle containing the subset of vertices A. We first claim that

$$|D_{R_A}\setminus D_A|\leq|R_A\setminus A| \tag{2}$$

Indeed, this equality can be shown by induction on $|R_A\setminus A|$. For $|R_A\setminus A|=0$ the equality is trivial. Suppose that it is true for $|R_A\setminus A|$. The induction step consists in showing that there is an element x in $R_A\setminus A$ such that $|D_{R_A}\setminus D_{A\cup\{x\}}|-|D_{R_A}\setminus D_A|\leq 1$ (note that

$D_{R_{A\cup\{x\}}} = D_{R_A}$) :

- If there exists x such that $x+\boldsymbol{j}$ and $x-\boldsymbol{i}$ are in A and $x-\boldsymbol{j}$ is not in A, then we select this x. From x the only new vertex we may add to D_A is $x+\ell_{\max}\boldsymbol{i}$.
- Otherwise, if there exists x such that $x-\boldsymbol{j}$ and $x-\boldsymbol{i}$ are in A and $x+\boldsymbol{i}$ is not in A, then we select this x. In this case the only new vertex we may add to D_A is $x-\ell_{\max}\boldsymbol{k}$.
- If none of the previous cases holds, since R_A is the smallest rectangle containing A, and A is maximal, then necessarily there exists an x such that $x+\boldsymbol{i}$ and $x-\boldsymbol{j}$ are in A and $x-\boldsymbol{i}$ is not in A. We select this x, and the only new vertex we may add to D_A is $x+\ell_{\max}\boldsymbol{j}$.

Thus, in all cases there exists an x adding at most one neighbor to $D_{R_A}\setminus D_A$, which finishes the induction step and proves Equation (2). To finish the proof of the fact that we can restrict ourselves to rectangles, we show that, for any subset A, if Inequality (1) holds for R_A, then it also holds for A. Indeed, Inequality (1) applied to R_A gives :

$$c|R_A| \leq |D_{R_A}| \ , \quad \text{which is equivalent to}$$

$$c(|A| + |R_A \setminus A|) \leq |D_A| + |D_{R_A} \setminus D_A| \tag{3}$$

Using Inequality (2) and the fact that $c \geq 1$, Inequality (3) clearly implies that Inequality (1) holds.

Henceforth we assume that A is a rectangle. The last simplification consists in proving that we can restrict ourselves to rectangles containing v. In other words, it will be sufficient to prove Inequality (1) for all rectangles $R^v_{a,b}$. Given a rectangle R not positioned at v, the rectangle R' of the same size positioned at v has less neighbors, hence if Inequality (1) holds for R', it also holds for R.

Finally let us prove that Inequality (1) holds for all rectangles $R^v_{a,b}$, with $1 \leq a, b < d$. We have that $|R^v_{a,b}| = ab$ and $|D_{R^v_{a,b}}| = (a+\ell_{\max})(b+\ell_{\max}) - d^2$.

By the choice of d, starting from the Inequality $d^2 \leq \frac{(\ell_{\max}+1)^2}{c+1}$ and using that $1 \leq a, b$, one obtains that $d^2 c \leq (\ell_{\max}+a)(\ell_{\max}+b) - d^2$ for any $1 \leq a, b < d$. This implies, using $a, b < d$, that $cab \leq (a+\ell_{\max})(b+\ell_{\max}) - d^2$ for any $1 \leq a, b < d$, hence Inequality (1) (i.e. $c|R^v_{a,b}| \leq |D_{R^v_{a,b}}|$) holds.

So by Theorem 4.1, each one of the d^2 nodes in $R^v_{d,d}$ can send ℓ messages to the nodes of D. Since the number of edges going from $R^v_{d,d}$ to D is $4d-1$, we apply the bisection bound discussed in Section 1.1.1 to conclude that there is an edge of the border of the square $R^v_{d,d}$ with congestion at least $\left\lceil \frac{\ell \cdot d^2}{4d-1} \right\rceil > \left\lceil \frac{\ell \cdot d}{4} \right\rceil$. This finishes the proof of the lemma. □

We observe that this second lower bound is strictly better than the first one if and only if

$$\frac{c}{\sqrt{c+1}} > \frac{4\ell_{\max}}{\ell_{\max}+1}$$

If both c and $\ell_{\max}$ are big, the condition becomes approximately :

$$\frac{\max\{\ell,k\}}{\min\{\ell,k\}} > 16$$

That is, the second lower bound is better when the difference between ℓ and k is big. This is the case of broadcast or gathering, where messages are originated (or destined) from (or to) a small set of nodes of the network.

The two lower bounds can be combined to give :

Lemma 4.3 (Combined lower bound) *The worst-case running time of any algorithm for (ℓ,k)-routing on full-duplex triangular grids using shortest path routing satisfies*

$$\textit{Running time} \geq \max\left(\ell_{\max}\cdot\min\{\ell,k\},\ \max\{\ell,k\}\cdot\left\lfloor\frac{\ell_{\max}+1}{4\sqrt{c+1}}\right\rfloor\right) \approx \ell_{\max}\cdot\max\left(\min\{\ell,k\},\ \frac{\max\{\ell,k\}}{4\sqrt{c+1}}\right)$$

Now we provide an algorithm from which we derive an upper bound.

Proposition 4.1 (Upper bound (algorithm)) *The algorithm for (ℓ,k)-routing on full-duplex triangular grids is the following : route all packets as in the permutation routing case. That is, at each node send packets first in their negative component, breaking ties arbitrarily (there can be ℓ packets in conflict in a negative component). If there are no packets with negative components, send any of the (at most k) packets with maximum remaining distance.*

$$\textit{Running time} \leq \begin{cases} \min\{\ell,k\}\cdot\frac{c(c-1)}{2} + \max\{\ell,k\}\cdot(\ell_{\max}-c+1) & \textit{, if } c \leq \ell_{\max} \\ \min\{\ell,k\}\cdot\frac{\ell_{\max}(\ell_{\max}+1)}{2} & \textit{, if } c > \ell_{\max} \end{cases}$$

Proof: Suppose again without loss of generality that $\ell \geq k$. We proceed by decreasing induction on $\ell_{\max}$. We prove that after $\min\{\ell,\ell_{\max}k\}$ steps, each paquet will be at distance at most $\ell_{\max}-1$ of its destination. This yields

$$\text{Running time}(\ell_{\max}) \leq \min\{\ell,\ell_{\max}k\} + \text{Running time}(\ell_{\max}-1)$$

$$\leq \min\{\ell,\ell_{\max}k\} + \begin{cases} \min\{\ell,k\}\cdot\frac{c(c-1)}{2} + \max\{\ell,k\}\cdot(\ell_{\max}-c) & \text{, if } c \leq \ell_{\max}-1 \\ \min\{\ell,k\}\cdot\frac{\ell_{\max}(\ell_{\max}-1)}{2} & \text{, if } c > \ell_{\max}-1 \end{cases}$$

$$\leq \begin{cases} \min\{\ell,k\}\cdot\frac{c(c-1)}{2} + \max\{\ell,k\}\cdot(\ell_{\max}-c+1) & \text{, if } c \leq \ell_{\max} \\ \min\{\ell,k\}\cdot\frac{\ell_{\max}(\ell_{\max}+1)}{2} & \text{, if } c > \ell_{\max} \end{cases}$$

Let us consider the messages at distance $\ell_{\max}$ to their destinations. They are of two types, the one moving according to their negative component and the one moving according to their positive component.

If $c \leq \ell_{\max}$ the first ones move after at most ℓ time steps. If $c < \ell_{\max}$ they move more quickly, indeed they move at least once every $\ell_{\max} k$ steps ($\ell_{\max} k \leq c \cdot k = \ell$). This is due to the fact that when $c < \ell_{\max}$ at a given vertex, at most $\ell_{\max} k$ messages may have to move according to their negative component toward a node at distance $\ell_{\max}$.

About the messages which move according to their positive component, since a node is the destination of at most k messages, they may wait at most k steps.

Consequently, $\ell_{\max}$ decreases by at least one every $\min\{\ell, \ell_{\max} k\}$ steps, which gives the result. □

This gives an algorithm which is fully distributed. Dividing the running time of this algorithm by the combined lower bound we obtain the following ratio :

$$\begin{cases} \frac{\min\{\ell,k\}\cdot\binom{c}{2}+\max\{\ell,k\}\cdot(\ell_{\max}-c+1)}{\ell_{\max}\cdot\max\left(\min\{\ell,k\}\ ,\ \frac{\max\{\ell,k\}}{4\sqrt{c+1}}\right)} & \text{, if } c \leq \ell_{\max} \\ \frac{\min\{\ell,k\}\cdot(\ell_{\max}+1)}{2\cdot\max\left(\min\{\ell,k\}\ ,\ \frac{\max\{\ell,k\}}{4\sqrt{c+1}}\right)} & \text{, if } c > \ell_{\max} \end{cases}$$

We observe that in all cases the running time of the algorithm is at most $\max\{\ell, k\} \cdot \ell_{\max}$. In particular, when $\ell = k$ (that is, $c = 1$) the running time is exactly $\max\{\ell, k\} \cdot \ell_{\max} = \min\{\ell, k\} \cdot \ell_{\max}$, and therefore it is tight (see lower bound of Lemma 4.1).

Corollary 4.1 *There exists a tight algorithm for (k, k)-routing in full-duplex triangular grids.*

The previous algorithms can be generalized for half-duplex triangular grids as well as for full and half-duplex hexagonal grids. The generalization to half-duplex grids is obtained by just adding a factor 2 in both the lower bound and the running time of the algorithm, as we did for the permutation routing algorithm. Thus, let us just focus on the case of full-duplex hexagonal grids, for which we have the following theorems :

Theorem 4.2 *There exists an algorithm for (ℓ, k)-routing in full-duplex hexagonal grids whose running time is at most :*

$$\text{Running time} \leq \begin{cases} 2\min\{\ell,k\} \cdot \frac{c(c-1)}{2} + 2\max\{\ell,k\} \cdot (\ell_{\max} - c + 1) & \text{, if } c \leq \ell_{\max} \\ 2\min\{\ell,k\} \cdot \frac{\ell_{\max}(\ell_{\max}+1)}{2} & \text{, if } c > \ell_{\max} \end{cases}$$

Lemma 4.4 (First lower bound) *No algorithm based on shortest path routing can route all messages using less than $2\min\{\ell, k\} \cdot \ell_{\max} - \min\{\ell, k\}$ steps in the worst case.*

Definition 4.2 *Given a vertex v, we call the* rectangle *of the hexagonal grid of side (a, b) starting at v to the subset of the hexagonal grid $R^v_{hex\,a,b} = \{v + \alpha\mathbf{i} + \beta\mathbf{j} + \gamma\mathbf{k}, 0 \leq \alpha < a, -\gamma < \beta < b, 0 \leq \gamma < b\} \cap H$ where H is the vertex set of the hexagonal grid. We call such a rectangle a* square *if $a = b$.*

The following lemma gives a second lower bound on the running time of any algorithm using shortest path routing on full-duplex hexagonal grids.

Lemma 4.5 (Second lower bound) *The worst-case running time of any algorithm using shortest path routing on full-duplex hexagonal grids satisfies :*

$$\textit{Running time} \geq \left\lceil \max\{\ell,k\}(2d + \frac{d-2}{2d+1}) \right\rceil \quad ,$$

where $d = \left\lfloor \frac{\sqrt{73c+64\ell_{\max}^2+121+144\ell_{\max}}}{8\sqrt{c+1}} - \frac{3}{8} \right\rfloor$ *and* $c = \left\lceil \frac{\max\{\ell,k\}}{\min\{\ell,k\}} \right\rceil$.

Notice that when $\frac{\ell_{\max}}{c}$ is big, this value tends to $2\max\{\ell,k\}\frac{\ell_{\max}}{\sqrt{c+1}}$, obtaining a performance around twice better than in triangular grids.

Proof: The proof consists in showing that the vertices of $R^v_{hex\,d,d}$ can simultaneously send $\max(\ell,k)$ messages to some vertices of $R^v_{hex\,d+\ell_{\max},d+\ell_{\max}} \setminus R^v_{hex\,d,d}$. This is done as for the triangular grid, using again Theorem 4.1. We do not give all the details, since the idea behind is the same as the proof of Lemma 4.2.

Since the number of vertices inside $R^v_{hex\,d,d}$ is $4d^2+d-2$, and the number of edges outgoing from $R^v_{hex\,d,d}$ is $2d+1$, the congestion on these edges is $\max\{\ell,k\}\frac{4d^2+d-2}{2d+1} = \max\{\ell,k\}(2d+\frac{d-2}{2d+1})$. □

5 Conclusions and Further Research

In this article we have studied the permutation routing, the r-central routing and the general (ℓ,k)-routing problems on plane grids, that is square grids, triangular grids and hexagonal grids. We have assumed the *store-and-forward* Δ-port model, and considered both full and half-duplex networks. The main new results of this article are the following :

1. Tight (also including the constant factor) permutation routing algorithms on full-duplex hexagonal grids, and half duplex triangular and hexagonal grids.
2. Tight (also including the constant factor) r-central routing algorithms on triangular and hexagonal grids.
3. Tight (also including the constant factor) (k,k)-routing algorithms on square, triangular and hexagonal grids.
4. Good approximation algorithms for (ℓ,k)-routing in square, triangular and hexagonal grids, together with new lower bounds on the running time of any algorithm using shortest path routing.

All these algorithms are completely distributed, i.e. can be implemented independently at each node. Finally, we have also formulated the (ℓ, k)-routing problem as a WEIGHTED EDGE COLORING problem on bipartite graphs.

There still remain several interesting open problems concerning (ℓ, k)-routing on plane grids. Of course, the most challenging problem seems to find a tight (ℓ, k)-routing algorithm for any plane grid, for $\ell \neq k$. Another interesting avenue for further research is to take into account the queue size. That is, to devise (ℓ, k)-routing algorithms with bounded queue size, or that optimize both the running time and the queue size, under a certain trade-off.

Acknowledgment. We want to thank Frédéric Giroire, Cláudia Linhares-Sales and Bruce Reed for insightful discussions.

Références

[1] J. Aspnes, Y. Azar, A. Fiat, S. Plotkin, and O. Waarts. On-line routing of virtual circuits with applications to load balancing and machine scheduling. *Journal of the ACM (JACM)*, 44(3) :486–504, 1997.

[2] J. Aspnes, C. Busch, S. Dolev, P. Fatourou, C. Georgiou, and A. Shvartsman. Eight Open Problems in Distributed Computing. *Bulletin of the EATCS no*, 90 :109–126, 2006.

[3] B. Awerbuch and Y. Azar. Local optimization of global objectives : competitive distributeddeadlock resolution and resource allocation. *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, pages 240–249, 1994.

[4] Y. Azar, E. Cohen, A. Fiat, H. Kaplan, and H. Räcke. Optimal oblivious routing in polynomial time. *Journal of Computer and System Sciences*, 69(3) :383–394, 2004.

[5] J.-C. Bermond, J. Galtier, R. Klasing, N. Morales, and S. Pérennes. Hardness and approximation of Gathering in static radio networks. *Parallel Processing Letters*, 16(2) :165–183, 2006.

[6] A. Borodin and J. Hopcroft. Routing, merging and sorting on parallel models of computation. *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, pages 338–344, 1982.

[7] C. Busch, M. Magdon-Ismail, and J. Xi. Oblivious routing on geometric networks. *Proceedings of the 17th annual ACM symposium on Parallelism in algorithms and architectures*, pages 316–324, 2005.

[8] C. Busch, M. Magdon-Ismail, and J. Xi. Optimal oblivious path selection on the mesh. *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS 2005), Denver, Colorado, USA, April*, 2005.

[9] J. Cogolludo and S. Rajasekaran. Permutation Routing on Reconfigurable Meshes. *Algorithmica*, 31 :44–57, 2001.

[10] A. Datta. A Fault-Tolerant Protocol for Energy-Efficient Permutation Routing in Wireless Networks. *IEEE Transactions on Computers*, 54(11) :1409–1421, November 2005.

[11] D. de Werra, M. Demange, B. Escoffier, J. Monnot, and V. T. Paschos. Weighted Coloring on Planar, Bipartite and Split Graphs : Complexity and Improved Approximation. *Lecture Notes in Computer Science*, 3341 :896–907, 2004.

[12] M. Demange, D. de Werra, J. Monnot, and V. T. Paschos. Weighted node coloring : when stable sets are expensive. *WG'02 LNCS*, 2573 :114–125, 2002.

[13] R. Diestel. *Graph Theory*. Springer-Verlag, 2005.

[14] T. Dobravec, B. Robič, and J. Žerovnik. Permutation routing in double-loop networks : design and empirical evaluation. *Journal of Systems Architecture*, 48 :387–402, 2003.

[15] T. Dobravec, J. Žerovnik, and B. Robič. An optimal message routing algorithm for circulant networks. *Journal of Systems Architecture*, 52 :298–306, 2006.

[16] P. Fraigniaud and E. Lazard. Methods and problems of communication in usual networks. *Discrete Applied Mathematics*, 53 :79–134, 1994.

[17] M. D. Grammatikakis, M. K. D. F. Hsu, and J. Sibeyn. Packet Routing in Fixed-Connection Networks : a Survey. *Journal of Parallel and Distributed Processing*, 54(2) :77–132, 1998.

[18] J. T. Havill. Online Packet Routing on Linear Arrays and Rings. *Lecture Notes in Computer Science*, 2076 :773–784, 2001.

[19] F. Hwang, T. Lin, and R. Jan. A Permutation Routing Algorithm for Double Loop Network. *Parallel Processing Letters*, 7(3) :259–265, 1997.

[20] F. Hwang, Y. Yao, and B. Dasgupta. Some permutation routing algorithms for low-dimensional hypercubes. *Theoretical Computer Science*, 270 :111–124, 2002.

[21] G. E. Jan and M.-B. Lin. Concentration, load balancing, partial permutation routing, and superconcentration on cube-connected cycles parallel computers. *Journal of Parallel and Distributed Compututing*, 65 :1471–1482, 2005.

[22] C. Kaklamanis, D. Krizanc, and T. Tsantilas. Tight bounds for oblivious routing in the hypercube. *Theory of Computing Systems*, 24(1) :223–232, 1991.

[23] D. Karimou and J. F. Myoupo. An Application of an Initialization Protocol to Permutation Routing in a Single-Hop Mobile Ad Hoc Networks. *The Journal of Supercomputing*,

31 :215–226, 2005.

[24] A. Kesselman and K. Kogan. Non-Preemptive Scheduling of Optical Switches. In *IEEE GLOBECOM*, pages 1840–1844, 2004.

[25] S. Klavžar, A. Vesel, and P. Žigert. On resonance graphs of catacondensed hexagonal graphs : structure, coding, and hamiltonian path algorithm. *MATCH Communications in Mathematical and in Computer Chemistry*, 49(49) :99–116, 2003.

[26] E. Kranakis, H. Sing, and J. Urrutia. Compas Routing in Geometric Graphs. In *11th Canadian Conference of Computational Geometry*, pages 51–54, 1999.

[27] M. Kunde, R. Niedermeier, and P. Rossmanith. Faster sorting and routing on grids with diagonals. In *11th Symposium of Theoretical Computer Science*, 775, pages 225–236. Lecture Notes on Computer Science, 1994.

[28] F. Leighton, B. M. Maggs, and A. W. Richa. Fast Algorithms for Finding O(Congestion + Dilation) Packet Routing Schedules. In *28th Annual Hawaii International Conference on System Sciences*, pages 555–563, 1995.

[29] F. T. Leighton, B. M. Maggs, and S. B. Rao. Packet Routing and Job-Shop Scheduling in O(congestion + dilation) Steps. *Combinatorica*, 14(2) :167–186, 1994.

[30] T. Leighton. *Introduction to Parallel Algorithms and Architectures : Arrays-Trees-Hypercubes*. Morgan-Kaufman, San Mateo, California, 1992.

[31] T. Leighton, B. Maggs, and S. Rao. Universal packet routing algorithms. *Foundations of Computer Science, 1988., 29th Annual Symposium on*, pages 256–269, 1988.

[32] T. Leighton, F. Makedon, and I. G. Tollis. A $2n-2$ Step Algorithm for Routing in an $n \times n$ Array with Constant-Size Queues. *Algorithmica*, 14 :291–304, 1995.

[33] W. Liang and X. Shen. Permutation Routing in All-Optical Product Networks. *IEEE Transactions on Circuits and Systems*, 49(4) :533–538, 2002.

[34] B. Maggs. A Survey of Congestion+ Dilation Results for Packet Scheduling. *40th Annual Conference on Information Sciences and Systems*, 22(24) :1505–1510, 2006.

[35] B. Maggs, F. auf der Heide, B. Vocking, and M. Westermann. Exploiting locality for data management in systems of limited bandwidth. *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science*, pages 284–293, 1997.

[36] F. Makedon and A. Symvonis. Optimal algorithms for the many-to-one routing problem on 2-dimensional meshes. *Microprocessors and Microsystems*, 17 :361–367, 1993.

[37] F. G. Nocetti, I. Stojmenović, and J. Zhang. Addressing and Routing in Hexagonal

Networks with Applications for Tracking Mobile Users and Connection Rerouting in Cellular Networks. *IEEE Transactions on Parallel and Distributed Systems*, 13(9) :963–971, 2002.

[38] R. Ostrovsky and Y. Rabani. Universal O(congestion + dilation + $\log^{1+\varepsilon} N$) Local Control Packet Switching Algorithms. In *29th Annual ACM Symposium on the Theory of Computing*, pages 644–653, New York, 1997.

[39] A. Pietracaprina and G. Pucci. Optimal Many-to-One Routing on the Mesh with Constant Queues. *Lecture Notes in Computer Science*, 2150 :645–649, 2001.

[40] H. Racke. Minimizing congestion in general networks. *Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*, pages 43–52, 2002.

[41] S. Rajasekaran and R. Overholt. Constant queue routing on a mesh. *Journal of Parallel and Distributed Computing*, 15 :160–166, 1992.

[42] B. Robič and J. Žerovnik. Minimum 2-terminal routing in 2-jump circulant graphs. *Computers and Artificial Intelligence*, 19(1) :37–46, 2000.

[43] I. Sau and J. Žerovnik. An Optimal Permutation Routing Algorithm for Full-Duplex Hexagonal Mesh Networks. *Preprint series, University of Ljubljana, Institute of Mathematics, Physics and Mechanics*, 44(1017), 2006. ISSN 1318–4865.

[44] I. Sau and J. Žerovnik. Optimal permutation routing on mesh networks. In *Proc. of International Network Optimization Conference*, Belgium, April 2007. 6 pages.

[45] C. Scheideler. *Universal Routing Strategies for Interconnection Networks.* Springer, 1998.

[46] J. Sibeyn. Routing on Triangles, Tori and Honeycombs. *International Journal of Fundations of Computer Science*, 8(3) :269–287, 1997.

[47] J. Sibeyn and M. Kaufman. Deterministic 1-k routing on meshes (with applications to worm-hole routing). In LNCS, editor, *11th Symposium on Theoretical Aspects of Computer Science*, volume 775, pages 237–248, 1994.

[48] J. F. Sibeyn, B. S. Chlebus, and M. Kaufmann. Deterministic Permutation Routing on Meshes. *Journal of Algorithms*, 22(1) :111–141, 1997.

[49] A. Srinivasan and C.-P. Teo. A constant-factor approximation algorithm for packet routing, and balancing local vs. global criteria. In *STOC '97 : Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 636–643, New York, NY, USA, 1997. ACM.

[50] I. Stojmenović. Honeycomb Networks : Topological Properties and Communication

Algorithms. *IEEE Transactions on Parallel and Distributed Systems*, 8(10) :1036–1042, 1997.

[51] T. Suel. Routing and Sorting on Meshes with Row and Column Buses. In *Parallel Processing Symposium*, volume Eighth International Volume, pages 411–417, 1994.

[52] R. Tošić, D. Masulović, I. Stojmenović, J. Brunvoll, B. Cyvin, and S. Cyvin. Enumeration of Polyhex Hydrocarbons up to h=17. *Journal of Chemical Information and Computer Sciences*, 35 :181–187, 1995.

[53] R. Trobec. Two-dimensional regular d-meshes. *Parallel Computing*, 26 :1945–1953, 2000.

[54] L. Valiant and G. Brebner. Universal schemes for parallel communication. *Proceedings of the thirteenth annual ACM symposium on Theory of computing*, pages 263–277, 1981.

[55] R. Williams. *The Geometrical Foundation of Natural Structure : A Source Book of Design*. Dover Publications, 1979.

A Defining the embeddings

The results already known for the square grid can be used for a triangular (resp. a hexagonal) grid if we have an adapted function mapping the square grid into the triangular (resp. hexagonal) grid. Here we propose both functions, namely *square2triangle* and *square2hexagon*.

The function *square2triangle* is illustrated in FIG. 7. We perform the same routing as in the grid, i.e. we just ignore the extra diagonal.

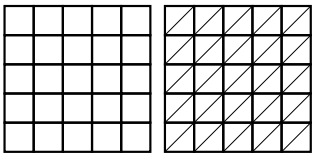


FIG. 7 – Square grid mapped into the triangular grid

In a square grid the distance between two vertices is at most twice the distance in a triangular grid. Similarly the congestion is at most doubled going from the triangular grid to the square grid. Nevertheless the maximal distance is unchanged. Indeed, the NW and SE nodes of FIG. 7 are at the same distance in both grids. Consequently, an algorithm which routes a permutation in $2n-2$ steps is still optimal in the worst case. This is the reason why, instead of considering a square grid with one extra diagonal, we look at a triangle grid in the shape of a triangle, c.f. FIG. 8. Using the routing of the square grid in this triangle grid yields a routing within twice the optimal (i.e. minimum time in the worst case).

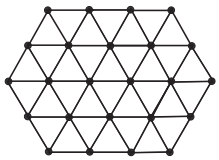


FIG. 8 – Triangular grid

The function *square2hexagon* is a little more complicated. Squares are mapped in two different ways on the hexagonal grid, as shown in FIG. 9. Some are mapped on the left side of a hexagon and some on the right side. Call them respectively *white* and *black* squares. White and black squares alternate on the grid like white and black on a chess board. The missing edge of a white square is mapped to the path of length 3 that goes on the right of the hexagon. The missing edge of a black square is mapped on the path of length 3 that goes on the right of the hexagon. In this way each edge of the square grid is uniquely mapped and each edge of the hexagonal grid is the image of exactly two edges of the square grid.

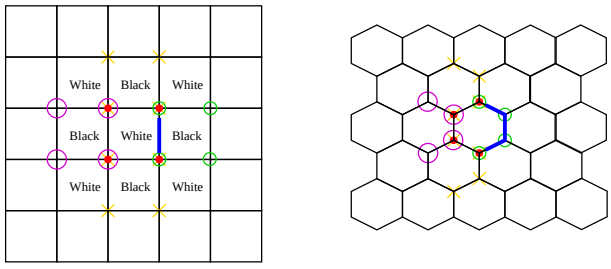


FIG. 9 – Square grid mapped into the hexagonal grid

The distance between 2 vertices in the hexagonal grid is twice the distance in the square grid plus one. Also when we adapt a routing from the square grid to the hexagonal grid using the function *square2hexagon*, the congestion may double since each edge of the hexagonal grid is the image of two edges of the square grid. Consequently, a routing obtained using the function *square2hexagon* will be within a constant multiplicative factor of the optimal.

B An Approach for (ℓ, k)-routing Using Weighted Coloring

In any physical topology, we can represent a given instance of the problem in the following way. Given a network on n nodes, we build a bipartite graph H with a copy of each node at both sides of the bipartition. We add an edge between u and v whenever u wants to send a message to v, and assign to each edge uv a weight $w(uv)$ equal to the length of a shortest path from u to v on the original grid. In this way we obtain an edge-weighted bipartite graph H on $2n$ nodes. Note that the maximum degree of H satisfies $\Delta \leq \max\{\ell, k\}$. An example for $\ell = 2$ and $k = 3$ is depicted in FIG. 10.

The key idea behind this construction is that each matching in H corresponds to an instance of a permutation routing problem. Hence, it can be solved optimally, as we have proved for all types of grids in Section 2. For each matching M_i, we define its cost as $c(M_i) := \max\{w(e)|e \in M_i)\}$. We assign this cost because on all grids the running time of the permutation routing algorithms we have described are proportional to the length of the longest shortest path (with equality on full-duplex triangular grids).

From the classical Hall's theorem we know that the edges of a bipartite graph can be partitioned into Δ disjoint matchings (that is, a coloring of the edges), Δ being the maximum degree of the graph. In our case we have $\Delta = \max\{\ell, k\}$. Thus, the problem consists in partitioning the edges of H into Δ matchings $M_1, \ldots, M_\Delta$, in such a way that $\sum_{i=1}^{\Delta} c(M_i)$ is minimized. That is, our problem, namely WEIGHTED BIPARTITE EDGE COLORING, can be stated in the following way :

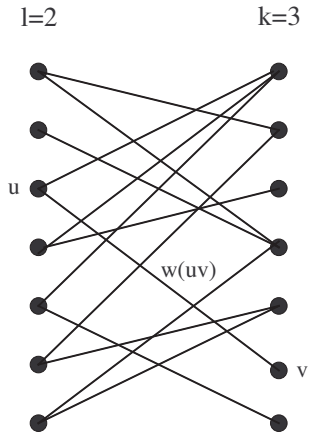


FIG. 10 – Bipartite graph modeling a $(2,3)$-routing instance

WEIGHTED BIPARTITE EDGE COLORING
Input : An edge-weighted bipartite graph H.
Output : A partition of the edges of H into matchings $M_1, \ldots, M_\Delta$, with $c(M_i) := \max\{w(e)|e \in M_i)\}$.
Objective : $\min \sum_{i=1}^{\Delta} c(M_i)$.

Therefore, $\min \sum_{i=1}^{\Delta} c(M_i)$ is the running time for routing an (ℓ, k)-routing instance using this algorithm.

Unfortunately, in [11] WEIGHTED EDGE COLORING is proved to be strongly NP-complete for bipartite graphs, which is the case we are interested in. In fact, the problem remains strongly NP-complete even restricted to cubic and planar bipartite graphs. Concerning approximation results, the authors [11] provide an inapproximability bound of $\frac{7}{6} - \varepsilon$, for any $\varepsilon > 0$. Furthermore, they match this bound with an approximation algorithm within 7/6 on graphs with maximum degree 3, improving the best known approximation ratio of 5/3 [12]. In [24] this innaproximability bound is proved independently on general bipartite graphs. Thus, if $\max\{\ell, k\} \leq 3$ we can find a solution of WEIGHTED BIPARTITE EDGE COLORING within $\frac{7}{6}$ times the optimal solution, and this will be also a solution for the (ℓ, k)-routing problem.

Remark B.1 *Although of theoretical value, the main problem of this algorithm is that finding these matchings is a* ***centralized*** *task. In addition, the* true *ratio, i.e. related to the optimum of the (ℓ, k)-routing, should be proved to provide an upper bound for the running time of this algorithm.*

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399

Improper colouring of weighted grid and hexagonal graphs

Jean-Claude Bermond* Frédéric Havet * Florian Huc * and C. Linhares †

November 21, 2008

1 Introduction

This paper is motivated by a problem posed by Alcatel Space Technologies (see [1]). A satellite sends informations to receivers on earth, each of which is listening several frequencies, one for each signal it needs to receive. Technically it is impossible to focus a signal sent by the satellite exactly on the destination receiver. So part of the signal is spread in an area around it creating noise for the other receivers displayed in this area and listening the same frequency. Each receiver is able to distinguish the signal directed to it from the extraneous noises it picks up if the sum of the noises does not become too big, i.e. does not exceed a certain threshold T. The problem is to assign frequencies to the receivers in such a way that each receiver gets its dedicated signals properly, while minimizing the total number of frequencies used.

Generally the "noise relation" is symmetric, that is if a receiver u is in the noise area of a receiver v then v is in the noise area of u. Hence, interferences may be modelled by a *noise graph* $G = (V(G), E(G))$ which vertices are the receivers and in which two vertices are joined by an edge if and only if they interfer. Moreover, the graph is attached a *weight function* $p : V(G) \to \mathbb{N}$, where the weight $p(v)$ of the vertex v is equal to the number of signals it has to receive. Hence we have a *weighted graph*, that is a pair (G, p) where G is a graph and p a weight function on the vertex set of G. The weight function can be intuitively extended to sets of vertices and to subgraphs: the *weight of a set of vertices* $S \subset V(G)$ is $p(S) = \sum_{v \in S} p(v)$ while the *weight of a subgraph* $H \subset G$ is $p(H) = p(V(H))$. The *maximum weight* of a vertex in a graph (G, p) is denoted $p_{\max}$.

In a simplified version, the intensity I of the noise created by a signal is independent of the frequency and the receiver. Hence to distinguish its signal from noises, a receiver must be in the noise area of at most $k = \lfloor \frac{T}{I} \rfloor$ receivers listening signals on the same frequency. Hence the problem come to find a colouring of the weighted graph which is k-improper. Let (G, p) be a weighted graph. A *colouring* of (G, p) is a function $C : V \to \mathcal{P}(S)$ such that $|C(v)| \geq p(v)$. The set S is called the set of *colours* and is usually $\{1, 2, \ldots, l\}$ for some integer l as we are only interested in its cardinality. A weighted colouring into a set S of cardinality l is called an *l-colouring* of (G, p). A weighted colouring C of (G, p) is *k-improper* if for any colour i, the set of vertices coloured i induces a graph of degree at most k. The *k-improper chromatic number* of (G, p), denoted $\chi_k(G, p)$, is the smallest l such that (G, p) admits a k-improper l-colouring. Note that a 0-improper colouring corresponds to a proper colouring.

*Projet Mascotte I3S (CNRS & UNSA) and INRIA, INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 Sophia-Antipolis Cedex. E-mail: (bermond|fhavet|fhuc)@sophia.inria.fr. Partially supported by the european project FET-AEOLUS.

†Universidade Federal do Ceará, Departamento de Computaçao, Bloco 910, Campus do Pici, Fortaleza, Ceará, CEP 60455-760, Brasil. linhares@lia.ufc.br

In [1] this problem is studied via linear programming. The objective of this paper is to build algorithms giving k-improper colouring of weighted graphs of a certain class $\mathcal{C}$ with as few colours as possible. An algorithm that gives a k-improper colouring of each weighted graph $(G,p) \in \mathcal{C}$ with at most $c_1 \times \chi_k(G,p) + c_2$ colours for some constants c_1 and c_2, is said to be c_1-*approximate* or to have *approximation ratio* c_1.

Let q be an integer. We denote by $\mathbf{q}$ the constant weight function equal to q. A natural method to find a k-improper colouring of (G,p) consists in finding first a k-improper colouring of $(G,\mathbf{q})$ with r colours, ideally $\chi_k(G,\mathbf{q})$, and then in dividing each weight into sets of size q; using r colours for each of these sets, we obtain a k-improper $r \times \left\lceil \frac{p_{\max}}{q} \right\rceil$-colouring of (G,p). As $\chi_k(G,p) \geq p_{\max}$, it gives an (r/q)-approximate algorithm.

Proposition 1 *Given a colouring C of $(G,\mathbf{q})$ with r colours, there is an easy polynomial algorithm that colours (G,p) with at most $r \left\lceil \frac{p_{\max}}{q} \right\rceil$ colours.*

In particular, $\chi_k(G,\mathbf{q}) \leq r$, then $\chi_k(G,p) \leq r \left\lceil \frac{p_{\max}}{q} \right\rceil$.

In this paper, we present improvements of this proposition. For any graph G, we denote $\rho_k(H,G,p)$, or simply $\rho_k(H)$ when (G,p) is clear from the context, the maximum of $\chi_k(H',p)$ over the subgraphs H' of G isomorphic to H. For example, $\rho_k(K_1) = p_{\max}$ and $\rho_0(K_2) = \max\{p(u)+p(v) \mid uv \in E(G)\}$ and $\rho_k(K_2) = p_{\max}$ if $k \geq 2$. By extension, if $\mathcal{H}$ is a family of graphs (finite or not), $\rho_k(\mathcal{H},G,p)$ is the maximum of $\rho_k(H,G,p)$ over all graphs $H \in \mathcal{H}$. Obviously, for any family $\mathcal{H}$, $\rho_k(\mathcal{H},G,p) \leq \chi_k(G,p)$. The idea to design approximate algorithms for k-improper colouring is to find a finite family of graphs $\mathcal{H}_k$ such that any weighted graph (G,p) in the considered class satisfies $\chi_k(G,p) \leq c_1.\rho_k(\mathcal{H}_k) + c_2$ with c_1 a small constant (ideally 1) and c_2 another constant. Hence computing $\chi_k(H',p)$ for all the subgraphs H' isomorphic to a graph in $\mathcal{H}_k$, we obtain a c_1-approximate algorithm for $\chi_k(G,p)$. Moreover we also exhibit algorithms that produce the corresponding c_1-approximate k-improper colouring.

We first show approximate algorithms for general graphs. We then make further improvements for specific graphs, namely the *grid graphs* and the *hexagonal graphs*. The next subsections recall the results known for grid graphs and hexagonal graphs.

1.1 Grid graphs

The (two-dimensionnal) *grid* is the graph GL defined as follows: the vertices are all integer linear combinations $a\mathbf{f_1} + b\mathbf{f_2}$ of the two vectors $\mathbf{f_1} = (1,0)$ and $\mathbf{f_2} = (0,1)$: thus we may identify the vertices with the pairs (a,b) of integers. Two vertices are adjacent when the Euclidean distance between them is 1. Thus each vertex $x = (a,b)$ has four neighbours: its *left neighbour* $(a-1,b)$, its *right neighbour* $(a+1,b)$, its *top neighbour* $(a,b+1)$ and its *down neighbour* $(a,b-1)$. A *grid graph* is an induced subgraph of the two-dimensionnal grid.

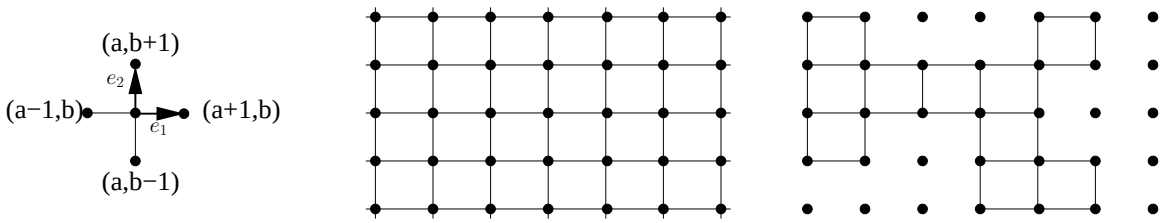


Figure 1: The two-dimensionnal grid and a grid graph.

As a grid graph G has maximum degree 4, for $k \geq 4$, k-improper colouring of G are trivial : giving any

set of $p(v)$ colours to vertex v is a k-improper colouring. Hence, for any $k \geq 4$, its k-improper chromatic number equals its maximum weight $\chi_k(G,p) = p_{\max}$.

Regarding proper (0-improper) colouring, by Proposition 1, $\chi_0(G,p) \leq 2p_{\max}$ as a grid graph is bipartite. This upper bound is tight when there is an edge uv such that $p(u) = p(v) = p_{\max}$. But such an edge may not exist. However, one can find the weighted chromatic number of a grid graph and more generally of any weighted bipartite graph.

Theorem 2 (McDiarmid and Reed [7]) *Let $G = ((A,B),E)$ be a bipartite graph. Then for any weight p, $\chi_0(G,p) = \rho_0(K_1,K_2)$.*

Proof. Let us colour every vertex a of A with $\{1,2,\ldots,p(a)\}$ and every vertex b of B with $\{\rho_0(\{K_1,K_2\}), \rho_0(\{K_1,K_2\})-1,\ldots,\rho_0(\{K_1,K_2\})-p(b)+1\}$. □

Note that $\rho_0(K_1,G,p) = p_{\max}$ and $\rho_0(K_2,G,p) = \max\{p(u)+p(v) \mid uv \in E(G)\} \leq 2p_{\max}$.

In Section 3, for $1 \leq k \leq 3$, we provide an α_k-approximate polynomial-time algorithms that compute a k-improper colouring of a weighted gridgraph with $\alpha_1 = 13/9$, $\alpha_2 = 27/20$ and $\alpha_3 = 19/16$.

It would be nice to prove (and even better disprove) the following problem:

Problem 3 For any fixed $1 \leq k \leq 3$, is it $\mathcal{NP}$-complete to find the k-improper chromatic number of a weighted grid graph?

1.2 Hexagonal graphs

The triangular lattice graph TL may be described as follows. The vertices are all integer linear combinations $a\mathbf{e_1} + b\mathbf{e_2}$ of the two vectors $\mathbf{e_1} = (1,0)$ and $\mathbf{e_2} = (\frac{1}{2}, \frac{\sqrt{3}}{2})$: thus we may identify the vertices with the pairs (a,b) of integers. Two vertices are adjacent when the Euclidean distance between them is 1. Thus each vertex $x = (a,b)$ has six neighbours: its *left neighbour* $(a-1,b)$, its *right neighbour* $(a+1,b)$, its *leftup neighbour* $(a-1,b+1)$, its *rightup neighbour* $(a,b+1)$, its *leftdown neighbour* $(a,b-1)$ and its *rightdown neighbour* $(a+1,b-1)$. A *hexagonal graph* is an induced subgraph of the triangular lattice.

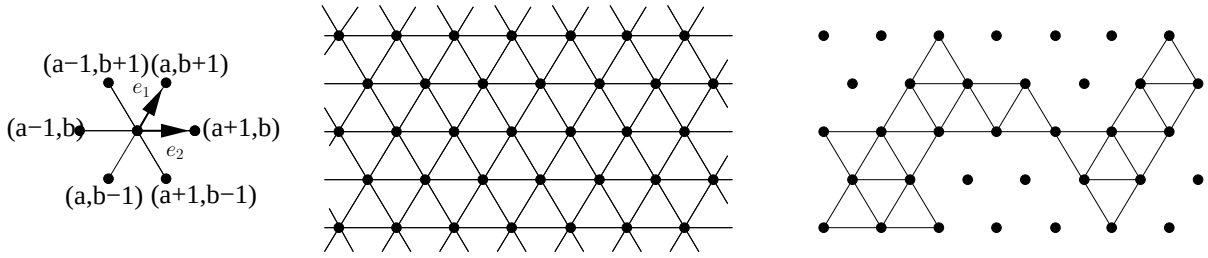


Figure 2: The triangular lattice and a hexagonal graph.

For any $k \geq 6$, the k-improper chromatic number of a hexagonal graph is its maximum weight because it has maximum degree 6.

McDiarmid and Reed [7] showed that it is $\mathcal{NP}$-complete to decide whether the chromatic number of a weighted hexagonal graph is 3 or 4. Hence there is no polynomial time algorithm for finding the weighted chromatic number of hexagonal graphs (unless $\mathcal{P} = \mathcal{NP}$). Hence one has to find approximate algorithms. The better known so far has approximation ratio 4/3 and is based on the following result.

Theorem 4 (McDiarmid and Reed [7]) *For any weigthed hexagonal graph (G,p),*

$$\chi_0(G,p) \leq \frac{4}{3}\rho_0(\{K_1,K_2,K_3\}).$$

A distributed algorithm which guarantees the $\frac{4}{3}\rho_0(\{K_1, K_2, K_3\})$ bound is reported by Narayanan and Schende [6]. However, one expects approximate algorithms with ratios better than 4/3. In particular, Reed and McDiarmid conjecture that for big weight the ratio may be decreased to almost 9/8.

Conjecture 5 (McDiarmid ans Reed [7]) *There exists a constant c such that for any weigthed hexagonal graph (G, p),*

$$\chi_0(G, p) \leq \frac{9}{8}\rho_0(\{K_1, K_2, K_3\}) + c.$$

Note that the ratio 9/8 in the above conjecture is best possible. Indeed consider a 9-cycle C_9 with constant weight k. A colour can be assigned to at most 4 vertices, so $\chi_0(C_9, k) \geq \frac{9k}{4}$. Clearly, $\rho_0(\{K_1, K_2, K_3\}, C_9, k) = 2k$. So $\chi_0(C_9, k) \geq \frac{9}{8}\rho_0(\{K_1, K_2, K_3\}, C_9, k)$. An evidence for this conjecture has been given by Havet [2] who proved that if a hexagonal graph G is triangle-free (i.e. has no K_3) then $\chi_0(G, p) \leq \frac{7}{6}\rho_0(\{K_1, K_2\}) + 5$. See also [9] for an alternative proof and [4] for a distributed algorithm for colouring triangle-free hexagonal graph with $\frac{5}{4}\rho_0(\{K_1, K_2\}) + 3$ colours.

Regarding improper colouring, Havet, Kang and Sereni [3, 8] generalized the above mentionned $\mathcal{NP}$-completeness result of McDiarmid and Reed:

Theorem 6 (Havet, Kang and Sereni [3, 8]) *For $0 \leq k \leq 5$, the following problem is $\mathcal{NP}$-complete:*
Instance: *a weighted hexagonal graph (G, p).*
Question: *is (G, p) k-improper 3-colourable?*

Hence one cannot expect polynomial-time algorithm to find the k-improper chromatic number of weighted hexagonal graphs. In Section 4, for $1 \leq k \leq 5$, we provide an α_k-approximate polynomial-time algorithms that compute a k-improper colouring of a weighted hexagonal graph with $\alpha_1 = 20/11$, $\alpha_2 = 12/7$, $\alpha_3 = 18/13$, $\alpha_4 = 80/63$ and $\alpha_5 = 41/36$.

2 General algorithms

In this section, we improve Proposition 1. To do so, instead of considering only $p_{\max}$, we consider the number of colours that a vertex and its neighbours may require. As shown by the following, this number may be larger than $p_{\max}$. The graph $K_{1,k+1}$ is the graph with $k+2$ vertices and $k+1$ edges linking one vertex, called the *centre* to the $k+1$ others, called *spikes*.

Proposition 7 *For every weight function p, $\chi_k(K_{1,k+1}, p) \geq \dfrac{1}{k+1} \displaystyle\sum_{v \in V(K_{1,k+1})} p(v) = \dfrac{p(K_{1,k+1})}{k+1}$.*

Proof. Let u be the centre of $K_{1,k+1}$ and $v_1, \ldots, v_{k+1}$ its spikes. Consider a k-improper colouring C of $K_{1,k+1}$. For $1 \leq i \leq k+1$, set $q(v_i) = |C(v_i) \setminus C(u)|$. The colouring C uses at least $M = \max_{1 \leq i \leq k+1}\{q(v_i) + p(u)\} \geq p(u) + \frac{1}{k+1}\sum_{i=1}^{k+1} q(v_i)$ colours. But a colour in $C(u)$ is assigned to at most k of the spikes because the colouring is k-improper. Thus $\sum_{i=1}^{k+1} q(v_i) \geq \sum_{i=1}^{k+1} p(v_i) - kp(u)$. It follows $M \geq \frac{1}{k+1}\left(p(u) + \sum_{i=1}^{k+1} p(v_i)\right)$. □

We call $(k+1)$-*star*, or simply *star*, a subgraph of G isomorphic to $K_{1,k+1}$. We set $\theta_k(G, p) = \max\{p(H)/(k+1) \mid H \text{ star of } G\}$ and $\omega_k(G, p) = \max\{p_{\max}, \theta_k(G, p)\}$. According to Proposition 7, $\omega_k(G, p) \leq \rho(\{K_1, K_{1,k+1}\}, G, p) \leq \chi_k(G, p)$.

Theorem 8 *Let $\alpha_k(r, q) = \frac{(k+1)r^2}{(k+2)rq - q^2}$ and $\beta_k(r, q) = \max\{(k+2)r^2 - rq, (k+1)r^2 + krq\}$. There is a algorithm that given a graph G and a k-improper colouring C of $(G, \mathbf{q})$ with r colours, returns a k-improper colouring (G, p) with at most $\alpha_k(r, q) \times \omega_k(G, p) + \beta_k(r, q)$ colours.*

In particular, if $\chi_k(G, \mathbf{q}) \leq r$, then $\chi_k(G, p) \leq \alpha_k(r, q) \times \omega_k(G, p) + \beta_k(r, q)$.

Proof. Consider the following algorithm :

Algorithm 1 0. Initialisation: $(G^0, p^0) := (G, p)$, $S := \emptyset$ and $i = 0$.

1. Add the vertices of low weight to S that will be treated at the end (Step 3):
   $S^i := \{v \in V(G) \mid p^i(v) \leq \gamma_k(r,q)\}$ with $\gamma_k(r,q) = \frac{q}{r}\beta_k(r,q)$, $S := S \cup S^i$ and for all $v \in S^i$, $s(v) := p^i(v)$. $G^{i+1} := G^i - S^i$.

2. If G^{i+1} is not empty:

   2.1. Give to each vertex v of G^{i+1} a certain number $n^i(v)$ of colours among a set of $(k+1)r^2$ colours in such a way that $\omega_k(G^{i+1}, p^i - n^i) \leq \omega_k(G^i, p^i) - (k+2)rq + q^2$.

   2.2. Set $p^{i+1} := p^i - n^i$ and $i := i + 1$ and go to Step 1.

3. Colour $(G\langle S\rangle, s)$ with $\beta_k(r,q)$ colours. It is possible using $\gamma_k(r,q)/q$ times the colouring C as $s_{\max} \leq \gamma_k(r,q)$.

At each Step 2, $(k+1)r^2$ colours are used and ω_k decreases by at least $(k+2)rq - q^2$. Therefore, Algorithm 1 yields a k-improper colouring of (G,p) with at most $\frac{(k+1)r^2}{(k+2)rq-q^2}\omega_k(G,p) + \beta_k(r,q)$ colours.

Let us now describe precisely how to perform Step 2.1. Set $\omega_k = \omega_k(G^i, p^i)$. A *big star* is a star H of G^{i+1} such that $p^i(H) \geq (k+1)\omega_k - q^2$ and a *big vertex* is a vertex such that $p^i(v) > \omega_k - rq$. A *small vertex* is a non-big vertex. It is *goofy* if it is adjacent to a big vertex and *regular* otherwise.

Set $a = (k+1)r - q$. At each step 2.1, we first use a times the colouring C : with ar colours, each vertex receives aq of them. Then we use rq additional colours: they are all assigned to big vertices and regular vertices receive q^2 (using q times C on regular vertices). Hence $n^i(v) = (k+2)rq - q^2$ if v is big, $n^i(v) = (k+1)rq$ if v is regular and $n^i(v) = (k+1)rq - q^2$ if v is goofy.

Notice that in G^{i+1}, no star is made of $k+2$ big vertices. Consequently the previous colouring is indeed k-improper.

Let us now check that $\omega_k(G^{i+1}, p^{i+1}) \leq \omega_k - (k+2)rq + q^2$.

We have $p^{i+1}_{\max} \leq \omega_k - (k+2)rq + q^2$: indeed for a vertex v, $p^{i+1}(v) \leq p^i(v) - n_i(v)$. If v is big $n^i(v) = (k+2)rq - q^2$ and if v is small $n^i(v) \geq (k+1)rq - q^2$ and $p^i(v) \leq \omega_k - rq$. In both cases, $p^{i+1}(v) \leq \omega_k - (k+2)rq + q^2$.

Consider now a star H of G^{i+1}. Each vertex receives at least qa colours. Hence if H is not big,

$$p^{i+1}(H) \leq p^i(H) - (k+2)qa \leq (k+1)\omega_k - q^2 - (k+2)qa = (k+1)(\omega_k - (k+2)rq + q^2).$$

Suppose now that H is big.

Claim 1 *H has a vertex x which is not goofy.*

Proof. Let u be the centre of H and $v_1, \ldots, v_{k+1}$ its spikes with $p^i(v_1) \geq \cdots \geq p^i(v_{k+1})$. Suppose for a contradiction that all the vertices of H are goofy. Then there exists a big vertex v_0 adjacent to u. The vertex v_0 is not one of the v_i for these are small. Let H' be the star with centre u and spikes $v_0, \ldots, v_k$. Set $S = \sum_{j=1}^{k} p^i(v_j)$. We have $p^i(H') = p^i(u) + S + p^i(v_0) \leq (k+1)\omega_k$ and $p^i(v_0) > \omega_k - rq$. Hence $S < k\omega_k + rq - p^i(u)$. But $p^i(v_{k+1}) \leq \frac{S}{k}$ so $p^i(v_{k+1}) < \omega_k + \frac{rq}{k} - \frac{p^i(u)}{k}$.

Now $p^i(H) = p^i(u) + S + p^i(v_{k+1})$ so by the above inequalities, $p^i(H) < (k+1)\omega_k + rq(\frac{k+1}{k}) - p^i(u)/k$. As $u \in G^{i+1}$, we have $p^i(u) \geq \gamma_k(r,q) \geq (k+1)rq + kq^2$, thus $p^i(H) < (k+1)\omega_k - q^2$. This contradicts the fact that H is big. □

The vertex x receives at least $qa + q^2$ colours. Hence

$$p^{i+1}(H) \leq p^i(H) - (k+2)qa - q^2 \leq (k+1)\omega_k - (k+2)qa - q^2 \leq (k+1)(\omega_k - (k+2)rq + q^2).$$

Thus $\theta_k(G^{i+1}, p^{i+1}) \leq \omega_k - (k+2)rq + q^2$, so $\omega_k(G^{i+1}, p^{i+1}) \leq \omega_k - (k+2)rq + q^2$.

Algorithm 1 requires to compute the value of $\omega_k(G,p)$. Since there are at most $n\binom{\Delta}{k+1}$ $k+1$-stars in a graph on n vertices with maximum degree Δ, it can be done in $O\left(n\binom{\Delta}{k+1}\right)$ operations. □

For 1-improper colouring or when the maximum degree Δ of the graph G is $k+1$, one can get better approximate algorithms. They are very similar to Algorithm 1 but one can improve Claim 1 and thus Theorem 8.

Theorem 9 *Let r and q be two integers. Set $a = 2r - 2q$ if $r \geq 2q$ and $a = r$ if $r \leq 2q$, and $\alpha'_1(r,q) = \frac{ar+rq}{aq+rq}$ and $\beta'_1(r,q) = 4r^2$. There is a polynomial algorithm that given a weighted graph G and a 1-improper colouring C of $(G, \mathbf{q})$ with r colours, produces a 1-improper colouring of (G,p) with at most $\alpha'_1(r,q) \times \omega_k(G,p) + \beta'_1(r,q)$ colours.*

In particular, if $\chi_1(G, \mathbf{q}) \leq r$, then $\chi_1(G,p) \leq \alpha'_1(r,q) \times \omega_1(G,p) + \beta'_1(r,q)$.

Proof. Consider the following algorithm :

Algorithm 2 0. Initialisation: $(G^0, p^0) := (G,p)$, $S := \emptyset$ and $i = 0$.

1. Add the vertices of low weight to S that will be treated at the end (Step 3):
   $S^i := \{v \in V(G) \mid p^i(v) \leq 4rq\}$, $S := S \cup S^i$ and for all $v \in S^i$, $s(v) := p^i(v)$. $G^{i+1} := G^i - S^i$.
2. If G^{i+1} is not empty:
   2.1. Give to each vertex v of G^{i+1} a certain number $n^i(v)$ of colours among a set of $ar + rq$ colours in such a way that $\omega_1(G^{i+1}, p^i - n^i) \leq \omega_1(G^i, p^i) - aq - rq$.
   2.2. Set $p^{i+1} := p^i - n^i$ and $i := i + 1$ and go to Step 1.
3. Colour $(G\langle S\rangle, s)$ with $4r^2$ colours. It is possible using $4r$ times the colouring C as $s_{\max} \leq 4rq$.

Algorithm 2 yields a 1-improper colouring of (G,p) with at most $\alpha'_1\omega_1(G,p) + \beta'_1$ colours.

Let us now describe precisely how to perform Step 2.1. Set $\omega_1 = \omega_1(G^i, p^i)$. As before, a *big vertex* is a vertex such that $p^i(v) > \omega_1 - rq$. A *small vertex* is a non-big vertex. It is *goofy* if it is adjacent to a big vertex and *regular* otherwise. A 2-star H is *big* if $p^i(H) > 2\omega_1 - 2rq$.

At each step 2.1, we first use a times the colouring C : with ar colours, each vertex receives aq of them. Then we use rq additional colours: they are all assigned to big and regular vertices receive q^2 of them (using q times C on regular vertices). Hence $n^i(v) = aq + rq$ if v is big, $n^i(v) = aq + q^2$ if v is regular and $n^i(v) = aq$ if v is goofy.

Let us now check that $\omega_1(G^{i+1}, p^{i+1}) \leq \omega_1 - aq - rq$.

We have $p^{i+1}_{\max} \leq \omega_1 - aq - rq$: indeed for a vertex v, $p^{i+1}(v) \leq p^i(v) - n_i(v)$. If v is big $n^i(v) = aq + rq$ and if v is small $n^i(v) \geq aq$ et $p^i(v) \leq \omega_1 - rq$. In both cases, $p^{i+1}(v) \leq \omega_1 - aq - rq$.

Consider a 2-star H of G^{i+1}. Each vertex receives at least qa colours. Hence if H is not big then $p^{i+1}(H) \leq p^i(H) - 3aq \leq 2\omega_1 - 2rq - arq$.

Suppose now that H is big.

Claim 2 *A 2-star with two goofy vertices and no big vertex is not big.*

Proof. Suppose H is a 2-star with centre u_2 and spikes u_1 and u_3 and assume that two vertices are goofy and none is big. W.l.o.g. there is a big vertex b_1 adjacent to u_1 and a big vertex b_2 adjacent to u_2 or u_3. As b_1, u_1 and u_2 form a $K_{1,2}$, by definition of ω_1, $\frac{1}{2}(p^i(b_1)+p^i(u_1)+p^i(u_2)) \leq \omega_1$. Similarly, $\frac{1}{2}(p^i(b_2)+p^i(u_2)+p^i(u_3)) \leq \omega_1$. Hence $p^i(u_1)+p^i(u_2)+p^i(u_3) \leq 4\omega_1 - p^i(b_1) - p^i(b_2) - p^i(u_2) \leq 2\omega_1 - p^i(u_2) + 2rq - 2$. As u_2 is in G^{i+1}, $p^i(u_2) \geq 4rq-2$, so $p^i(H) \leq 2\omega_1 - 2rq$. □

So H contains a big vertex or two regular ones. Hence $p^{i+1}(H) \leq p^i(H) - 3qa - \min\{rq, 2q^2\}$. By our choice of a, we get $p^{i+1}(H) \leq 2(\omega_1 - aq - rq)$. □

Theorem 10 *Let r and q be two integers. Set $a = (k+1)r - 2$ if $r \geq 2q$ and $a = kr$ if $r \leq 2q$, and $\alpha''_k(r,q) = \frac{ar+rq}{aq+rq}$ and $\beta''_k(r,q) = ar + r^2$. There exists a polynomial algorithm that given a graph G with maximum degree $k+1$ and a k-improper colouring C of $(G, \mathbf{q})$ with r colours, produces a k-improper colouring of (G,p) with at most $\alpha''_k(r,q) \times \omega_k(G,p) + \beta''_k(r,q)$ colours.*

In particular, if $\chi_k(G, \mathbf{q}) \leq r$, then $\chi_k(G,p) \leq \alpha''_k(r,q) \times \omega_k(G,p) + \beta''_k(r,q)$.

Proof. Consider the following algorithm :

Algorithm 3 0. Initialisation: $(G^0, p^0) := (G,p)$, (S,s) is the empty graph and $i = 0$.

1. Add the vertices of low weight to S that will be treated at the end (Step 3):
   $S^i := \{v \in V(G) \mid p^i(v) \leq aq + rq$, $S := S \cup S^i$ and for all $v \in S^i$, $s(v) := p^i(v)$. $G^{i+1} := G^i - S^i$.
2. If G^{i+1} is not empty:
   2.1. Give to each vertex v of G^{i+1} a certain number $n^i(v)$ of colours among a set of $(a+q)r$ colours in such a way that $\omega_k(G^{i+1}, p^i - n^i) \leq \omega_k(G^i, p^i) - aq - rq$.
   2.2. Set $p^{i+1} := p^i - n^i$ and $i := i+1$ and go to Step 1.
3. Colour $(G\langle S\rangle, s)$ with β''_k colours. It is possible using $a+r$ times the colouring C as $s_{\max} \leq (a+r)q$.

Algorithm 3 yields a k-improper colouring of (G,p) with at most $\alpha''_k \omega_k(G,p) + \beta''_k$ colours.

Let us now describe precisely how to perform Step 2.1. Set $\omega_k = \omega_k(G^i, p^i)$. A *big vertex* is a vertex such that $p^i(v) > \omega_k - rq$. A *small vertex* is a non-big vertex. It is *goofy* if it is adjacent to a big vertex and *regular* otherwise. A regular vertex is *isolated* if it is adjacent to no regular vertex.

At each step 2.1, we first use a times the colouring C : with ar colours, each vertex receives aq of them. Then we use rq additional colours: they are all assigned to big and isolated regular vertices and non-isolated regular vertices receive q^2 (using q times C on non-isolated regular vertices). Hence $n^i(v) = aq + rq$ if v is big or isolated regular, $n^i(v) = aq + q^2$ if v is non-isolated regular and $n^i(v) = aq$ if v is goofy.

Let us now check that $\omega_k(G^{i+1}, p^{i+1}) \leq \omega_k - aq - rq$.

We have $p^{i+1}_{\max} \leq \omega_k - aq - rq$: indeed for a vertex v, $p^{i+1}(v) \leq p^i(v) - n_i(v)$. If v is big $n^i(v) = aq + rq$ and if v is small $n^i(v) \geq aq$ et $p^i(v) \leq \omega_k - rq$. In both cases, $p^{i+1}(v) \leq \omega_k - aq - rq$.

Consider a star H of G^{i+1}. Each vertex receives at least aq colours.

Claim 3 *A star H contains a big vertex or an isolated regular vertex x or two non-isolated regular vertices.*

Proof. Let u be the centre of H and $v_1, \ldots, v_{k+1}$ its spikes. Since G has maximum degree $k+1$ the only neighbours of u are the v_i. Hence, if no vertex of H is big then u is regular. Moreover if it is not isolated one of the v_i is also regular. □

Hence H contains a vertex that receives at least $aq+rq$ colours or two vertices receiving at least $aq+q^2$ colours. Hence $p^{i+1}(H) \leq p^i(H) - (k+2)qa - \min\{rq, 2q^2\}$. By our choice of a, we get $p^{i+1}(H) \leq (k+1)(\omega_k - aq - rq)$. Thus $\theta_k(G^{i+1}, p^{i+1}) \leq \omega_k - aq - rq$, so $\omega_k(G^{i+1}, p^{i+1}) \leq \omega_k - aq - rq$. □

3 Grid graphs

3.1 General algorithms applied to grid graphs

In order to apply Theorems 8 and 10 to grid graphs, we determine $\chi_k(GL, \mathbf{q})$ for every positive integer q and $1 \leq k \leq 3$.

We first prove a preliminary lemma. Let C be a colouring of a weighted graph (G, p). We denote by $c_{u,v}$ the number of colours assigned to both u and v, that is $c_{u,v} = |C(u) \cap C(v)|$. We use the standard notation $N(u)$ for the *neighborhood* of u.

Lemma 11 *Let C be a k-improper colouring of a weighted graph (G, p).*

(i) $\sum_{v \in N(u)} c_{u,v} \leq kp(u)$;

(ii) $\forall u, v$, *C uses at least* $p(u) + p(v) - c_{u,v}$ *colours;*

(iii) $\forall u, v, w$, *C uses at least* $p(u) + p(v) + p(w) - c_{u,v} - c_{u,w} - c_{v,w}$.

Proof. (i) A colour assigned to u is assigned to at most k neighbours of u because C is k-improper.

(ii) and (iii) follow from the Inclusion-Exclusion Formula:

$$|C(u) \cup C(v)| = |C(u)| + |C(v)| - |C(u) \cap C(v)| = p(u) + p(v) - c_{u,v};$$

$$\begin{aligned} |C(u) \cup C(v) \cup C(w)| &= |C(u)| + |C(v)| + |C(w)| - |C(u) \cap C(v)| - |C(u) \cap C(w)| - |C(v) \cap C(w)| \\ &\quad + |C(u) \cap C(v) \cap C(w)| \\ &\geq p(u) + p(v) + p(w) - c_{u,v} - c_{u,w} - c_{v,w}. \end{aligned}$$

□

Theorem 12 *For the grid graph GL, we have:*

(i) $\chi_1(GL, \mathbf{q}) = 2q$,

(ii) $\chi_2(GL, \mathbf{q}) = \lceil \frac{3q}{2} \rceil$, *and*

(iii) $\chi_3(GL, \mathbf{q}) = \lceil \frac{5q}{4} \rceil$.

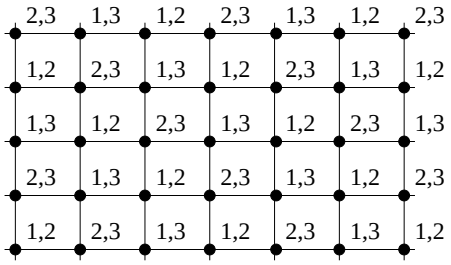


Figure 3: A 2-improper colouring of the square grid.

Proof. Let us first show the k-improper colourings of $(GL, \mathbf{q})$ with the required number of colours.

(i) The grid is bipartite so $(GL, \mathbf{q})$ has a 0-improper (and thus also 1-improper) colouring with $2q$ colours.

(ii) For $1 \leq j \leq 3$, let $U_j = \{(a, b) \mid a + b = j \mod 3\}$. Assign the colours $\left\lceil \frac{(j-1)q}{2} \right\rceil + 1, \ldots, \lceil \frac{jq}{2} \rceil$ to vertices which are not in U_j. See Figure 3.

(iii) For $1 \leq j \leq 5$, let T_j be the set of vertices obtained from the vertex $(0, j)$ by adding the linear combinations of the vectors $2\mathbf{f_1} + \mathbf{f_2}$ and $5\mathbf{f_1}$. For $1 \leq j \leq 5$, assign the colours $\left\lceil \frac{(j-1)q}{4} \right\rceil + 1, \ldots, \lceil \frac{jq}{4} \rceil$ to vertices not in T_j. See Figure 4.

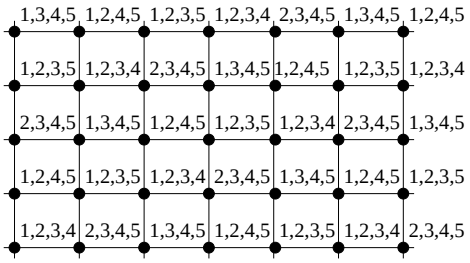


Figure 4: A 3-improper colouring of the square grid.

Let us now show that these colourings are optimal. Hence let C be a k-improper colouring of $(GL, \mathbf{q})$ with $\chi_k(GL, \mathbf{q})$ colours.

(i) If $k = 1$, consider a 4-cycle in GL. A colour may be used on at most 2 of these vertices. Hence $2q$ colours are needed.

(ii) and (iii) Let u be a vertex of GL. Applying Lemma 11 (ii) to the four neighbours of a vertex u, we obtain: $4\chi_k(GL, \mathbf{q}) \geq 8q - \sum_{v \in N(u)} c_{u,v}$. Now by Lemma 11 (i), $4\chi_k(GL, \mathbf{q}) \geq (8 - k)q$. We obtain respectively $\chi_2(GL, \mathbf{q}) \geq \frac{3q}{2}$ and $\chi_3(GL, \mathbf{q}) \geq \frac{5q}{4}$. □

Corollary 13 *For $1 \leq k \leq 3$, there are α_k-approximate algorithms for finding a k-improper colouring of a weighted grid graph, where $\alpha_1 = \frac{3}{2}$, $\alpha_2 = \frac{27}{20}$, and $\alpha_3 = \frac{19}{16}$.*

Proof. Theorems 9 and 12 give the result for $k = 1$. Theorems 8 and 12 give the result for $k = 2$. Theorems 10 and 12 give the result for $k = 3$. □

The following theorem improves this last result for 1-improper colouring of weighted grid graphs:

Theorem 14 *There is a $\frac{39}{27}$-approximate algorithm for finding a 1-improper colouring of weighted grid graph.*

Proof. The idea of the algorithm is still similar but this time we use more the position of big vertices and quasi-big vertices (a new type of vertex to be described) to 1-improper colour the grid graph (cf Lemma 4).

Algorithm 4 0. Initialisation: $(G^0, p^0) := (G, p)$, $S := \emptyset$ and $i = 0$.

1. Add the vertices of low weight to S that will be treated at the end (Step 3):
   $S^i := \{v \in V(G) \mid p^i(v) < p_{\inf}\}$ with $p_{\inf} = 133$, $S := S \cup S^i$ and for all $v \in S^i$, $s(v) := p^i(v)$.
   $G^{i+1} := G^i - S^i$.

2. If G^{i+1} is not empty:

   2.1. Give to each vertex v of G^{i+1} a certain number $n^i(v)$ of colours among a set of 156 colours in such a way that $\omega_k(G^{i+1}, p^i - n^i) \leq \omega_k(G^i, p^i) - 108$.

   2.2. Set $p^{i+1} := p^i - n^i$ and $i := i + 1$ and go to Step 1.

3. Colour $(G\langle S\rangle, s)$ with 266 colours. It is possible using as $s_{\max} \leq p_{\inf} - 1$ and $\chi_1(GL, q) = 2q$.

Algorithm 4 yields a 1-improper colouring of (G, p) with at most $\frac{156}{108}\omega_k(G, p) + 266$ colours.

Let us now describe precisely how to perform Step 2.1. Set $\omega_1 = \omega_1(G^i, p^i)$. A *big vertex* is a vertex such that $p^i(v) > \omega_1 - d_1$ with $d_1 = 24$. A *small vertex* v is a non-big vertex. It is *goofy* if it is adjacent to a big vertex (we note *Goof* the set of goofy vertices), it is quasi-big if $p^i(v) > \omega_1 - d_2$ with $d_2 = 67$ and *regular* otherwise. A 2-star is *big* if its weight is bigger than $2\omega_1 - d_3$ with $d_3 = 45$.

Claim 4 *Let u and v be two vertices which are big or quasi-big in G^i. In G^{i+1}, they are at distance at least 3 or form a connected component.*

Proof. Suppose that u and v are at distance at most 2. If they do not form a connected component in G^{i+1}, there is a vertex $w \in V(G^{i+1})$ such that the subgraph induced by u, v and w is a 2-star. Hence $p^i(u) + p^i(v) + p^i(w) \leq 2\omega_1$. But $p^i(u) + p^i(v) + p^i(w) > 2\omega_1 - 2d_2 + 1 + p_{\inf} = 2\omega_1$, a contradiction. Thus u and v form a connected component. □

Set $a = 48$, $b = 6$ and $c = 3$. At each step 2.1, we give all the $2a + 8b + 4c$ colours to the vertices in connected component of order 2.

For the vertices not in such small components, we use three different colourings. The two first are based on the following eight sets of vertices:

- $U_1 = \{(0,0) + 2k\mathbf{f_1} + k'(\mathbf{f_1} - 2\mathbf{f_2}), (0,1) + 2k\mathbf{f_1} + k'(\mathbf{f_1} - 2\mathbf{f_2})\}$,
- $U_2 = \{(1,0) + 2k\mathbf{f_1} + k'(\mathbf{f_1} - 2\mathbf{f_2}), (1,1) + 2k\mathbf{f_1} + k'(\mathbf{f_1} - 2\mathbf{f_2})\}$,
- $U_3 = \{(0,1) + 2k\mathbf{f_1} + k'(\mathbf{f_1} - 2\mathbf{f_2}), (0,2) + 2k\mathbf{f_1} + k'(\mathbf{f_1} - 2\mathbf{f_2})\}$,
- $U_4 = \{(1,1) + 2k\mathbf{f_1} + k'(\mathbf{f_1} - 2\mathbf{f_2}), (1,2) + 2k\mathbf{f_1} + k'(\mathbf{f_1} - 2\mathbf{f_2})\}$,
- $U_5 = \{(0,0) + 2k\mathbf{f_2} + k'(2\mathbf{f_1} - \mathbf{f_2}), (1,0) + 2k\mathbf{f_2} + k'(2\mathbf{f_1} - \mathbf{f_2})\}$,
- $U_6 = \{(0,1) + 2k\mathbf{f_2} + k'(2\mathbf{f_1} - \mathbf{f_2}), (1,1) + 2k\mathbf{f_2} + k'(2\mathbf{f_1} - \mathbf{f_2})\}$,
- $U_7 = \{(1,0) + 2k\mathbf{f_2} + k'(2\mathbf{f_1} - \mathbf{f_2}), (2,0) + 2k\mathbf{f_2} + k'(2\mathbf{f_1} - \mathbf{f_2})\}$,

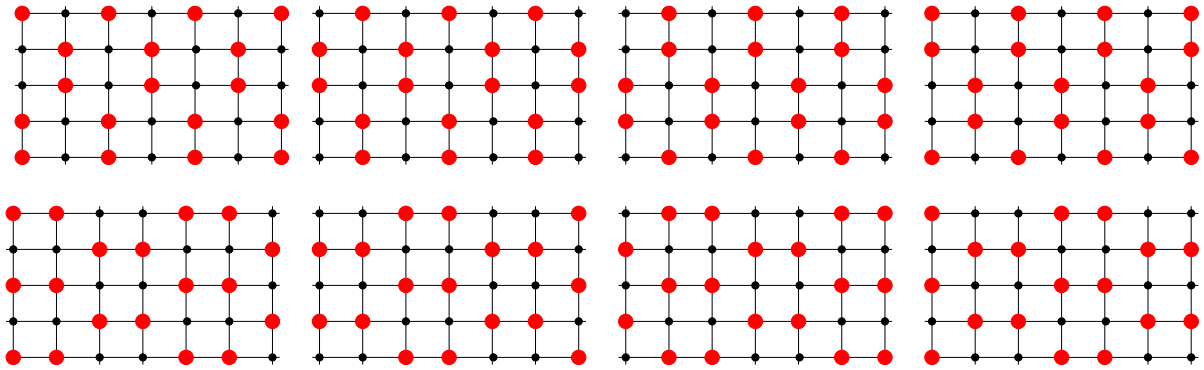


Figure 5: The sets U_1 up to U_8.

- $U_8 = \{(1,1) + 2k\mathbf{f_2} + k'(2\mathbf{f_1} - \mathbf{f_2}), (2,1) + 2k\mathbf{f_2} + k'(2\mathbf{f_1} - \mathbf{f_2})\}$.

We first assign $2a$ colours such that each vertex of U_1 receive a colours and each vertex of U_2 receives a other colours. Since U_1 and U_2 form a partition of V, each vertex receives a colours, and since $G\langle U_1\rangle$ and $G\langle U_2\rangle$ have maximum degree 1, the colouring we obtain is 1-improper.

Then we use $8b$ colours denoted by $(i,j), 1 \leq i \leq b, 1 \leq j \leq 8$. We give all the colours to the big vertices. Then, for $1 \leq j \leq 8$, we give to each vertex of $U_j \setminus Goof$ the colours $(i,j), 1 \leq i \leq b$. A vertex appears in four of the U_j, so each non-goofy vertex receives $4b$ colours. Finally, by Claim 4, a goofy vertex u has a unique big neigbour v. There is an integer j such that both u and v are in U_j. We assign to u the colours $(i,j), 1 \leq i \leq b$. Doing so, each goofy vertex receives b colours.

Finally we use 4 sets of c colours A_t, A_d, A_l and A_r. Each big or quasi-big vertex receive the colour of all these sets and each top (resp. down, left, right) receives the colour of A_t (resp. A_d, A_l and A_r). This is 1-improper by Claim 4.

Let us now check that $\omega_1(G^{i+1}, p^{i+1}) \leq \omega_1 - 108$.

Let v be a vertex. If it is big then $n^i(v) = a + 8b + 4c$; if it is quasi-big then $n^i(v) = a + 4b + 4c$; if it is goofy then $n^i(v) = a + b + c$; if it is regular then $n^i(v) \geq a + 4b$ and if in addition it is adjacent to a quasi-big vertex $n^i(v) \geq a + 4b + c$. Hence, by our choice of a, b, c d_1 and d_2 we have

$$p^{i+1}_{\max} \leq \omega_1 - \min\{a + 8b + 4c, d_1 + a + 4b + 4c, d_2 + a + b + c, d_2 + a + 4b\} \leq p^i_{\max} - 108. \qquad (1)$$

Claim 5 *Let H be a big 2-star in (G^{i+1}, p^i). Then the following hold:*

(i) if the centre of H is goofy then H has a big vertex;

(ii) if H has two goofy vertices then the third one is big;

(iii) if H has a goofy vertex then it has a big or quasi-big vertex.

Proof. Let v be the centre of H and u_1 and u_2 its spike.

(i) Suppose that v is goofy. Let w be its big neighbour. Suppose for a contradiction that $w \notin \{u_1, u_2\}$. Considering the two 2-stars with vertex sets $\{w, v, u_1\}$ and $\{w, v, u_2\}$, we obtain that $p^i(u_1) + p^i(v) \leq 2\omega_1 - p^i(w) \leq \omega_1 + d_1 - 1$ and $p^i(u_2) + p^i(v) \leq 2\omega_1 - p^i(w) \leq \omega_1 + d_1 - 1$. Hence $2\omega_1 + 2d_1 - 2 \geq p^i(u_1) + p^i(u_2) + 2p^i(v) \geq p^i(v) + p^i(H) \geq p^i(v) + 2\omega_1 - d_3 + 1$, so $p^i(v) \leq 2d_1 + d_3 - 3$, but then v is in S^i, which is a contradiction.

(ii) Suppose for a contradiction that H has two goofy vertices and no big vertex. By (i), u_1 and u_2 are the goofy vertices. Let w_1 (resp. w_2) be the big neighbour of u_1 (resp. u_2). (We may have

$w_1 = w_2$). As in (i), considering the two 2-stars with vertex sets $\{w_1, u_1, v\}$ and $\{w_2, u_2, v\}$, we obtain $p^i(u_1) + p^i(v) \leq \omega_1 + d_1 - 1$ and $p^i(u_2) + p^i(v) \leq \omega_1 + d_1 - 1$. These inequalities yield the contradiction as in (i).

(iii) If v is goofy then H has a big vertex by (i). Suppose now that one spike of H, say u_1, is goofy. Let w be the big neighbour of u_1. If $w = v$, we have the result. Assume now that $w \neq v$. Considering the 2-star induced by $\{w, u_1, v\}$, we obtain $p^i(u_1) + p^i(v) \leq \omega_1 + d_1 - 1$. Since H is big $p^i(u_1) + p^i(v) + p^i(u_2) \geq 2\omega_1 - d_3 + 1$. So $p^i(u_2) \geq \omega_1 - d_1 - d_3 + 2$, that is u_2 is quasi-big. □

Let H be a 2-star. If it is not big, its weigth decreases by at least $3a + 3b + 3c$. If H is big then by Claim 5, it has either three non-goofy vertices, in which case its weight decreases by at least $3a + 12b$, one big vertex and two goofy vertices, in which case its weight decreases by $3a + 10b + 6c$, one big vertex, one regular vertex and one goofy vertex, in which case its weight decreases by $3a + 13b + 5c$ or one quasi-big vertex, one regular vertex adjacent to this quasi-big and one goofy vertex in which case its weight decreases by $3a + 9b + 6c$. Hence by our choice of a, b, c, for a big star H,

$$p^{i+1}(H) \leq \omega_1 - \min(3a + 12b, 3a + 9b + 6c, 3a + 13b + 5c) \leq p^i(H) - 216. \quad (2)$$

For Equations (1) and (2) yield $\omega_1(G^{i+1}, p^{i+1}) \leq \omega_1 - 108$. □

4 Hexagonal graphs

4.1 General algorithms applied to hexagonal graphs

Theorem 15 *For the triangular lattice TL, we have:*

(i) $\chi_1(TL, \mathbf{q}) = \lceil \frac{5q}{2} \rceil$,

(ii) $\chi_2(TL, \mathbf{q}) = 2q$,

(iii) $\chi_3(TL, \mathbf{q}) = \lceil \frac{3q}{2} \rceil$,

(iv) $\chi_4(TL, \mathbf{q}) = \lceil \frac{4q}{3} \rceil$ *and*

(v) $\chi_5(TL, \mathbf{q}) = \lceil \frac{7q}{6} \rceil$.

Proof. Let us first show the k-improper colourings of $(TL, \mathbf{q})$ with the required number of colours.

(i) For $1 \leq j \leq 5$, let A_j be the set of vertices obtained from the vertex $(0, j)$ by adding the linear combinations of the vectors $2\mathbf{e_1} + \mathbf{e_2}$ et $5\mathbf{e_1}$. For $1 \leq j \leq 5$, assign the colours $\left\lceil \frac{(j-1)q}{2} \right\rceil + 1, \ldots, \lceil \frac{jq}{2} \rceil$ to vertices of $A_j \cup A_{j+1}$ (with $A_6 = A_1$).

(ii) Colour a vertex (a, b) with $1, \ldots, q$ if a is even and with $q + 1, \ldots, 2q$ otherwise.

(iii) For $1 \leq j \leq 3$, let S_j be the set of vertices obtained from the vertex $(0, j)$ by adding the linear combinations of the vectors $\mathbf{e_1} + \mathbf{e_2}$ et $3\mathbf{e_1}$. Assign the colours $\left\lceil \frac{(j-1)q}{2} \right\rceil + 1, \ldots, \lceil \frac{jq}{2} \rceil$ to vertices which are not in S_j.

(iv) For $0 \leq j_1 \leq 1$ and $1 \leq j_2 \leq 2$, $T_{j_1+j_2} = \{(a, b) \mid a \equiv j_1 \mod 2 \text{ et } b \equiv j_2 \mod 2\}$. Assign the colours $\left\lceil \frac{(j-1)q}{3} \right\rceil + 1, \ldots, \lceil \frac{jq}{3} \rceil$ to vertices which are not in T_j.

(v) For $1 \leq j \leq 7$, let U_j be the set of vertices obtained from the vertex $(0, j)$ by adding the linear combinations of the vectors $\mathbf{e_1} + 2\mathbf{e_2}$ et $7\mathbf{e_1}$. Assign the colours $\left\lceil \frac{(j-1)q}{6} \right\rceil + 1, \ldots, \lceil \frac{jq}{6} \rceil$ to vertices which are not in U_j.

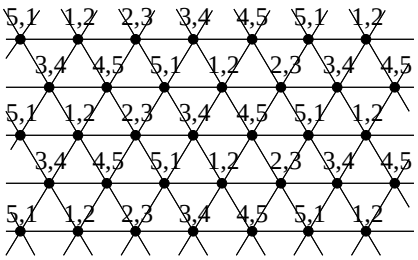


Figure 6: A 1-improper colouring of the triangular lattice.

Let us now show that these colourings are optimal. Hence let C be a k-improper colouring of $(TL, \mathbf{q})$ with $\chi_k(TL, \mathbf{q})$ colours.

Let u be a vertex of TL. Applying Lemma 11 (i) and (ii) to a vertex u of TL and its six neighbours, we obtain: $6\chi_k(TL, \mathbf{q}) \geq (12-k)q$. For $k = 3$, 4 or 5, we obtain respectively $\chi_3(TL, \mathbf{q}) \geq \frac{3q}{2}$, $\chi_4(TL, \mathbf{q}) \geq \frac{4q}{3}$ and $\chi_5(TL, \mathbf{q}) \geq \frac{7q}{6}$.

For $k = 1$ and 2, we need an extra argument. Let u and v be two adjacent vertices and t and w their two common neighbours. Set $f(u, v) = c_{u,t} + c_{u,v} + c_{u,w}$.

Consider now the ordered pair (u, v) which minimizes $f(u, v)$, i.e. $f(u, v) = f_{min}$. Then $f(v, u) = c_{v,t} + c_{u,v} + c_{v,w}$. So $f(u, v) + f(v, u) = (c_{u,t} + c_{u,v} + c_{v,t}) + (c_{u,w} + c_{u,v} + c_{v,w})$. By Lemma 11 (iii), we have $f(u, v) + f(v, u) \geq 6q - 2\chi_k(TL, \mathbf{q})$.

Let u' be the vertex symmetrical to u compared to v. By Lemma 11 (i), $f(v, u) + f(v, u') \leq kq$, so $f(v, u') \leq kq - f(v, u) \leq kq - 6q + 2\chi_k(TL, \mathbf{q}) + f_{min}$.

For $k = 1$, if $\chi_k(TL, \mathbf{q}) < \frac{5q}{2}$, we would have $f(v, u') < f_{min}$. Similarly, for $k = 2$, if $\chi_k(TL, \mathbf{q}) < 2q$, we would have $f(v, u') < f_{min}$. In both cases, it contradicts the minimality of f in (u, v). □

To prove the optimality of the colouring proposed for $k = 1$ and $k = 2$, one can also use Lemma 16. Indeed this Lemma says that for $k = 1$ for every two vertices of a given colour, there is at least three vertices that do not have this colour. For $k = 2$, it says that for every ℓ vertices of a given colour there is at least ℓ vertices that do not have this colour.

Lemma 16 *Let P be a path in the triangular lattice made of ℓ vertices such that any vertex of P is adjacent to atmost 2 vertices of P. P has exactly $\ell + 1$ leftdown and rightdown neighbours.*

Let C be a cycle in the triangular lattice made of ℓ vertices such that any vertex of C is adjacent to exactly 2 vertices of C. C has exactly ℓ leftdown and rightdown neighbours.

Proof. We prove both assertions by induction: $\mathcal{P}(\ell)$: "Let P be a path in the triangular lattice made of ℓ vertices, such that any vertex of P is adjacent to atmost 2 vertices of P. P has exactly $\ell + 1$ leftdown and rightdown neighbours."

$\mathcal{P}(1)$ is true.

Let $\ell \geq 2$, $\mathcal{P}(\ell) \Rightarrow \mathcal{P}(\ell + 1)$: Let v be one of the two end verices of P. $P \setminus \{v\}$ has ℓ vertices, so by $\mathcal{P}(\ell)$, it has $\ell + 1$ leftdown and rightdown neighbours. According to the 6 positions of v in respect to P as depicted on Figure 4, one can see that P has exactly one more leftdown and rightdown neighbour than $P \setminus \{v\}$. (Either v is leftdown or rightdown neighbour of $P \setminus \{v\}$ and has two neighbours which weren't leftdown or rightdown neighbour of $P \setminus \{v\}$ or it is not and v brings one new leftdown or rightdown neighbour. In the figures, we use the following code: there is six pairs of grids, in each pairs, the left grid correspond to P and the right grid to $P \setminus \{v\}$. v is represented in full purple, it's only neighbour in full red. The common neighbours of P and $P \setminus \{v\}$ are in dashed empty red cycles while the different ones are in empty purple cycles.)

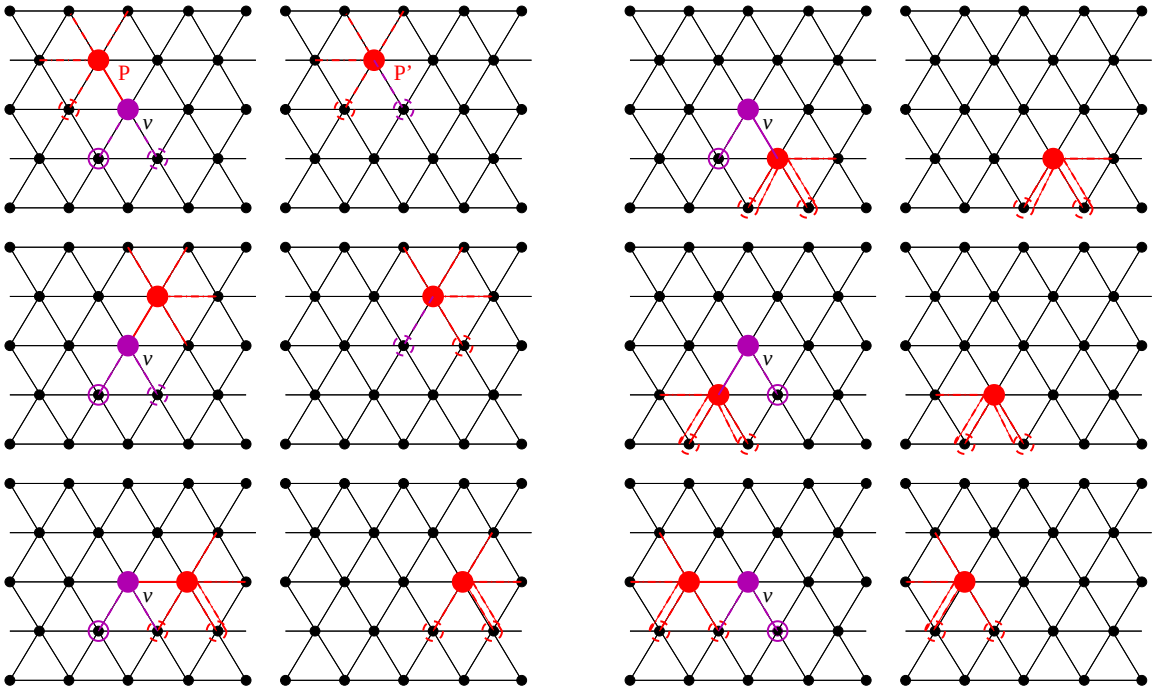


Figure 7: The 6 possible positions of an end vertex of P.

$\mathcal{P}'(\ell)$: "Let C be a cycle in the triangular lattice made of ℓ vertices, such that any vertex of C is adjacent to exactly 2 vertices of C. c has exactly ℓ leftdown and rightdown neighbours."

$\mathcal{P}'(3)$ and $\mathcal{P}'(6)$ are true.

Let $\ell \geq 7$, $\forall \ell' \leq \ell, \mathcal{P}'(\ell') \Rightarrow \mathcal{P}'(\ell+1)$:

Let C be a cycle made of ℓ vertices so that each vertex of C is adjacent to exactly two vertices of C if such cycle exists. We distinguish two cases according to the structures of the interior of C.

C divides the nodes into two connected subsets, we call interior the finite one and denote it by I_C.

The first case is when there is a vertex $v \in I_C$ wich is connected to the rest of I_C by exactly one leftdown, left, rightup or right vertex. We consider the cycle C' wich has $I_C \setminus v$ as interior. C' has less than ℓ vertices so we can apply the induction. According to the position of v the result follow, as shown on Figure 5. In this figure, the vertices of C are full cycles, the leftdown and rightdown neighbours are the empty cycles. Given a pair of grids, on the left is represented the cycle C, on the right the reduced cycle. The vertices of the cycles and the neighbours that change are in purple, the other in red. Notice that there is as many extra full purple cycles as there is extra empty purple cycle in the left grid than in the right one. This number is exactly the number of full line empty cycles.

If there is no such v, we consider a subset of vertices $V \subset I_C$ with no upright and upleft neighbours such that they form a maximal right-left line. We let C' be the cycle with interior $I_C \setminus V$. It satisfies that each vertex of C' is adjacent to exactly two vertices of C' and C' has less than $\ell - 1$ vertices so we can apply the induction. All possible situations are illustrated on Figure 6.

□

Remark 17 Note that the above proofs work also for hexagonal graphs containing a sufficiently large hexagonal graphs. For the first proof, for $k \geq 3$ it suffices that the graph contains a vertex of degree 6.

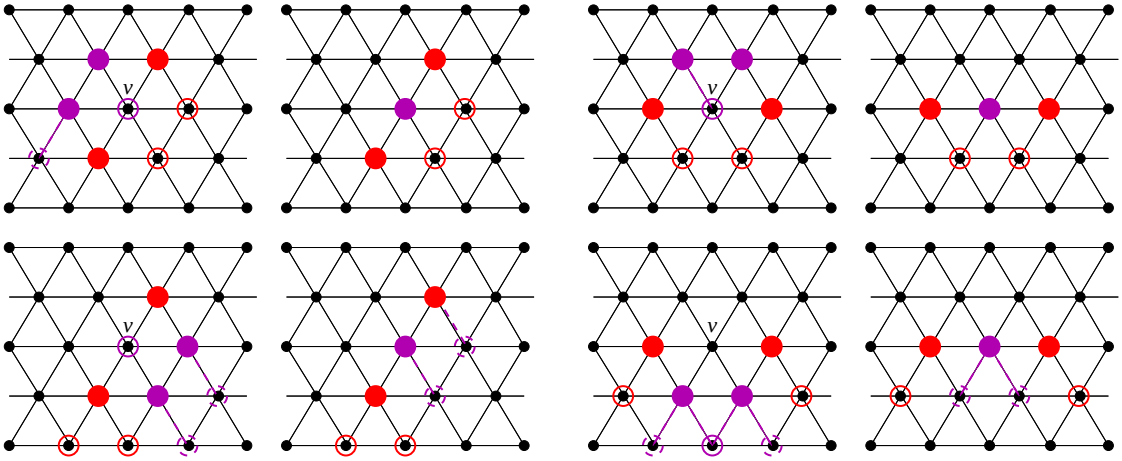


Figure 8: The 4 possible positions for v.

For $k = 1, 2$, it must contain all the vertices at distance at most $\frac{kq}{2}$ of a given node u : starting from u and decreasing f at most $\frac{kq}{2}$ times, yields the desired contradiction. For the proof using Lemma 16, it also works, but one needs a bigger subgrid.

Corollary 18 *For $1 \leq k \leq 5$, there are α_k-approximate algorithms for finding a k-improper colouring of a weighted hexagonal graph, where $\alpha_1 = \frac{20}{11}$, $\alpha_2 = \frac{12}{7}$, $\alpha_3 = \frac{18}{13}$, $\alpha_4 = \frac{80}{63}$, and $\alpha_5 = \frac{41}{36}$.*

Proof. Theorems 9 and 15 give the result for $k = 1$. Theorems 8 and 15 give the result for $2 \leq k \leq 4$. Theorems 10 and 15 give the result for $k = 5$. □

5 Conclusion

We have proposed several approximate algorithms whose approximation ratios are summarized in the following table:

	Grid	Hexagonal
$k = 1$	13/9	20/11
$k = 2$	27/20	12/7
$k = 3$	19/16	18/13
$k = 4$	X	80/63
$k = 5$	X	41/36

It remains to answer Problem 3, ie to find wether or not optimally improper colouring a weighted grid graph is NP-complete or not.

References

[1] S. Alouf, E. Altman, J. Galtier, J.-F. Lalande, et C. Touati, Quasi-optimal bandwidth allocation for multi-spot MFTDMA satellites, in *Proc. IEEE INFOCOM* 2005, Miami, FL, 2005.

[2] F. Havet. Channel assignement and multicolouring of the induced subgraphs of the triangular lattice. *Discrete Mathematics*, 233:219–231, 2001.

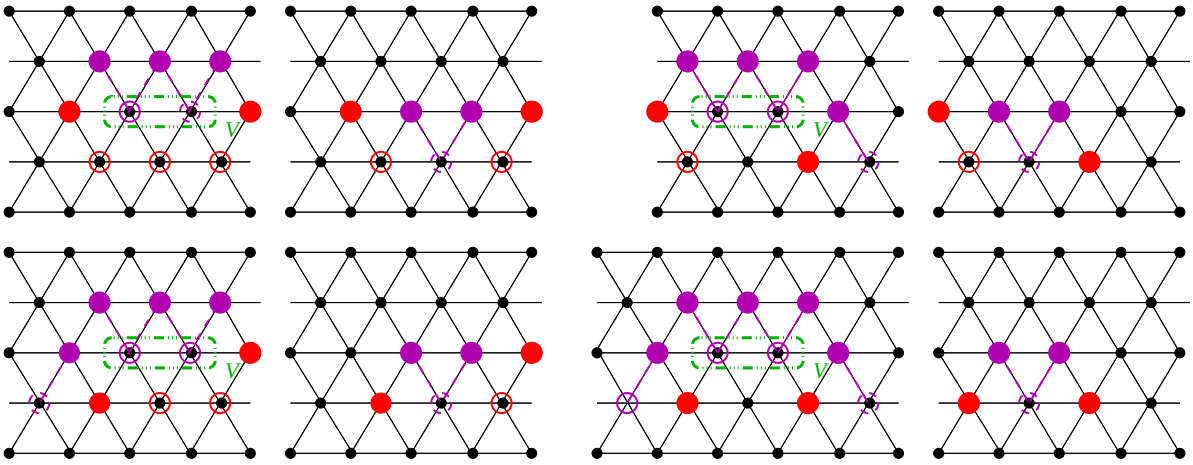


Figure 9: The 4 different positions for V.

[3] F. Havet, R. J. Kang and J.-S. Sereni, Improper colourings of unist disk graphs, *Electronic Notes in Discrete Mathematics* **22** (2005), 123–128.

[4] F. Havet and J. Zerovnik. Finding a five bicolouring of a triangle-free subgraph of the triangular lattice. *Discrete Mathematics*, 244:103–108, 2002.

[5] J. van den Heuvel, R.A. Leese and M.A. Sheperd, Graph Labelling and radio channel assignment, *J. Graph Theory* **29** (1998), no. 4, 263–283.

[6] L. Narayanan and S. Schende, Static Frequency Assignment in Cellular Networks, In *Sirocco 97, (Proceedings of the 4th international Colloqium on structural information and communication complexity, Ascona, Switzerland)*, D. Krizanc and P. Wildmayer (eds.), Carleton Scientific 1997, pp. 215–227.

[7] C. McDiarmid and B. Reed, Channel assignment and weighted coloring, *Networks* **36** (2000), 114–117.

[8] J.-S. Sereni, *Coloration de graphes et applications*, PhD thesis, University of Nice-Sophia-Antipolis, July 2006

[9] K. S. Sudeep and S. Vishwanathan, A technique for multicoloring triangle-free hexagonal graphs. *Discrete Math.* **300** (2005), no. 1-3, 256–259.

The Proportional Colouring Problem: Optimising Buffers in Radio Mesh Networks [1]

Florian Huc [2]

Project team Mascotte - I3S (CNRS-UNS)/INRIA Sophia Antipolis, France

Cláudia Linhares-Sales [3]

Universidade Federal do Ceará, Brazil

Hervé Rivano

Project team Mascotte - I3S (CNRS-UNS)/INRIA Sophia Antipolis, France

Abstract

In this paper, we consider a new edge colouring problem modelling call scheduling optimisation issues in wireless mesh networks: the proportional edge-colouring. It consists in finding a minimum cost edge-colouring of a graph which preserves the proportion given by the weights associated to each of its edges. We show that deciding if a weighted graph admits a proportional colouring is polynomial while determining its *proportional chromatic index* is NP-hard. We then give lower and upper bounds for this parameter that can be computed in polynomial time. We finally identify a class of graphs and a class of weighted graphs for which the proportional chromatic index can be determined exactly.

Keywords: edge colouring, proportional colouring, mesh networks, call scheduling.

1 Introduction

Wireless mesh networks (WMNs) are cost-effective solutions for ubiquitous high-speed services [1]. They are self-organised networks with a fixed infrastructure of wireless mesh routers interconnected to provide Internet access to mobile network users. This infrastructure, forming a wireless backhaul network, is connected to Internet by special routers called mesh gateways as depicted in Figure 1.

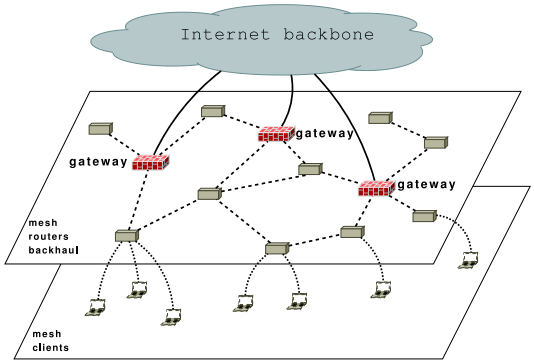


Fig. 1. A WMN topology: mobile clients access Internet through a multi-hop wireless backhaul network of routers and gateways.

The fixed infrastructure of the WMN is often modelled by a directed graph $G = (V, E)$ of N nodes representing mesh points. Each device has a single interface allowing to send or receive packets on the only channel available. This channel is therefore shared between all the nodes [2]. In the following, the network is assumed to be synchronous. We focus on the steady state of the network, which is therefore periodic with a period duration T. The communication protocol is a slotted TDMA multiplexing. Computing an optimal allocation of the slots to the links taking radio interference into account refers to the throughput provided to each router. This throughput corresponds to the ratio of the volume of data sent during a period over T, the duration of the period [3].

In the following, we assume that these ratio are given in order to achieve the

[1] This work has been partially funded by European project IST/FET AEOLUS and French project ANR-JC OSERA and ARC CARMA.
[2] Contact author. Email: `Florian.Huc@sophia.inria.fr`. F. Huc is supported by CNRS and Région PACA.
[3] This work has been done while C. Linhares-Sales was visiting the Mascotte project-team with support by CNPq/Brazil.

throughput requirements defined by a set of Constant Bit Rate requests. The main optimisation issue is to find an achievable scheduling of the links, that is an assignment of time slots to the links during which they are activated in order to realise the needed throughput. Minimising the total number of slots is related to optimising the cost of the wireless routers as explained in Section 3.

Given a weighted graph (G, w) where w is a non-negative weight function over $E(G)$ to, several distinct colouring problems of G have been defined. In [5], one wants to colour the vertices of G while minimising the sum of the weights of the edges whose extremities receive the same colour. On the other hand, in [8], one wants to colour the vertices of G so that for each edge uv, $|c(v) - c(u)|$ is at least the weight of uv, where $c(u)$ and $c(v)$ are the colours assigned to u and v. There exists still the case when all the weights are integers and one must assign to each edge e exactly $w(e)$ colours in such way that the colouring is proper and we use the minimum number of colours (observe that this is exactly the classical edge colouring problem applied to a multigraph obtained from the original graph by replacing each edge by $w(e)$ edges).

In this paper, we consider a new edge colouring problem: the proportional edge-colouring. Given a weighted graph (G, w), we want to find a colouring which preserves the proportion given by the weights associated to each edge. If such colouring exists, we want to find one using a minimum number of colours. We proved that deciding if a weighted graph admits a proportional colouring is polynomial while determining its proportional chromatic index is NP-hard. We then give a lower bound and an upper bound for this parameter that can be computed in polynomial time. Finally, we exhibit a class of graphs and a class of weighted graphs for which we can exactly determine the proportional chromatic index.

2 Preliminaries

Throughout the paper, (G, w) denotes a weighted simple graph where w is a positive function called *weight function* defined on the edges of G, $w : E(G) \rightarrow [0, 1]$. We denote by Δ the maximum degree of G. A proper edge colouring of G is an assignment of colours to the edges of G such that no adjacent edges have the same colour.

The classical edge colouring problem is to determine the chromatic index of a simple graph G, $\chi'(G)$, that is the minimum integer k such that G admits a proper edge colouring using k colours. In 1964, Vizing proved that $\chi'(G)$ is at most $\Delta + 1$ [10]. Since it is at least Δ, we can classify every graph: a

graph is in *Class 1* if its chromatic number is Δ and in *Class 2* otherwise. Surprisingly, deciding if a graph is Class 1 or 2 is hard [6], even for cubic graphs [7].

The upper bound presented in this work is closely related to the fractional edge colouring index, whose definition follows:

Definition 2.1 • For any natural number $k \geq 1$, let G^k be the graph obtained from G by replacing each edge by k parallel edges. Then

$$\chi'^*(G) = \min_{k \geq 1} \frac{\chi'(G^k)}{k}.$$

- A fractional edge colouring is an assignment of a non-negative weight w_M to each matching M of G so that for each edge we have: $\sum_{M:e \in M} w_M \geq 1$. $\chi'^*(G)$ is the minimum $\sum_M w_M$ over all fractional edge colourings of G .

Recall that the fractional edge colouring index can be determined in polynomial time [9].

3 Proportional colouring

The proportional colouring problem is motivated by the following telecommunication problem. We consider a slotted time division multiplexing radio mesh network connecting routers through directional antennas. We denote by *call* two antennas communicating together. We suppose that a call is achievable in both direction. The network topology defines a graph $G = (V, E)$ whose vertices are the routers and edges are the achievable calls. For the sake of radio interferences and near-far effect, each router can be involved in at most one call at a time. Therefore a set of simultaneously achievable calls is a matching of the graph. The classical *timeslot assignment* problem for a given set of calls consists in decomposing the set into a minimum number of subsets of simultaneously achievable calls. It is equivalent to the proper edge colouring problem [4]. Remark that without the assumption of directional antennas, a set of simultaneously achievable calls would be an induced matching.

The proportional colouring problem arises when we consider *Constant Bit Rates* (CBR) requests. In these settings, we are given a set of communication requests, each request being a source-destination path in the network, and a bit rate. Sending this amount of data on the paths induces that each call has to be periodically activated in a given proportion of the time. This is modelled by a weight function $w : E(G) \to [0, 1]$ obtained for each link e by summing,

over all connexions whose path uses edge e, the bit rates and divided by the capacity of the link.

The problem is now, if possible, to find a *periodical schedule* of the calls satisfying the CBR requests, that is such that a number of timeslots proportional to its weight is given to each call. The periodical schedule is composed of a *base period* which repeats indefinitely. By *base period*, we mean a timeslot assignment scheme that satisfies the following constraint: all packets present at the beginning of the period make exactly one hop during the base period. Hence in the periodical schedule an antenna sends the packet it received and the one created at the previous period while it stocks the packets it receives during this period. Consequently, a packet will take as many base periods to reach its destination as there are hops in its route. This condition is natural since it allows us not to worry on the order of the packet transmission during a period. Since we consider CBR requests, the number of packets to stock in the router queues at each node is proportional to the length of the base period. The size of these queues induces a cost in terms of memory to install in the routers as well as QoS parameters, like end-to-end delay. Decreasing the cost of the network and improving the QoS, is hence dependent on minimising the length of the longest queue i.e. the length of the base period. To illustrate this we give an example in Figure 2.

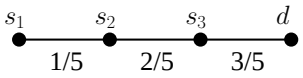


Fig. 2. An example of constant requests on a network composed of 4 antennas

Figure 2 represents four antennas positioned on a line. Antennas s_1, s_2 and s_3 send one packet at destination of d every five units of time. The routing we want to use is the shortest path (and unique path in our example) routing. It implies that a call s_1s_2 is made one fifth of the time, a call s_2s_3 two fifth of the time and a call s_3d three fifth of the time. For this set of CBR requests and this routing, we can use the following base period: node s_1 and s_3 send the packet generated during previous period (one slot), then node s_2 sends the packet generated during the previous period plus the one received (two slots). Then node s_3 sends the two packets received during the previous period (two slots). This base period of length five is optimal in the sense that no base period can length less. In this case, we need two buffers, one of size one at s_2 and one of size two at s_3.

Our problem is therefore to find a proper edge colouring of G that preserves the proportions given by the weights of the edges, with a minimum number of

colours. This is *the proportional colouring problem* formally defined as follows and illustrated in Figure 3.

Definition 3.1 [proportional colouring] Given a weighted graph (G, w), a *proportional colouring* of (G, w) is a function $\mathcal{C} : E \to \mathcal{P}(\{1, \ldots, c\})$ such that for all $e \in E$, we have: $|\mathcal{C}(e)| \geq cw(e)$ and for all $e, f \in E^2$, $e \cap f \neq \emptyset \Rightarrow \mathcal{C}(e) \cap \mathcal{C}(f) = \emptyset$. We call the proportional chromatic index of G, $\chi'_\pi(G, w)$, the minimum number of colours for which a proportional colouring of G exists. If it does not exist then $\chi'_\pi(G, w) = \infty$.

To link the proportional chromatic index to the fractional chromatic index, we will use a multiple graph constructed from the weighted graph (G, w) and an integer m, the m-graph, whose definition follows.

Definition 3.2 [m-graph] Given a weighted graph (G, w) and an integer m, the m-graph G_m is constructed on vertex set $V(G)$ as follows: given an edge $e = (uv) \in E(G)$, we put $\lceil mw(e) \rceil$ multiple edges uv in $E(G_m)$.

Definition 3.3 [mcd(w)] Given a weighted graph (G, w) with w taking value in $\mathbb{Q}$, we set $mcd(w)$ as the minimum common denominator of all the values taken by w.

Remark 3.4 Given a weighted graph (G, w) and an integer m, if it exists a colouring using km colours (where k is a constant) and giving the same number of colours to all the edges of G_m, then this colouring can be easily transformed into a proportional colouring of (G, w).

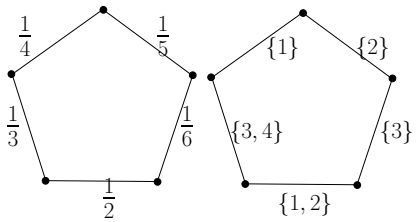


Fig. 3. A weighted C_5 and a proportional colouring using 4 colours

Figure 3 is an example of a weighted graph and its proportional colouring. The proportional colouring problem is divided into two subproblems: the first one consists in proving that there exists an integer c such that a c-proportional colouring of (G, w) exists. The second one is to determine the proportional chromatic index of (G, w). The object of next subsection is to characterise the complexity of each subproblem.

3.1 Complexity results

We start by giving simple facts that enable us to say if a weighted graph has a finite or an infinite proportional chromatic index and eventually to bound it. We conclude this section with Theorem 3.6 stating the complexity results.

Fact 3.5 *Let (G, w) be a weighted graph.*

- *If there is a vertex u of G with $\sum_{uv \in E(G)} w(uv) > 1$ then $\chi'_\pi(G, w) = \infty$.*
- *If for all $uv \in E(G)$, $w(uv) \leq 1/(\Delta + 1)$ then $\chi'_\pi(G, w) \leq \Delta + 1$.*
- *Similarly, if for all $uv \in E(G)$, $w(uv) \leq 1/\chi'^*(G_{mcd})$ then $\chi'_\pi(G, w) \leq q$ where q is the numerator of $\chi'^*(G_{mcd})$.*

Theorem 3.6 *Let (G, w) be a weighted graph with w taking value in $\mathbb{Q}$.*

i) Determining if there exists a proportional colouring for (G, w) is polynomial.

ii) Determining the proportional chromatic index of (G, w) is NP-hard.

Proof.

i) Let (G, w) be any weighted graph with $w : E(G) \to [0, 1] \cap \mathbb{Q}$. Let $mcd = mcd(w)$. In the mcd-graph, we have $\chi'^*(G_{mcd}) \leq mcd \Leftrightarrow \chi'_\pi(G, w) < \infty$.

Indeed suppose $\chi'_\pi(G, w) = k < \infty$, then there exist a colouring using k colours such that each edge receive at least the correct proportion of colours. If we repeat this colouring mcd times, we still have a proportional colouring even if it is not minimal. This colouring gives a colouring of G_{mcd} using $k.mcd$ colours in which each edges receive at least k colours then $\chi'^*(G_{mcd}) \leq \frac{k.mcd}{k}$.

On the other hand, suppose that $\chi'^*(G_{mcd}) = \frac{q}{p}$ and consider an optimal fractional edge colouring of G_{mcd} using $q.c$ colours for some c (we can choose it to be minimal), this colouring gives at least $p.c$ colours to each edge. Furthermore this colouring can be extended to an edge colouring of G that uses $q.c$ colours and gives $p.c.mcd.w(e)$ to each edge e. This colouring is proportional since, by assumption, $mcd.w(e).p.c \geq q.c.w(e)$, and hence $\chi'_\pi(G, w) = k \leq q.c < \infty$.

ii) Let G be any graph of maximum degree Δ. Now, set the weights of all edges to $\frac{1}{\Delta+1}$. Since G is $(\Delta + 1)$-colourable, G is proportionally $(\Delta + 1)$-colourable. But now observe that if (G, w) is admits proper edge colouring with Δ colours then it admits a proportional edge colouring with Δ colours. Since determining if G is Δ or $(\Delta + 1)$-colourable is NP-hard [6], it is NP-hard to determine the proportional chromatic index of an instance (G, w).

□

Remark 3.7 [w taking value in $\mathbb{Q}$ is realistic] Notice that the hypothesis "w taking value in $\mathbb{Q}$ "makes sense because of the origin of the problem. Indeed any instance of the original telecommunication problem would induce rational weights over the edges since they represent a ratio of a number of timeslots over the length of the period. As for an example the weights could be computed from an initial time slot assignment that we want to improve, hence yielding rational weights with, more precisely, rather small operands.

3.2 Partial characterisation

In despite of the difficulty of computing the proportional chromatic index of a weighted graph (G, w), we can deduce polynomially computable lower and upper bounds. Clearly, for a weighted graph (G, w), $\chi'_\pi(G, w) \geq \Delta(G)$. We can get a better lower bound by trying to satisfy the proportions at every vertex of G.

Theorem 3.8 (Lower bound) *Let m be the minimum integer satisfying for all $u \in V$:*

$$\sum_{v \ st \ uv \in E} \lceil mw(uv) \rceil \leq m \tag{1}$$

If no m satisfies all the above equations then no proportional colouring exists. Otherwise, if there is a solution m and that (G, w) admits a proportional colouring, $\chi'_\pi(G, w)$ is at least m.

If (G, w) admits no proportional colouring, its proportional chromatic index is infinite, however, if it admits a proportional colouring, we can deduce an upper bound from the fractional colouring of the corresponding *mcd*-graph.

Theorem 3.9 (Upper bound) *Let (G, w) be a weighted graph, if a proportional colouring exists and that w takes value in $\mathbb{Q}$ then a proportional colouring of (G, w) using q colours exists, where q is the minimum number of colours used in a minimal fractional edge colouring of $G_{mcd(w)}$.*

Proof. Theorem 3.9 is a consequence of the proof of Theorem 3.6 i). Indeed to prove that deciding if a proportional colouring of G exist is polynomial, we construct a proportional colouring of G using q colours where q is the upper bound of the Theorem. □

In general, given a weighted graph (G, w), the *mcd* of the values taken by w is not an upper bound. Indeed, consider $(P, \frac{1}{3})$: the Petersen graph P with a weight function uniformly equal to $\frac{1}{3}$ on all edges of the graph, as

depicted in Figure 4. Since $\chi'^*(P) = 3$, we have $\chi'_\pi(P, \mathbf{\frac{1}{3}}) < \infty$. However $\chi'_\pi(P, \mathbf{\frac{1}{3}}) > 3 = mcd(\mathbf{\frac{1}{3}})$ since P is not 3-colourable.

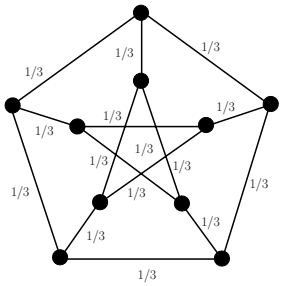


Fig. 4. The Petersen graph with the weight function uniformly equal to $\frac{1}{3}$ on all edges.

The proportional chromatic index is also different from the upper bound given by the fractional chromatic index as an edge can receive proportionally more than what it asks for. This is illustrated by the example of Figure 3 (C_5, w). Indeed in this example, we have $\chi'^*(C_{5mcd(w)}) \geq 50 > 4 = \chi'_\pi(C_5, w)$.

One can also ask for which classes of graphs the proportional colouring problem can be solved in polynomial time. The next theorems announce two positive results, the first one giving a condition in which the lower bound is reached.

Theorem 3.10 *Let (G, w) be a weighted graph. If there is a solution m to the set of Equations (1), then $\chi'_\pi(G, w) = m \Leftrightarrow \chi'(G_m) = m$.*

Proof. We start to prove $\chi'_\pi(G, w) = m \Rightarrow \chi'(G_m) = m$: A proportional colouring using m colours, is a colouring of G which gives $\lceil m.w(e) \rceil$ to each edge e. In G_m each edge e of G is replaced by $\lceil m.w(e) \rceil$ edges, so the previous colouring gives a proper colouring of G_m.

We now prove $\chi'_\pi(G, w) = m \Leftarrow \chi'(G_m) = m$. If G_m has a colouring using m colours, it gives a proportional colouring of (G, w) using m colours. Hence $\chi'_\pi(G, w) \leq m$ and with Theorem 3.8, we can conclude $\chi'_\pi(G, w) = m$. □

This Theorem gives the proportional chromatic index of some simple class of weighted graphs. It is sufficient that, given an m, the corresponding m-graphs have their chromatic index equal to their maximum degree. It includes trees, grids and more generally any bipartite graph as stated by Corollary 3.11.

Corollary 3.11 *Let (G, w) be a weighted bipartite graph. If there is a solution m to the set of Equations (1), then (G, w) accepts a proportional colouring using m colours.*

Proof. Given a weighted bipartite graph (G, w) with a solution m to the Equations (1), the maximun degree of G_m is m. Observe that G_m is also bipartite and hence is edge chromatic number is m: $\chi'(G_m) = m$ and so the previous theorem gives the result. □

If G is an outerplanar (a planar graph which has a planar representation such that all edges are incident to the unbounded face) which is not an odd cycle, Theorem 3.10 also includes the weighted graphs (G, w) if w is a weight function uniform on all edges. Indeed, we know that a simple outerplanar is of class 1 iff it is not an odd cycle. Hence an outerplanar which is not an odd cycle and such that all edges have the same multiplicity has its chromatic index equals to its maximum degree. Given a weighted outerplanar graph (G, w) with w constant, for any m, G_m is an outerplanar graph with all edges having the same multiplicity, its maximum degree is m and hence $\chi'(G_m) = m$. For m solution to the set of Equations (1), Theorem 3.10 applies.

Recall that the condition w uniform over all edges is not sufficient since we proved Theorem 3.6 using a weighted graph which has a constant weight function. Furthermore, if G is an outerplanar but w is not uniform on all edges then we can not conclude using Theorem 3.10. To see this, consider a simple outerplanar graph G with a triangle, a weigth function w and m the solution to the Equations (1). If w is such that the m-graph G_m has too many multiple edges on a triangle it may not have edge chromatic index m. Figure 5 presents an outerplanar G and an outerplanar with multiple edges which may be an m-graph of G. In this example we have a triangle with edge multiplicity 4 which forces the chromatic index to be at least 12 while the maximun degree of the multiple graph is 10.

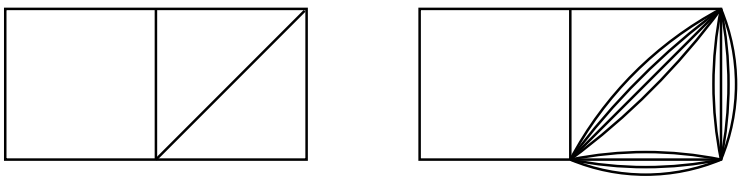


Fig. 5. Example of an outerplanar G with multiple edges and $\chi'(G) \neq \Delta(G)$.

We can also give more information on the proportional chromatic index of a weighted graph if its weight function verifies some properties. Next Theorem goes in this direction:

Theorem 3.12 *Let (G, w) be a weighted graph with w taking values in $\mathbb{Q}$. Let e be an edge with an end vertex v such that $\sum_{uv \in E} w(uv) = 1$. Then,*

the denominator of $w(e)$ divides the number of colours used in a proportional colouring of (G, w), in particular it divides $\chi'_\pi(G, w)$.

Proof. Consider a proportional colouring of (G, w) using m colours. Let e be an edge with an end vertex v such that $\sum_{uv \in E} w(uv) = 1$. We have $\sum_{uv \in E} \lceil m.w(uv) \rceil \leq m = \sum_{uv \in E} m.w(uv)$. Hence $\lceil m.w(e) \rceil = m.w(e)$. It is in particular true for $m = \chi'_\pi(G, w)$. □

Remark 3.13 [why w should takes value in $\mathbb{Q}$] As Remark 3.7 says, the origin of the problem justifies the hypothesis that w takes value in $\mathbb{Q}$. In the context of this Theorem, there is a second reason. Indeed, if at a vertex v where the weights sum at 1, $\sum_{uv \in E} w(uv) = 1$, we have an irrational weight then the set of Equations (1) do not have a solution, which means that no proportional colouring exist.

From Theorem 3.12, we can deduce the following corollary:

Corollary 3.14 *Let (G, w) be a weighted graph. If the proportional chromatic number of (G, w) is finite and every edge e has an end vertex v with $\sum_{uv \in E} w(uv) = 1$, then $\chi'_\pi(G, w) = mcd(w)$.*

Notice that we do not need the hypothesis "w taking values in $\mathbb{Q}$ "from the above remark.

Proof. The previous Theorem implies that $mcd(w)$ is a lower bound. Since, by Theorem 3.9 it is also an upper bound, the result follows. □

In a weighted graph (G, w), a vertex v with $\sum_{uv \in E} w(uv) = 1$, represents a *saturated node*, i.e. a node through which no extra request can be routed. A weighted graph (G, w) with w taking values in $\mathbb{Q}$ such that every edge is incident to a saturated vertex represents a *saturated network*, i.e. a network in which no extra request can be routed. The previous corollary gives the value of the proportional chromatic index of such a network.

The presence of a saturated node also allows to have an inapproximability result:

Corollary 3.15 *Given a weighted graph (G, w) with w taking values in $\mathbb{Q}$ and a vertex v such that $\sum_{uv \in E} w(uv) = 1$. Let q be the mcd of $w(uv)$, $uv \in E$. $\chi'_\pi(G, w)$ is not approximable within the constant $q - 1$.*

Proof. Theorem 3.12 implies that q divides the number of colours used in a proportional colouring, hence no proportional colouring exist with $\chi'_\pi(G, w)+i$ colours for $0 < i < q$. □

4 Conclusion

Motivated by the applications modelled by the proportional colouring problem and its hardness, we let the following general open questions: find approximation algorithms for classes of graphs which usually occur in telecommunication, as circular-arc graphs and triangular lattices and determine other classes of graphs (and weighted classes of graphs) for which the proportional chromatic index can be calculated in polynomial time.

References

[1] I. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: a survey," *Computer Networks*, vol. 47, no. 4, pp. 445–487, 2005.

[2] H. Liu and B. Zhao, "Optimal scheduling for link assignment in traffic-sensitive stdma wireless ad-hoc networks," in *ICCNMC*, ser. LNCS, vol. 3619, 2005, pp. 218–228.

[3] R. Klasing, N. Morales, and S. Perennes, "On the complexity of bandwidth allocation in radio networks with steady traffic demands," *Theoretical Computer Science*, 2008, to appear.

[4] J. Bibb Cain, T. Billhartz, L. Foore *A link scheduling and ad hoc networking approach using directional antennas*, Military comm. conf., **1** Oct 2003, 643–648.

[5] Carlson, R.C. and Nemhauser, G.L., *Scheduling to minimize interaction cost*, Oper. Res. **1**4 (1966), 52–58.

[6] Holyer, I. *The NP-completeness of edge colourings*, SIAM J. Computing **1**0 (1981), 718–720.

[7] Koreas, D.P., *The NP-completeness of chromatic index in triangle free graphs with maximum vertex of degree 3*, App. Math. and Comp. **8**3 (1) (1997), 13–17.

[8] McDiarmid, C., *Discrete mathematics and radio channel assignment*, In: Recent advances in algorithms and combinatorics, eds. B. Reed and C. Linhares-Sales, Springer, New York, 2003, 27–63.

[9] Padberg, M. and Rao, M., *Odd minimum cutsets and b-matchings*, Math. Oper. Res. **7** (1982), 67–80.

[10] Vizing, V.G., *On an estimate of a chromatic class of a p-graph* (In Russian), Diskret. Analiz. **3** (1964), 25–30.

Annexe C

Réseaux d'accès et de cœur

C.1 WDM and Directed Star Arboricity

Article soumis à Combinatorics, Probability and Computing.

C.2 Muticast Routing and Wavelength Assignment in Passive Optical Access Networks

Coming soon.

WDM and directed star arboricity

Omid Amini* Frédéric Havet* Florian Huc* Stéphan Thomassé†

Abstract

A digraph is *m-labelled* if every arc is labelled by an integer in $\{1,\dots,m\}$. Motivated by wavelength assignment for multicasts in optical networks, we study *n-fiber colourings* of labelled digraphs. These are colourings of the arcs of D such that at each vertex v, and for each colour λ, $in(v,\lambda)+out(v,\lambda)\leq n$ with $in(v,\lambda)$ the number of arcs coloured λ entering v and $out(v,\lambda)$ the number of labels l such that there is at least one arc of label l leaving v and coloured with λ. The problem is to find the minimum number of colours $\lambda_n(D)$ such that the m-labelled digraph D has an n-fiber colouring. In the particular case, when D is 1-labelled, $\lambda_n(D)$ is called the *directed star arboricity* of D, and is denoted by $dst(D)$. We first show that $dst(D)\leq 2\Delta^-(D)+1$ and conjecture that if $\Delta^-(D)\geq 2$ then $dst(D)\leq 2\Delta^-(D)$. We also prove that if D is subcubic then $dst(D)\leq 3$ and that if $\Delta^+(D),\Delta^-(D)\leq 2$ then $dst(D)\leq 4$. Finally, we study $\lambda_n(m,k)=\max\{\lambda_n(D)\mid D \text{ is } m\text{-labelled and } \Delta^-(D)\leq k\}$. We show that if $m\geq n$ then $\left\lceil\frac{m}{n}\left\lceil\frac{k}{n}\right\rceil+\frac{k}{n}\right\rceil\leq\lambda_n(m,k)\leq\left\lceil\frac{m}{n}\left\lceil\frac{k}{n}\right\rceil+\frac{k}{n}\right\rceil+C\frac{m^2\log k}{n}$ for some constant C. We conjecture that the lower bound should be the right value of $\lambda_n(m,k)$.

1 Introduction

The origin of this paper is the study of wavelength assignment for multicasts in star network. Partial results are already obtained by Brandt and Gonzalez [4]. We are given a star network in which a central node is connected by optical fibers to a set of nodes V. The nodes of V communicates together using WDM (*wavelength-division multiplexing*), hence different signals may be sent at the same time through the same fiber but on different wavelengths. The central node is an all-optical transmitter which can redirect a signal arriving from a node on a particular wavelength to some (one or more) of the others nodes on the same wavelength. It means that the central node is able to duplicate a message incoming on a wavelength to different fibers without changing its wavelength. Therefore if a node v send a multicast to a set of nodes $S(v)$, v should send the message to the central node on a set of wavelengths so that the central node redirect it to each node of $S(v)$ using one of these wavelengths. The aim is to minimize the total number of used wavelengths.

We first study the very fundamental case when the fiber is unique and each vertex v sends a unique multicast $M(v)$ to a set $S(v)$ of nodes. Let D be the digraph with vertex set V such that the outneighbourhood of a vertex v is $S(v)$. Note that this is a digraph and not a multidigraph (there is no multiple arcs) as $S(v)$ is a set. Then the problem is to find the smallest k such that there exists a mapping $\phi:V(D)\to\{1,\dots,k\}$ satisfying the two conditions:

(i) $\phi(uv)\neq\phi(vw)$;

(ii) $\phi(uv)\neq\phi(u'v)$.

*Projet Mascotte, CNRS/INRIA/UNSA, INRIA Sophia-Antipolis, 2004 route des Lucioles BP 93, 06902 Sophia-Antipolis Cedex, France. oamini,fhavet,fhuc@sophia.inria.fr. These authors are partially supported by the European project AEOLUS

†LIRMM, 161 rue ADA, Montpellier, Francethomasse@lirmm.fr

Such a mapping is called *directed star k-colouring*. The *directed star arboricity* of a digraph D, denoted by $dst(D)$, is the minimum integer k such that there exists a directed star k-colouring. This notion has been introduced in [6] and is an analog of the *star arboricity* defined in [1]. An *arborescence* is a connected digraph in which every vertex has indegree 1 except one, called *root*, which has indegree 0. A *forest* is the disjoint union of arborescences. A *star* is an arborescence in which the root dominates all the other vertices. A *galaxy* is a forest of stars. Clearly, every colour class of a directed star colouring is a galaxy. Hence, the directed star arboricity of a digraph D is the minimum number of galaxies into which $A(D)$ may be partitioned.

For a vertex v, its indegree $d^-(v)$ corresponds to the number of multicasts it receives. A sensible assumption is that a node receives a bounded number of multicasts. Hence, Brandt and Gonzalez [4] studied the directed star arboricity of a digraph D with regards to its maximum indegree. The *maximum indegree* of a digraph D, denoted $\Delta^-(D)$ or simply Δ^-, is $\max\{d^-(v) \mid v \in V(D)\}$. Brandt and Gonzalez showed that $dst(D) \leq \lceil 5\Delta^-/2 \rceil$. This upper bound is tight if $\Delta^- = 1$ because odd circuits have directed star arboricity 3. However it can be improved for larger value of Δ^-. We conjecture that if $\Delta^- \geq 2$, then $dst(D) \leq 2\Delta^-$.

Conjecture 1 Every digraph D with maximum indegree $\Delta^- \geq 2$ satisfies $dst(D) \leq 2\Delta^-$.

This conjecture would be tight as Brandt [3] showed that for every Δ^-, there is an acyclic digraph D_{Δ^-} with maximum indegree Δ^- and $dst(D_{\Delta^-}) = 2\Delta^-$. Note that to prove this conjecture, it is sufficient to prove it for $\Delta^- = 2$ and $\Delta^- = 3$. Indeed a digraph with maximum indegree $\Delta^- \geq 2$ has an arc-partition into $\Delta^-/2$ digraphs with maximum indegree 2 if Δ^- is even, and into $(\Delta^- - 1)/2$ digraphs with maximum indegree 2 and one with maximum indegree 3 if Δ^- is odd. In section 2, we show that $dst(D) \leq 2\Delta^- + 1$ and settle Conjecture 1 for acyclic digraphs.

Remark 2 Note that we restrict ourselves to simple digraphs, i.e. circuits of length two are permitted, but not multiple arcs. When multiple arcs are allowed, all the bounds above do not hold. Indeed, given an integer Δ^-, the multidigraph T_{Δ^-} with three vertices u, v and w and Δ^- parallel arcs uv, vw and wu satisfies $dst(T_{\Delta^-}) = 3\Delta^-$. Moreover, this example is extremal since every multidigraph satisfies $dst(D) \leq 3\Delta^-$. Indeed let us show it by induction: pick a vertex v with outdegree at most Δ^- in a terminal strong component. A strong component C of a digraph is *terminal* if there is no arc leaving C, i.e. with tail in C and head outside of C. If v has no inneighbour, it is isolated and we remove it. Otherwise, we consider any arc uv. Its colour must be different from the colours of the $d^-(u)$ arcs entering u, the $d^+(v)$ arcs leaving v and the $d^-(v) - 1$ other arcs entering v, so at most $3\Delta^- - 1$ arcs in total. Hence, remove the arc uv, apply induction, and extend the colouring to uv. Therefore, for multidigraphs, the bound $dst(D) \leq 3\Delta^-$ is sharp.

We then study the directed star arboricity of a digraph with bounded maximum degree. The *degree* of a vertex v is $d(v) = d^-(v) + d^+(v)$. It corresponds to the degree of the vertex in the underlying multigraph. (We have edges with multiplicity 2 in the underlying multigraph each time there is a circuit of length two in the digraph.) The *maximum degree* of a digraph D, denoted $\Delta(D)$, or simply Δ when D is clearly understood from the context, is $\max\{d(v), v \in V(D)\}$. Let us denote by $\mu(G)$, the maximum multiplicity of an edge in a multigraph. By Vizing's theorem, one can colour the edges of a multigraph with $\Delta(G)+\mu(G)$ colours so that two edges have different colours if they are incident. Since the multigraph underlying a digraph has maximum multiplicity at most two, for any digraph D, $dst(D) \leq \Delta + 2$. We conjecture the following:

Conjecture 3 *Let D be a digraph with maximum degree $\Delta \geq 3$. Then $dst(D) \leq \Delta$.*

This conjecture would be tight since every digraph with $\Delta = \Delta^-$ has directed star arboricity at least Δ. In section 3, we prove that Conjecture 3 holds when $\Delta = 3$.

Pinlou and Sopena [9] studied a stronger form of directed star arboricity, called *acircuitic directed star arboricity*. They add the extra condition that any circuit has to have at least three distinct colours. Note that such a notion applies only to *oriented graphs* that are digraphs without circuit of length 2. Indeed such a circuit may not receive 3 colours. They showed that the acircuitic directed star arboricity of a *subcubic* (i.e. each vertex has degree at most 3) oriented graph is at most four. We give a new and very short proof of this result.

A first step towards Conjectures 1 and 3 would be to prove the following statement which is weaker than these two conjectures.

Conjecture 4 *Let $k \geq 2$ and D be a digraph. If $\max(\Delta^-, \Delta^+) \leq k$ then $dst(D) \leq 2k$.*

This conjecture holds and is far from being tight for large values of k. Indeed Guiduli [6] showed that if $\max(\Delta^-, \Delta^+) \leq k$ then $dst(D) \leq k + 20 \log k + 84$. Guiduli's proof is based on the fact that, when both out and indegree are bounded, the colour of an arc depends on the colour of few other arcs. This bounded dependency allows the use of the Lovász Local Lemma. This idea was first used by Algor and Alon [1], for the star arboricity of undirected graphs. Note also that Guiduli's result is (almost) tight since there are digraphs D with $\max(\Delta^-, \Delta^+) \leq k$ and $dst(D) \geq k + \Omega(\log k)$ (see [6]). Furthermore, as for Conjecture 1, it is sufficient to prove Conjecture 4 for $k = 2$ and k=3. In Section 4, we prove that Conjecture 4 holds for $k = 2$. By the above remark, it implies that Conjecture 4 holds for all even k.

In Section 5, we investigate the complexity of finding the directed star arboricity of a digraph. Unsurprisingly, this is an $\mathcal{NP}$-hard problem. More precisely, we show that determining the directed star arboricity of a digraph with out- and indegree at most 2 is $\mathcal{NP}$-complete.

Next, we study the more general (and more realistic) problem in which the center is connected to the nodes of V with n optical fibers. Moreover each node may send several multicasts. We note $M_1(v), \ldots, M_{s(v)}(v)$ the $s(v)$ multicasts that node v send. For $1 \leq i \leq s(v)$, the set of nodes to which the multicast $M_i(v)$ is sent is denoted by $S_i(v)$. The problem is still to find the minimun number of wavelength used considering that all fibers are identical.

We model it as a *labelled* digraph problem: We consider a digraph D on vertex set V. For each multicast $M_i(v) = (v, S_i(v))$, $v \in V$, $1 \leq i \leq s(v)$, we add the set of arcs $A_i(v) = \{vw, w \in S_i(v)\}$ with label i. The label of an arc a is denoted by $l(a)$. Thus for every couple (u, v) of vertices and label i there is at most one arc uv labelled by i. If each vertex sends at most m multicasts, there are at most m labels on the arcs. Such a digraph is said to be *m-labelled*. One wants to find an *n-fiber wavelength assignment* of D, that is a mapping $\Phi : A(D) \to \Lambda \times \{1, \ldots, n\} \times \{1, \ldots n\}$ in which every arc uv is associated a triple $(\lambda(uv), f^+(uv), f^-(uv))$ such that :

(i) $(\lambda(uv), f^-(uv)) \neq (\lambda(vw), f^+(vw))$;

(ii) $(\lambda(uv), f^-(uv)) \neq (\lambda(u'v), f^-(u'w))$;

(iii) if $l(vw) \neq l(vw')$ then $(\lambda(vw), f^+(vw)) \neq (\lambda(vw'), f^+(vw'))$.

$\lambda(uv)$ corresponds to the wavelength of uv, and $f^+(uv)$ and $f^-(uv)$ the fiber used in u and v respectively. Hence we can describe the equations as follow:

(i) corresponds to the fact that an arc entering v and an arc leaving v have either different wavelength or different fibers;

(ii) corresponds to the fact that two arcs entering v have either different wavelength or different fibers;

(iii) corresponds to the fact that two arcs leaving v with different labels have either different wavelengths or different fibers.

The problem is to find the minimum cardinality $\lambda_n(D)$ of Λ such that there exists an *n-fiber wavelength assignment* of D.

The crucial thing in an n-fiber wavelength assignment is the function λ which assigns colours (wavelengths) to the arcs. It must be an *n-fiber colouring*, that is a function $\phi : A(D) \to \Lambda$, such that at each vertex v, for each colour $\lambda \in \Lambda$, $in(v,\lambda) + out(v,\lambda) \leq n$ with $in(v,\lambda)$ the number of arcs coloured λ entering v and $out(v,\lambda)$ the number of labels l such that there exists an arc leaving v coloured λ. Once we have an n-fiber colouring, one can easily find a suitable wavelength assignment. For every vertex v and every colour λ, this is done by assigning a different fiber to each arc of colour λ entering v and to each set of arcs of colour λ leaving v and of the same label. Hence $\lambda_n(D)$ is the minimum number of colours such that there exists an n-fiber colouring.

We are particularly interested in $\lambda_n(m,k) = \max\{\lambda_n(D) \mid D \text{ is } m\text{-labelled and } \Delta^-(D) \leq k\}$ that is the maximum number of wavelengths that may be necessary if there are n-fibers and each node sends at most m and receives at most k multicasts. In particular, $\lambda_1(1,k) = \max\{dst(D) \mid \Delta^-(D) \leq k\}$. So our above mentioned results show that $2k \leq \lambda_1(1,k) \leq 2k+1$. Brandt and Gonzalez showed that for $n \geq 2$ we have $\lambda_n(1,k) \leq \left\lceil \frac{k}{n-1} \right\rceil$. In Section 6, we study the case when $n \geq 2$ and $m \geq 2$. We show that if $m \geq n$ then

$$\left\lceil \frac{m}{n} \left\lceil \frac{k}{n} \right\rceil + \frac{k}{n} \right\rceil \leq \lambda_n(m,k) \leq \left\lceil \frac{m}{n} \left\lceil \frac{k}{n} \right\rceil + \frac{k}{n} \right\rceil + C\frac{m^2 \log k}{n} \qquad \text{for some constant } C.$$

We conjecture that the lower bound is the right value of $\lambda_n(m,k)$ when $m \geq n$. We also show that if $m < n$ then

$$\left\lceil \frac{m}{n} \left\lceil \frac{k}{n} \right\rceil + \frac{k}{n} \right\rceil \leq \lambda_n(m,k) \leq \left\lceil \frac{k}{n-m} \right\rceil.$$

The lower bound generalizes Brandt and Gonzalez [4] results which established this inequality in the particular cases when $k \leq 2$, $m \leq 2$ and $k = m$. The digraphs used to show this lower bound are all acyclic. We show that if $m \geq n$ then this lower bound is tight for acyclic digraphs. Moreover the above mentioned digraphs have large outdegree. Generalizing the result of Guiduli [6], we show that for an m-labelled digraph D with both in- and outdegree bounded by k then few colours are needed:

$$\lambda_n(D) \leq \frac{k}{n} + C'\frac{m^2 \log k}{n} \qquad \text{for some constant } C'.$$

2 Directed star arboricity of digraphs with bounded indegree

Our goal in this section is to approach Conjecture 1. It is easy to see that a forest has directed star arboricity 2. Hence, an idea to prove Conjecture 1 would be to show that every digraph has an arc-partition into Δ^- forests. However this statement is false. Indeed A. Frank [5] (see also [10], p.908) characterized digraphs having an arc-partition into k forests. Let $D = (V,A)$. For any $U \subset V$, the digraph induced by the vertices of U is denoted $D[U]$.

Theorem 5 (A. Frank) *A digraph $D = (V,A)$ has an arc-partition into k forests if and only if $\Delta^-(D) \leq k$ and for every $U \subset V$, the digraph $D[U]$ has at most $k(|U|-1)$ arcs.*

Still, Theorem 5 implies that every digraph D has an arc-partition into $\Delta^- + 1$ forests. Indeed for any $U \subset V$, $\Delta^-(D[U]) \leq \min\{\Delta^-, |U|-1\}$, so $D[U]$ has at most $\min\{\Delta^-, |U|-1\} \times |U| \leq (\Delta^-+1)(|U|-1)$ arcs. Hence, every digraph has directed star arboricity at most $2\Delta^- + 2$.

Corollary 6 *Every digraph D satisfies $dst(D) \leq 2\Delta^- + 2$.*

We now lessen this upper bound by one.

Theorem 7 *Every digraph D satisfies $dst(D) \leq 2\Delta^- + 1$.*

The idea to prove Theorem 7 is to show that every digraph has an arc-partition into Δ^- forests and a galaxy G. To do so, we prove a stronger result (Lemma 8) by induction.

A *sink* is a vertex with outdegree 0. A *source* is a vertex with indegree 0. Let D be a multidigraph. It is *k-nice* if $\Delta^- \leq k$ and if the tails of parallel arcs, if any, are sources. A *k-decomposition* of D is an arc-partition into k forests and a galaxy G such that every source of D is isolated in G. Let u be a vertex of D. A k-decomposition of D is *u-suitable* if no arc of G has head u.

Lemma 8 *Let u be a vertex of a k-nice multidigraph D. Then D has a u-suitable k-decomposition.*

Proof. We proceed by induction on $n + k$. We now discuss the connectivity of D:

- If D is not connected, we apply induction on every component.
- If D is strongly connected, every vertex has indegree at least one. Remember also that there is no parallel arcs. Let v be an outneighbour of u. There exists a spanning arborescence T with root v which contains all the arcs with tail v. Let D' be the digraph obtained from D by removing the arcs of T and v. Observe that D' is $(k-1)$-nice. By induction, it has a u-suitable $(k-1)$-decomposition $(F_1, \ldots, F_{k-1}, G)$. Note that the F_i, $1 \leq i \leq k-1$, T and G contain all the arcs of D except those with head v. By construction, $G' = G \cup uv$ is a galaxy since no arc of G has head u. Let $u_1, \ldots, u_{l-1}$ be the inneighbours of v distinct from u, where $l \leq k$. Let $F'_i = F_i \cup u_i v$, for all $1 \leq i \leq l-1$. Then each F'_i is a forest, so $(F_1, \ldots, F_{k-1}, T, G')$ is a u-suitable k-decomposition of D.
- If D is connected but not strongly connected, we consider a strongly connected terminal component D_1. Set $D_2 = D \setminus D_1$. Let u_1 and u_2 be two vertices of D_1 and D_2, respectively, such that u is one of them.

  If D_2 has a unique vertex v (thus $u_2 = v$), since D is connected and D_1 is strong, there exists a spanning arborescence T of D with root v. Now $D' = D \setminus A(T)$ is a $(k-1)$-nice multidigraph, so by induction it has a u-suitable $(k-1)$-decomposition. Adding T to this decomposition, we obtain a u-suitable k-decomposition.

  If D_2 has more than one vertex, by induction, it admits a u_2-suitable k-decomposition $(F_1^2, \ldots, F_k^2, G^2)$. Moreover the digraph D'_1 obtained by contracting D_2 to a single vertex v is k-nice and so has a u_1-suitable k-decomposition $(F_1^1, \ldots, F_k^1, G^1)$. Moreover, since v is a source, it is isolated in G^1. Hence $G = G^1 \cup G^2$ is a galaxy. We now let F_i be the union of F_i^1 and F_i^2 by replacing the arcs of F_i^1 with tail v by the corresponding arcs in D. Then $(F_1, \ldots, F_k, G)$ is a k-decomposition of D which is suitable for both u_1 and u_2.

□

2.1 Acyclic digraphs

It is not hard to show that $dst(D) \leq 2\Delta^-$ when D is acyclic, but we will prove this result in a more constrained way. A *cyclic n-interval* of $\{1, 2, \ldots, p\}$ is a set of n consecutive numbers modulo p. Now for the directed star colouring, we will insist that for every vertex v, the (distinct) colours used to colour the arcs with head v are chosen in a *cyclic k-interval* of $\{1, 2, \ldots, 2k\}$. Thus, the number of possible sets of colours used to colour the entering arcs of a vertex drastically falls from $\binom{2k}{k}$ when every set is *a priori* possible, to $2k$. Note that having consecutive colours on the arcs entering a vertex corresponds to having consecutive wavelengths on the link between the corresponding node and the central one. This is very important for grooming issues. For more details about grooming, we refer to the two comprehensive surveys [7, 8].

Theorem 9 *Let D be an acyclic digraph with maximum indegree k. D admits a directed star $2k$-colouring such that for every vertex, the colours assigned to its entering arcs are included in a cyclic k-interval of $\{1, 2, \ldots, 2k\}$.*

To prove this result, we need the following result on set of distinct representatives.

Lemma 10 *Let $I_1, \ldots, I_k$ be k non necessary distinct cyclic k-intervals of $\{1, 2, \ldots, 2k\}$. Then $I_1, \ldots, I_k$ admit a set of distinct representatives forming a cyclic k-interval.*

Proof. We consider $I_1, \ldots, I_k$ as a set of p *distinct* cyclic k-intervals $I_1, \ldots, I_p$ with respective multiplicity $m_1, \ldots, m_p$ such that $\sum_{i=1}^{p} m_i = k$. Such a system will be denoted by $((I_1, m_1), \ldots, (I_p, m_p))$. We shall prove the existence of a cyclic k-interval J, such that J can be partitioned into p subsets J_i, $1 \leq i \leq p$, such that $|J_i| = m_i$ and $J_i \subset I_i$. This proves the lemma (by associating distinct elements of J_i to each copy of I_i).

We proceed by induction on p. The result holds trivially for $p = 1$. We have two cases:

- There exists i and j such that $|I_j \setminus I_i| = |I_i \setminus I_j| \leq \max(m_i, m_j)$.

  Suppose without loss of generality that $i < j$ and $m_i \geq m_j$. We apply the induction hypothesis to $((I_1, m_1), \cdots, (I_i, m_i + m_j), \cdots, (I_{j-1}, m_{j-1}), (I_{j+1}, m_{j+1}), \cdots, (I_p, m_p))$, in order to find a cyclic interval J', such that J' admits a partition into subsets J'_r, such that for any $r \neq i, j$, $J'_r \subset I_r$ is a subset of size m_r, and $J'_i \subset I_i$ is of size $m_i + m_j$. We now partition J'_i into two sets J_i and J_j with respective size m_i and m_j, in such a way that $(I_i \setminus I_j) \cap J'_i \subseteq J_i$. Remark that this is possible exactly because of our assumption $|I_j \setminus I_i| = |I_i \setminus I_j| \leq m_i$. Since $J_i \subset I_i$ and $J_j \subset I_j$, this refined partition of J' is the desired one.

- For any i, j we have $|I_j \setminus I_i| = |I_i \setminus I_j| \geq \max(m_i, m_j) + 1$.

  Each I_i intersects exactly $2m_i - 1$ other cyclic k-intervals on less than m_i elements. Since there are $2k$ cyclic k-intervals in total and $\sum_{i=1}^{p}(2m_i - 1) = 2k - p < 2k$, we conclude the existence of a cyclic k-interval J which intersects each I_i in an interval of size at least m_i.

  Let us prove that one can partition J in the desired way. By Hall's matching theorem, it suffices to prove that for every subset $\mathcal{I}$ of $\{1, \ldots, p\}$, $|\bigcup_{i \in \mathcal{I}} I_i \cap J| \geq \sum_{i \in \mathcal{I}} m_i$.

  Suppose for a contradiction that a subset $\mathcal{I}$ of $\{1, \ldots, p\}$ violates this inequality. Such a subset will be called *contracting*. Without loss of generality, we assume that $\mathcal{I}$ is a contracting set with minimum cardinality and that $\mathcal{I} = \{1, \ldots, q\}$. Remark that by the choice of J, we have $q \geq 2$. The set $K := \bigcup_{i \in \mathcal{I}} I_i \cap J$ consists of one or two intervals of J, each containing one extremity of J. By the minimality of $\mathcal{I}$, K must be a single interval (if not, one would take $\mathcal{I}_1$ (resp. $\mathcal{I}_2$), all the elements of $\mathcal{I}$ which contains the first (resp. the second) extremity of J. Then one of I_1 or I_2 would be contracting). Thus, one of the two extremities of J is in every I_i, $i \in \mathcal{I}$. Without loss of generality, we may assume that $(I_1 \cap J) \subset (I_2 \cap J) \subset \cdots \subset (I_q \cap J)$. Now, for every $2 \leq i \leq q$, $|I_i \setminus I_{i-1}| = |(I_i \cap J) \setminus (I_{i-1} \cap J)| \geq \max(m_i, m_{i-1}) + 1 \geq m_i + 1$. But $|\bigcup_{i \in \mathcal{I}} I_i \cap J| = |(I_1 \cap J)| + \sum_{i=2}^{q} |(I_i \cap J) \setminus (I_{i-1} \cap J)|$. So $|\bigcup_{i \in \mathcal{I}} I_i \cap J| \geq \sum_{i=1}^{q} m_i + q - 1$, which is a contradiction.

□

Proof of Theorem 9. By induction on the number of vertices, the result being trivial if D has one vertex. Suppose now that D has at least two vertices. Then D has a sink x. By the induction hypothesis, $D \setminus x$ has a directed star $2k$-colouring c such that for every vertex, the colours assigned to its entering arcs are included in a cyclic k-interval. Let $v_1, v_2, \ldots, v_l$ be the inneighbours of x in D, where $l \leq k$ because $\Delta^-(D) \leq k$. For each $1 \leq i \leq l$, let I'_i be a cyclic k-interval which contains all the colours of the

arcs with head v_i. We set $I_i = \{1, \ldots, 2k\} \setminus I'_i$. Clearly, I_i is a cyclic k-interval and the arc $v_i x$ can be coloured by any element of I_i. By Lemma 10, $I_1, \ldots, I_l$ have a set of distinct representatives included in a cyclic n-interval J. Hence assigning J to x, and colouring the arc $v_i x$ by the representative of I_i gives a directed star $2k$-colouring of D. □

Theorem 9 is tight : Brandt [3] showed that for every k, there is an acyclic digraph such that $\Delta^-(D_k) = k$ and $dst(D_k) = 2k$. His construction is the special case for $n = m = 1$ of the construction given in Proposition 21.

3 Subcubic digraphs

Recall that a *subcubic* digraph is a graph with degree at most three. In this section, we first show that the directed star arboricity of a subcubic digraph is at most 3, so proving Conjecture 3 when $\Delta = 3$. We then give a very short proof of a result of Pinlou and Sopena asserting that the acircuitic directed star arboricity of a subcubic digraph is at most 4.

3.1 Directed star arboricity of subcubic digraphs

The aim of this subsection is to prove the following theorem:

Theorem 11 *Every subcubic digraph has directed star arboricity at most* 3.

To do so, we need to establish some lemmas to enable us to extend some partial directed star colouring into directed star colouring of the whole digraph. These lemmas need the following definition. Let $D = (V, A)$ be a digraph and S a subset of $V \cup A$. Suppose that each element x of S is assigned a list $L(x)$. A colouring c of S is an *L-colouring* if $c(x) \in L(x)$ for every $x \in S$.

Lemma 12 *Let C be a circuit in which every vertex v receives a list $L(v)$ of two colours among $\{1, 2, 3\}$ and each arc a receives the list $L(a) = \{1, 2, 3\}$. Then there is no L-colouring c of the arcs and vertices such that $c(x) \neq c(xy)$, $c(y) \neq c(xy)$, and $c(xy) \neq c(yz)$, for all arcs xy and yz if and only if C is odd and all the vertices have the same list.*

Proof. Assume first that every vertex is assigned the same list, say $\{1, 2\}$. If C is odd, it is simple matter to see that we can not find the desired colouring. Indeed, among two consecutive arcs, one has to be coloured 3. If C is even, we colour the vertices by 1 and the arcs alternately by 2 and 3.

Now assume that $C = x_1 x_2 \ldots x_k x_1$ and x_1 and x_2 are assigned different lists. Say $L(x_1) = \{1, 2\}$ and $L(x_2) = \{2, 3\}$. We colour the arc $x_1 x_2$ by 3, the vertex x_2 by 2 and the arc $x_2 x_3$ by 1. Then we colour x_3, $x_3 x_4$, ..., x_k. It remains to colour $x_k x_1$ and x_1. Two cases may happen: If we can colour $x_k x_1$ by 1 or 2, we do it and colour x_1 by 2 or 1 respectively. Otherwise the set of colours assigned to x_k and $x_{k-1} x_k$ is $\{1, 2\}$. Hence, we colour $x_k x_1$ with 3, x_1 by 1, and recolour $x_1 x_2$ by 2 and x_2 by 3. □

Lemma 13 *Let D be a subcubic digraph with no vertex of outdegree two and indegree one. Assume that every arc a has a list of colours $L(a) \subset \{1, 2, 3\}$ such that:*

- *If the head of a is a sink s (a is called a final arc), $|L(a)| \geq d^-(s)$.*
- *If a is not a final arc and the tail of a is a source (a is called an initial arc), $|L(a)| \geq 2$.*
- *In other cases $|L(a)| = 3$.*
- *If a vertex is the head of at least two initial arcs the union of their lists of colours contains at least three colours.*

- *If all the vertices of an odd circuit are the tails of initial arcs, the union of the lists of colours of these initial arcs contains at least three colours.*

Then D has a directed star L-colouring.

Proof. We colour the graph inductively. Consider a terminal strong component C of D. Since D has no vertex with indegree one and outdegree two, C induces either a singleton or a circuit.

1) Assume that C is a singleton v which is the head of a unique arc $a = uv$. If u has indegree 0, colour a with a colour of its list. If u has indegree 1, and thus total degree 2, colour a by the colour of its list and remove this colour from the list of the arc with head u. If u is the head of e and f, observe that $L(e)$ and $L(f)$ have at least two colours and their union have at least three colours. To conclude, colour a with a colour in its list, remove this colour from $L(e)$ and $L(f)$, remove a, split u into two vertices, one with head e, and the other with head f. Now, choose in their respective lists different colours for the arcs e and f to form the new list $L(e)$ and $L(f)$.

2) Assume that C is a singleton v which is the head of several arcs, including $a = uv$. In this case, we reduce $L(a)$ to a single colour, remove this colour from the other arcs with head v and split v into v_1 which becomes the head of a, and v_2 which becomes the head of the other arcs.

3) Assume that C is a circuit. Every arc entering C has a list of at least two colours. We can apply Lemma 12 to conclude.

□

Proof of Theorem 11. Assume for contradiction that the digraph D has directed star arboricity more than three and is minimum with respect to the number of arcs for this property. Observe that D has no source, otherwise we simply delete it with its incident arcs, apply induction and extend the colouring since arcs leaving from a source can be coloured arbitrarily. Let D_1 be the subdigraph of D induced by the vertices of indegree at most 1. We denote by D_2 the digraph induced by the other vertices, and by $[D_i, D_j]$ the set of arcs with tail in D_i and head in D_j. We claim that D_1 contains no even circuit. If not, we simply remove the arcs of this even circuit, apply induction and extend the colouring to the arcs of the even circuit since every arc of the circuit has two colours available.

A *critical set* of vertices of D_2 is either a vertex of D_2 with indegree at least two in D_1, or an odd circuit of D_2 having all its inneighbours in D_1. Observe that critical sets are disjoint. For every critical set S, we select two arcs entering S from D_1, called *selected arcs* of S.

Let D' be digraph induced by the arc set $A' = A(D_1) \cup [D_2, D_1]$. Now we define a *conflict graph* on the arcs of D' in the following way:

- Two arcs xy, yv of D' are in conflict, called *normal conflict* at y.
- Two arcs xy, uv of D' are also in conflict if there exists two selected arcs of the same set S with tails y and v. These conflicts are called *selected conflicts* at y and v.

Let us analyse the structure of the conflict graph. Observe first that an arc is in conflict with three arcs : one normal conflict at its tail and at most two (normal or selected) at its head.

We claim that there is no K_4 in the conflict graph. Suppose there is one, then there are four arcs pairwise in conflict. Since each arc has degree 3, it has a normal conflict at its tail, the digraphs induced by these four arcs contains a circuit. It cannot be a circuit of even length (2 or 4) so it has length 3. It follows that the four arcs a, b, c, d are as in Figure 1 below. Let D^* be the digraph obtained from D by removing the arcs a, b, c, d and their four incident vertices. By minimality of D, D^* admits a directed star 3-colouring which can be extended to D as depicted below depending if the two leaving arcs are coloured the same or differently. This proves the claim.

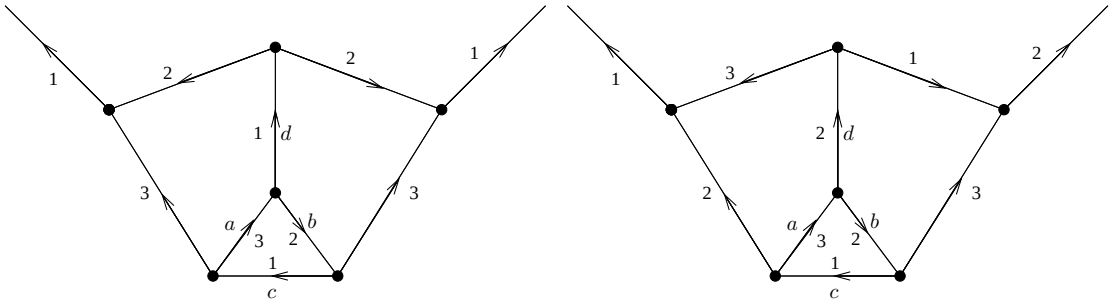


Figure 1: A K_4 in the conflict graph and the two ways of extending the colouring.

Brooks Theorem asserts that every subcubic graph without K_4 is 3-colourable. So the conflict graph admits a 3-colouring c. This gives a colouring of the arcs of D'. Let D'' be the digraph and L be the list-assignment on the arcs of D'' obtained as follow:

- Remove the arcs of D_1 from D,
- Assign to each arc of $[D_2, D_1]$ the singleton list containing the colour it has in D',
- For each arc uv of $[D_1, D_2]$, there is a unique arc tu in $A(D')$, so assign the list $L(uv) = \{1, 2, 3\} \setminus c(tu)$.
- Assign the list $\{1, 2, 3\}$ to the other arcs.
- If there are vertices with indegree one and outdegree two (they were in D_1), split each of them into one source of degree two and a sink of degree one.

Note that there is a trivial one-to-one correspondence between $A(D'')$ and $A(D) \setminus A(D')$. By the definition of conflict graph and D'', one can easily check that D'' and L satisfies the condition of Lemma 13. Hence D'' admits a directed star L-colouring which union with c is a directed star 3-colouring of D, a contradiction. □

3.2 Acircuitic directed star arboricity

A directed star colouring is *acircuitic* if there is no *bicoloured* circuits, i.e. circuits for which only two colours appears on its arcs. The *acircuitic directed star arboricity* of a digraph D is the minimum number k of colours such that there exists an acircuitic directed star k-colouring of D. In this subsection, we give a short alternative proof of the following theorem due to Pinlou and Sopena.

Theorem 14 (Pinlou and Sopena [9]) *Every subcubic oriented graph has acircuitic directed star arboricity at most* 4.

In order to prove this theorem, we need the following lemma.

Lemma 15 *Let D be an acyclic subcubic digraph. Let L be a list-assignment on the arcs of D such that for every arc uv, $|L(uv)| \geq d(v)$. Then D admits a directed star L-colouring.*

Proof. We prove the result by induction on the number of arcs of D, the result holding trivially if D has no arcs.

Since D is acyclic, it has an arc xy with y a sink. Let a be a colour in $L(xy)$. For any arc e distinct from xy, set $L'(e) = L(e) \setminus \{a\}$ if e incident to xy (and thus has head in $\{x, y\}$ since y is a sink), and $L'(e) = L(e)$ otherwise. Then in $D' = D - xy$, we have $|L'(uv)| \geq d(v)$. Hence, by induction hypothesis, D' admits a directed star L'-colouring that can be extended in a directed star L-colouring of D by colouring xy with a. □

Proof of Theorem 14.

Let V_1 be the set of vertices of outdegree at most 1 and $V_2 = V \setminus V_1$. Then every vertex of V_2 has outdegree at least 2 and so indegree at most 1.

Let M be the set of arcs with tail in V_1 and head in V_2. Colour all the arcs of M with 4. Moreover for every circuit C in $D[V_1]$ and $D[V_2]$ choose an arc $e(C)$ and colour it by 4. Note that, by definition of V_1 and V_2, the arc $e(C)$ is not incident to any arc of M and C is the unique circuit containing $e(C)$. Let us denote M_4 the set of arcs coloured 4. Then M_4 is a matching and $D - M_4$ is acyclic.

We shall now find a directed star colouring of $D - M_4$ with $\{1, 2, 3\}$ that creates any bicoloured circuit. If such a circuit exists, 4 would be one of its colour because $D - M_4$ is acyclic and all its arcs coloured 4 would be in M because the arcs of $M_4 \setminus M$ is in a unique circuit which has a unique arc coloured 4. Hence we just have to be careful when colouring arcs in the digraph induced by the endvertices of the arcs of M.

Let us denote the arcs of M by x_iy_i, $1 \leq i \leq p$ and set $X = \{x_i, 1 \leq i \leq p\}$ and $Y = \{y_i, 1 \leq i \leq p\}$. Then $x_i \in V_1$ and $y_i \in V_2$. Let E' be the set of arcs with tail in Y and head in X. Let H be the graph with vertex set E' such that an arc y_ix_j is adjacent to an arc y_kx_l if

(a) either $k = l$,

(b) or $j = k$ and $i > j$ and $l > j$.

Since a vertex of X has indegree at most 2 and a vertex of Y has outdegree at most 2, H has maximum degree 3. Moreover H contains no K_4 because two arcs of E' with same tail y_k are not adjacent in H. Hence, by Brooks Theorem, H has a vertex-colouring in $\{1, 2, 3\}$ which is corresponds to a colouring c of the arcs of E'. Since (a) is satisfied c is a directed star colouring. Moreover this colouring creates no bicoloured circuits: indeed a circuit contains a subpath $y_ix_jy_jx_l$ with $i > j$ and $k > j$, whose three arcs are coloured differently by (b).

Let $D' = D - (M_4 \cup E')$. For any arc uv in D', let $L(uv) = \{1, 2, 3\} \setminus \{c(wv) \mid wv \in E'\}$. The set $L(uv)$ is the set of colours in $\{1, 2, 3\}$ that may be assigned to uv without creating any conflict with the already coloured arcs. D' is acyclic and $|L(uv)| \geq d(v)$, so by Lemma 15, it admits a directed star L-colouring and thus D has an acircuitic directed star colouring in $\{1, 2, 3, 4\}$. □

Remark 16 Note that in the acircuitic directed star 4-colouring provided in the proof of Theorem 14 the arcs coloured 4 form a matching.

4 Directed star arboricity of digraphs with maximum in and outdegree two

The goal of this section is to prove that every digraph with outdegree and indegree at most two has directed star arboricity at most four.

Theorem 17 *Let D be a digraph. If $\Delta^- \leq 2$ and $\Delta^+ \leq 2$, then $dst(D) \leq 4$.*

Thus, Conjecture 4 holds for $k = 2$ and hence for all even k. However, the class of digraphs with in and outdegree at most two is certainly not an easy class with respect to directed star arboricity, as we will show in Section 5.

In order to prove Theorem 17, it suffices to show that D contains a galaxy G which spans all the vertices of degree four. Then $D' = D - A(G)$ has maximum degree at most 3. So, by Theorem 11, $dst(D') \leq 3$, so $dst(D) \leq 4$. Hence Theorem 17 is directly implied by the following lemma:

Lemma 18 *Let D be a digraph with maximum indegree and outdegree two. Then D contains a galaxy which spans the set of vertices with degree four.*

In order to prove this lemma, we need some preliminaries:
Let V be a set. An *ordered digraph* on V is a pair $(\leq, D)$ where:

- $\leq$ is a partial order on V.
- D is a digraph with vertex set V.
- D contains the Hasse diagram of $\leq$ (i.e. when $x \leq y \leq z$ implies $x = y$ or $y = z$, then xz is an arc of D).
- If xy is an arc of D, the vertices x, y are $\leq$-comparable.

The arcs xy of D thus belong to two different types: the *forward arcs* when $x \leq y$, and the *backward arcs* when $y \leq x$.

Lemma 19 *Let $(\leq, D)$ be an ordered digraph on V. Assume that every vertex is the tail of at most one backward arc and at most two forward arcs and that the indegree of every vertex of D is at least 2, except possibly one vertex x with indegree 1. Then D contains two arcs ca and bd such that $a \leq b \leq c$, $b \leq d$ and $c \not\leq d$, all four vertices being distinct except possibly $a = b$.*

Proof. Let us consider a counterexample with minimum $|V|$.

An *interval* is a subset I of V which has a minimum m and a maximum M such that $I = \{z : m \leq z \leq M\}$. An interval I is *good* if every arc with tail in I and head outside I has tail M and every backward arc in I has tail M.

Let I be an interval of D. The digraph D/I obtained from D by *contracting* I is the digraph with vertex set $(V \setminus I) \cup \{v_I\}$ such that xy is an arc if and only either $v_I \notin \{x, y\}$ and $xy \in A(D)$, or $x = v_I$ and there exists $x_I \in I$ such that $x_I y \in A(D)$, or $y = v_I$ and there exists $y_I \in I$ such that $xy_I \in A(D)$.

Similarly, the binary relation $\leq_{/I}$ obtained from $\leq$ by *contracting* I is the binary relation on $(V \setminus I) \cup \{v_I\}$ such that $x \leq_{/I} y$ if and only if either $v_I \notin \{x, y\}$ and $x \leq y$, or $x = v_I$ and there exists $x_I \in I$ such that $x_I \leq y$, or $y = v_I$ and there exists $y_I \in I$ such that $x \leq y_I$. We claim that if I is good then $\leq_{/I}$ is a partial order. Indeed suppose it is not, there are two elements u and t such that $u \leq_{/I} v_I$, $v_I \leq t$ and $u \not\leq_{/I} t$. Then $M \not\leq t$. Let $a \in I$ be the a maximal element of I such that $a \leq t$, d be a successor of a in I and c a successor of a not in I (it exists as $t \notin I$ and $a \leq t$ and maximal in I with this property). Then d and c are incomparable and ac and ad are in the Hasse diagram of $\leq$ then because I is good it follows that ca and ad are arcs of D, which is impossible as D is a counterexample.

Hence if I is a good interval, $(\leq_{/I}, D/I)$ is an ordered digraph. Note that if $x \leq_{/I} v_I$ then $x \leq M$ with M the maximum of I. The crucial point is that if I a good interval of D for which the conclusion of Lemma 19 holds for $(\leq_{/I}, D/I)$, then it holds for $(\leq, D)$. Indeed, suppose there exists two arcs ca and bd of D/I such that $a \leq_{/I} b \leq_{/I} c$, $b \leq_{/I} d$ and $c \not\leq_{/I} d$. Note that since I is good $v_I \neq c$. Let M be the maximum of I.
If $v_I \notin \{a, b, c, d\}$, then ca and bd gives the conclusion for D.
If $v_I = a$ then cM is an arc. Let us show that $M \leq b$. Indeed let x be a maximal vertex in I such that $x \leq b$ and y a minimal vertex such that $x \leq y \leq b$. Since the Hasse diagram of $\leq$ is included in D then xy is an arc so $x = M$ since I is good. Thus cM and bd are the desired arcs.

If $v_I = b$ then Md is an arc and $a \leq M$, so ca and Md are the desired arcs.
If $v_I = d$ then there exists $d_I \in I$ such that bd_I, so ca and bd_I are the desired arcs.

Hence to get a contradiction, it is sufficient to find a good interval I such that $(\leq_{/I}, D/I)$ satisfies the hypotheses of Lemma 19.

Observe that there are at least two backward arcs. Indeed, if there are two minimal elements for $\leq$, there are at least three backward arcs entering these vertices (since one of them can be x). And if there is a unique minimum m, by letting m' minimal in $V \setminus m$, at least two arcs are entering m, m'.

Let M be a vertex which is the tail of a backward arc and which is minimal for $\leq$ for this property. Since two arcs cannot have the same tail, M is not the maximum of $\leq$ (if any). Let Mm be the backward arc with tail M.

We claim that the interval J with minimum m and maximum M is good. Indeed, by the definition of M, no backward arc has its tail in $J \setminus \{M\}$. Moreover, any forward arc bd with its tail in $J \setminus \{M\}$ and its head outside J would give our conclusion (with $a = m$ and $c = M$), a contradiction.

Now consider a good interval I with maximum M which is maximal with respect to inclusion. We claim that if $x \in I$, then there is at least one arc entering I, and if $x \notin I$, there are at least two arcs entering I with different tails.

Call m_1 the minimum of I and m_2 any minimal element of $I \setminus m_1$. First assume that x is in I. There are at least three arcs with heads m_1 or m_2. One of them is m_1m_2, one of them can be with tail M, but there is still one left with tail not in I. Now assume that x is not in I. There are at least two arcs with heads m_1 or m_2 and tails not in I. If the tails are different, we are done. If the tails are the same, say v, observe that vm_1 and vm_2 are both backward of both forward (otherwise v would be in I). Since both can not be backward vm_1 and vm_2 are forward. Hence the interval with minimum v and maximum M is a good interval, contradicting the maximality of I. This proves the claim.

This claim implies that $(\leq_{/I}, D/I)$ satisfies the hypotheses of Lemma 19, yielding a contradiction. □

Proof of Theorem 18. Let G be a galaxy of D which spans a maximum number of vertices of degree four. Suppose for contradiction that some vertex x with degree four is not spanned.

An *alternating path* is an oriented path ending at x, starting by an arc of G, and alternating with arcs of G and arcs of $A(D) \setminus A(G)$. We denote by $\mathcal{A}$ the set of arcs of G which belong to an alternating path.

Claim 1 *Every arc of $\mathcal{A}$ is a component of G.*

Proof. Indeed, if uv belongs to $\mathcal{A}$, it starts some alternating path P. Thus, if u has outdegree more than one in G, the digraph with set of arcs $A(G) \triangle A(P)$ is a galaxy and spans $V(G) \cup x$. □

Claim 2 *There is no circuits alternating arcs of $\mathcal{A}$ and arcs of $A(D) \setminus \mathcal{A}$.*

Proof. Assume that there is such a circuit C. Consider a shortest alternating path P starting with some arc of $\mathcal{A}$ in C. Now the digraph with arcs $A(G) \triangle (A(P) \cup A(C))$ is a galaxy which spans $V(G) \cup x$, contradicting the maximality of G. □

We now endow $\mathcal{A} \cup x$ with a partial order structure by letting $a \leq b$ if there exists an alternating path starting at a and ending at b. The fact that this relation is a partial order relies on Claim 2. Observe that x is the maximum of this order.

We also construct a digraph $\mathcal{D}$ on vertex set $\mathcal{A} \cup x$ and all arcs $uv \to st$ such that us or vs is an arc of D (and $uv \to x$ such that ux or vx is an arc of D).

Claim 3 *The pair $(\mathcal{D}, \leq)$ is an ordered digraph. Moreover an arc of $\mathcal{A}$ is the tail of at most one backward arc and two forward arcs and x is the tail of at most two backward arcs.*

Proof. The fact that the Hasse diagram of $\leq$ is contained in $\mathcal{D}$ follows from the fact that if $uv \leq st$ belongs to the Hasse diagram of $\leq$, there is an alternating path starting by $uvst$, in particular, the arc vs belongs to D, and thus $uv \to st$ in $\mathcal{D}$.

Suppose that $uv \to st$ and then vs or us is an arc of D. If vs is an arc, then because there is no alternating circuit, st follows uv on some alternating path so $uv \leq st$. In this case, $uv \to st$ is forward. If us is an arc of D, we claim that $st \leq uv$. Indeed, if an alternating path P starting at st does not contain uv, the galaxy with arcs $(A(G)\triangle A(P)) \cup \{us\}$ spans $V(G) \cup x$ contradicting the maximality of G. In this case, $uv \to st$ is backward.

It follows that an arc uv of $\mathcal{A}$ is the tail of at most one backward arc since this arc and uv are the two arcs leaving u in D and the tail of at most two forward arcs since v has outdegree at most 2. Furthermore, since x has outdegree at most two, it follows that x is the tail of at most two backward arcs. □

Claim 4 *The indegree of every vertex of $\mathcal{D}$ is two.*

Proof. Let uv be a vertex of $\mathcal{D}$ which starts an alternating path P. If u has indegree less than two, and thus does not belong to the set of vertices of degree four, the galaxy with arcs $A(G)\triangle A(P)$ spans more vertices of degree four than G, a contradiction. Let s and t be the two inneighbours of u in D. An element of $\mathcal{A} \cup x$ contains s otherwise the galaxy with arcs $(A(G)\triangle A(P)) \cup \{su\}$ spans $V(G) \cup x$ and contradicts the maximality of G. Similarly an element of $\mathcal{A} \cup x$ contains t.

Observe that the same element of $\mathcal{A} \cup x$ cannot contain both s and t (either the arc st or the arc ts), otherwise the arcs su and tu would be both backward or forward, which is impossible. □

At this stage, in order to apply Lemma 19, we just need to insure that the backward outdegree of every vertex is at most one. Since the only element of $\mathcal{D}$ which is the tail of two backward arcs is x, we simply delete any of these two backward arcs. The indegree of a vertex of $\mathcal{D}$ decreases by one but we are still fulfilling the hypothesis of Lemma 19.

Hence according to this lemma, $\mathcal{D}$ contains two arcs ca and bd such that $a \leq b \leq c$, $b \leq d$ and $c \not\leq d$. Keep in mind that a, b, c, d are elements of $\mathcal{A} \cup x$. In particular, there is an alternating path P containing a, b, d (in this order) which does not contain c. Setting $a = a_1a_2$ and $c = c_1c_2$, note that the backward arc ca corresponds to the arc c_1a_1 in D. We reach a contradiction by considering the galaxy with arcs $(A(G)\triangle A(P)) \cup \{c_1a_1\}$ which spans $V(D') \cup x$. □

5 Complexity

The digraphs with directed star arboricity 1 are the galaxies, so one can polynomially decide if $dst(D) = 1$. Deciding whether $dst(D) = 2$ or not is also easy since we just have to check that the conflict graph (with vertex set the arcs of D, two distinct arcs xy, uv being in conflict when $y = u$ or $y = v$) is bipartite. However for larger value, as expected, it is $\mathcal{NP}$-complete to decide if a digraph has directed star arboricity at most k. This is illustrated by the next result:

Theorem 20 *The following problem is $\mathcal{NP}$-complete:*
INSTANCE: A digraph D with $\Delta^+(D) \leq 2$ and $\Delta^-(D) \leq 2$.
QUESTION: Is $dst(D)$ at most 3?

Proof. The proof is a reduction to 3-edge-colouring of 3-regular graphs. To see this, consider a 3-regular graph G. It admits an orientation D such that every vertex has in and outdegree at least 1. Let D' be the digraph obtained from D by replacing every vertex with indegree 1 and outdegree 2 by the subgraph H depicted in Figure 2 which has also one entering arc (namely a) and two leaving arcs (b and

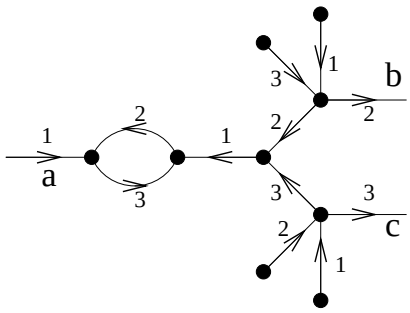


Figure 2: The graph H and one of its directed star 3-colouring

c). It is easy to check that in any directed star 3-colouring of H, the three arcs a, b and c get different colours. Moreover if these three arcs are precoloured with three different colours, we can extend this to a directed star 3-colouring of H. Such a colouring with a coloured 1, b coloured 2 and c coloured 3 is given in Figure 2. Furthermore, a vertex with indegree 2 and outdegree 1 must have its three incident arcs coloured differently in a directed star 3-colouring. So $dst(D') = 3$ if and only if G is 3-edge colourable. □

6 Multiple fibers

In this section we consider the problem with $n \geq 2$ fibers. More precisely, we give some bounds on $\lambda_n(m,k)$. We first give a lower bound on $\lambda_n(m,k)$.

Proposition 21 $\lambda_n(m,k) \geq \left\lceil \frac{m}{n} \left\lceil \frac{k}{n} \right\rceil + \frac{k}{n} \right\rceil$

Proof. Consider the following m-labelled digraph $G_{n,m,k}$ with vertex set $X \cup Y \cup Z$ such that :

- $|X| = k$, $|Y| = 2^{(m+1)k^2}$ and $|Z| = m\binom{|Y|}{k}$.
- For any $x \in X$ and $y \in Y$ there is an arc xy (of whatever label).
- For every set S of k vertices of Y and integer $1 \leq i \leq m$, there is a vertex z_S^i in Z which is dominated by all the vertices of S via arcs labelled i.

Suppose there exists an n-fiber colouring of $G_{n,m,k}$ with $c < \left\lceil \frac{m}{n} \left\lceil \frac{k}{n} \right\rceil + \frac{k}{n} \right\rceil$ colours. For $y \in Y$ and $1 \leq i \leq m$, let $C_i(y)$ be the set of colours assigned to the arcs labelled i leaving y. For $0 \leq j \leq n$, let P_j the set of colours used on j arcs entering y (and necessarily with two different fibers). Then $\sum_{j=0}^{n} j|P_j| = k$ as k arcs enter y. Moreover $(P_0, P_1, \ldots, P_n)$ is a partition of the set of colours so $\sum_{j=0}^{n} |P_j| = c$. Now each colour of P_j may appear in at most $n - j$ of the $C_i(y)$, so

$$\sum_{i=1}^{m} |C_i(y)| \leq \sum_{j=0}^{n} (n-j)|P_j| = n \sum_{j=0}^{n} |P_j| - \sum_{j=0}^{n} j|P_j| = cn - k.$$

Because $|Y| = 2^{(m+1)k^2}$, there is a set S of k vertices y of Y having the same m-uple $(C_1(y), \ldots, C_m(y)) = (C_1, \ldots, C_m)$. Without loss of generality, we may assume $|C_1| = min\{|C_i| \mid 1 \leq i \leq m\}$. Hence $|C_1| \leq \frac{cn-k}{m}$. But the vertex z_S^1 has indegree k so $|C_1| \geq k/n$. Since $|C_1|$ is an integer, we have

$\left\lfloor \frac{cn-k}{m} \right\rfloor \geq |C_1| \geq \lceil k/n \rceil$. So $c \geq \frac{m}{n}\left\lceil \frac{k}{n} \right\rceil + \frac{k}{n}$. Since c is an integer, we get $c \geq \left\lceil \frac{m}{n}\left\lceil \frac{k}{n} \right\rceil + \frac{k}{n} \right\rceil$, a contradiction. □

Note that the graph $G_{n,m,k}$ is acyclic. The following lemma shows that, if $m \geq n$, one cannot expect better lower bounds by considering acyclic digraphs. Indeed $G_{n,m,k}$ is the m-labelled acyclic digraph with indegree at most k for which an n-fiber colouring requires the more colours.

Lemma 22 *Let D be an acyclic m-labelled digraph with $\Delta^- \leq k$. If $m \geq n$ then $\lambda_n(D) \leq \left\lceil \frac{m}{n}\left\lceil \frac{k}{n} \right\rceil + \frac{k}{n} \right\rceil$.*

Proof. Since D is acyclic, its vertex set admits an ordering $(v_1, v_2, \ldots, v_p)$ such that if $v_j v_{j'}$ is an arc then $j < j'$.

By induction on q, we shall find an n-fiber colouring of $D[\{v_1, \ldots, v_q\}]$ together with sets $C_i(v_r)$, $1 \leq i \leq m$ and $1 \leq r \leq q$, of $\lceil k/n \rceil$ colours such that, in the future, assigning a colour in $C_i(v_r)$ to an arc labelled i leaving v_r will fulfill the condition of n-fiber colouring at v_r.

Starting the process is easy. We may take as $C_i(v_1)$ any $\lceil k/n \rceil$-sets such that a colour appears in at most n of them.

Suppose now that we have an n-fiber colouring of $D[\{v_1, \ldots, v_{q-1}\}]$ and that, for $1 \leq i \leq m$ and $1 \leq r \leq q-1$, the set $C_i(v_r)$ is determined. Let us colour the arcs entering v_q. Each of these arcs $v_r v_q$ may be assigned one of the $\lceil k/n \rceil$ colours of $C_{l(v_r v_q)}(v_r)$. Since a colour may be assigned to n arcs (using different fibers) entering v_q, one can assign a colour and fiber to each such arc. It remains to determine the $C_i(v_q)$, $1 \leq i \leq m$.

For $0 \leq j \leq n$, let P_j be the set of colours assigned to j arcs entering v_q. Let $N = \sum_{i=0}^{n}(n-j)|P_j|$ and $(c_1, c_2, \ldots, c_N)$ be a sequence of colours such that each colour of P_j appears exactly $n-j$ times and consecutively. For $1 \leq i \leq m$, set $C_i(v_q) = \{c_a \mid a \equiv i \mod m\}$. As $n \leq m$, a colour appears at most once in each $C_i(v_q)$. Moreover, $N = n\left\lceil \frac{m}{n}\left\lceil \frac{k}{n} \right\rceil + \frac{k}{n} \right\rceil - k \geq m\left\lceil \frac{k}{n} \right\rceil$. So for $1 \leq i \leq m$, $|C_i(v_q)| \geq \left\lceil \frac{k}{n} \right\rceil$. □

Lemma 22 shows that the lower bound of Proposition 21 is tight for acyclic digraphs. In fact, we conjecture that is it tight also for digraphs in general:

Conjecture 23 $$\lambda_n(m,k) = \left\lceil \frac{m}{n}\left\lceil \frac{k}{n} \right\rceil + \frac{k}{n} \right\rceil$$

We now establish an upper bound on $\lambda_n(m,k)$ for general digraphs. To do so, we first give an upper bound on $\lambda_n(D)$ for m-labelled digraphs with bounded in- and outdegree In this case, one can derive from the following theorem of Guiduli that "few" colours are needed. Note that the graphs $G_{n,m,k}$ requires lots of colours but have very large outdegree.

Theorem 24 (Guiduli [6]) *If $\Delta^-, \Delta^+ \leq k$ then $dst(D) \leq k + 20\log k + 84$. Moreover D admits a directed star colouring with $k + 20\log k + 84$ colours such that for each vertex v there are at most $10\log k + 42$ colours assigned to its leaving arcs.*

The proof of Guiduli's Theorem can be modified to obtain the following statement for m-labelled digraphs.

Theorem 25 *Let $f(n,m,k) = \left\lceil \dfrac{k + (10m^2+5)\log k + 80m^2 + m + 21}{n} \right\rceil$ and D be an m-labelled digraph with $\Delta^-, \Delta^+ \leq k$. Then $\lambda_n(D) \leq f(n,m,k)$. Moreover D admits an n-fiber colouring with $f(n,m,k)$ such that for each vertex v and each label l, there are at most $g(m,k) = \lceil (10m+5)\log k + 40m + 21 \rceil$ colours assigned to the arcs labelled l leaving v.*

Note that Theorem 25 in the case $n = m = 1$ is a bit better than Theorem 24. Indeed it shows that if $\Delta^-, \Delta^+ \leq k$ then $dst(D) \leq k + 15\log k + 102$. It is due to Lemma 7 which is a bit better than Guiduli's one because it uses Theorem 7 ($dst \leq 2\Delta^- + 1$) whereas Guiduli uses $dst \leq 3\Delta^-$. However the method is identical.

Definition 26 Given a family of sets $\mathcal{F} = (A_i, i \in I)$ A *transversal* of $\mathcal{F}$ is a family of distinct elements $(t_i, i \in I)$ with $\forall i, t_i \in A_i$.

Lemma 27 *Let D be a m-labelled digraph with $\Delta^- \leq k$. Suppose that for each vertex v, there are m disjoint lists $L_v^1, ..., L_v^m$ of c colours each being a subset of $\{1, ..., k + c\}$. If for each vertex v, the family $\{L_y^i \mid yx \in E(D)$ and yx is labelled $i\}$ has a transversal, then there is a 1-fiber colouring of D with $k + (2m^2 + 1)c + m$ colours such that for each vertex v and label l, at most $(2m + 1)c + 1$ colours are assigned to arcs labelled l leaving v.*

Proof. Using the transversal to colour the entering arcs at each vertex, we obtain a colouring with few conflicts. Indeed there is no conflict between arcs entering a same vertex. So the only possible conflict are between an arc entering a vertex v and an arc leaving v. Since arcs leaving v use at most $m.c$ colours (those of $L_v^1 \cup ... \cup L_v^m$), there are at most $m.c$ arcs entering v having the same colour as an arc leaving c. Removing such entering arcs for every vertex v, we obtain a digraph D' for which the colouring with the $k + c$ colours is a 1-fiber colouring. We now want to colour the arcs of $D - D'$ with few extra colours. Consider a label $1 \leq l \leq m$, let D'_l be the digraph induced by the arcs of $D - D'$ labelled l. Then D'_l has indegree at most $m.c$. By Theorem 7, we can partition D'_l in $2m.c + 1$ star forests. Thus D can be 1-fiber coloured with $k + c + m(2m.c + 1)$ colours. Moreover, in the above described colouring, arcs labelled l leaving a vertex v have a colour in L_v^l or corresponding to one of the $2m.c + 1$ star forests of D'_l. So at most $(2m + 1)c + 1$ colours are assigned to arcs labelled l leaving v. □

Theorem 28 (N. Alon, C. McDiarmid, B. Reed, 1992 [2]) *Let k and c be positive integers with $k \geq c \geq 5 \log k + 20$. Choose independent random subsets $S_1, \ldots, S_k$ of $X = \{1, \ldots, k + c\}$ as follows. For each i choose S_i by performing c independent uniform samplings from X. Then the probability that $S_1, \ldots, S_k$ do not have a transversal is at most $k^{3-\frac{c}{2}}$*

Proof of Theorem 25. It suffices to prove the result for $n = 1$. Indeed we can extend a 1-fiber colouring satisfying the conditions of the theorem to a n-fiber colouring satisfying the conditions by replacing the colour $qn + r$ with $1 \leq r \leq n$ by the colour $q + 1$ on fiber r.

Let $c = \lceil 5 \log k + 20 \rceil$ We can assume $k \geq m.c$. For all vertices x, select $m.c$ different ordered elements $e_1, e_2, \cdots, e_{m.c}$ independently and uniformly. For all $1 \leq i \leq m$, let $L_x^i = \{e_{ci+1}, \cdots, e_{c(i+1)}\}$. Each set has the same distribution as the c elements where chosen uniformly and independently.

Let A_x be the event that the family $\{L_y^i \mid yx \in E(D)$ and yx is labelled $i\}$ fails to have a transversal. By Theorem 28, $P(A_x) \leq k^{3-c/2}$. Furthermore, the event A_x is independent of all A_y for which there is no vertex z such that both zx and zy are in $E(D)$. The dependency graph for the events has degree at most k^2 so we can apply Lovász Local Lemma. We obtain that there exists a family of lists satisfying conditions of Lemma 27. This lemma gives the desired colouring. □

Any digraph D may be decomposed into an acyclic digraph D_a and an *eulerian* digraph D_e, that is such that for every vertex v, $d^-_{D_e}(V) = d^+_{D_e}(v)$. Indeed consider an eulerian subdigraph D_e of D which has a maximum number of arcs. Then the digraph $D_a = D - D_e$ is necessarily acyclic. Hence by Theorems 25 and 22, if $m \geq n$ then $\lambda_n(D) \leq \lceil \frac{m}{n} \lceil \frac{k}{n} \rceil + \frac{k}{n} \rceil + f(n, m, k)$. But we will now lessen this bound by roughly $\frac{k}{n}$.

Theorem 29 *If $n \leq m$, then*

$$\lambda_n(m, k) \leq \left\lceil \frac{m}{n} \left\lceil \frac{k}{n} \right\rceil + \frac{k}{n} \right\rceil + 2m \frac{\lceil (10m + 5) \log k + 40m + 21 \rceil}{n}.$$

Proof. Let D be an m-labelled digraph with $\Delta^-(D) \leq k$. Consider a decomposition of D into an eulerian digraph D_e and an acyclic digraph D_a. We first apply Theorem 25 and n-fiber colour the arcs of D_e with $f(n,m,k)$ colours such that at most $g(m,k)$ colours are assigned to the arcs leaving each vertex.

We shall extend the n-fiber colouring of D_e to the arcs of D_a in a way similar to the proof of Lemma 22, i.e. we will assign to each vertex v sets $C_i(v)$, $1 \leq i \leq m$ of $\lceil k/n + m.g(m,k) \rceil$ colours such that an arc labelled i leaving v will be labelled using a colour in $C_i(v)$.

Let $(v_1, \ldots, v_n)$ be an ordering of the vertices of A such that if $v_j v_{j'}$ is an arc then $j < j'$.

We start to build the $C_i(v_1)$ with the colours assigned to the leaving arcs of v_1 labelled i. The vertex v_1 has at most k entering arcs. Each of them forbid one type (colour, fiber). In the colouring of D_e induced by Theorem 25, there are at most $m.g(m,k)$ types assigned to the arcs leaving v_1. So there are at least $\lceil \frac{mk}{n^2} \rceil + \lceil \frac{k}{n} \rceil + 2m\frac{g(m,k)}{n} - k - m.g(m,k) \geq m \lceil \frac{k}{n} \rceil + m.g(m,k)$ types unused at vertex v_1. Since $n \leq m$, we can partition these types into m sets of size at least $\frac{k}{n}$ such that no two types having the same colour are in the same set. These sets are the $C_i(v_1)$.

Suppose that the sets have been defined for v_1 up to v_{q-1} and that all the arcs $v_i v_j$ for $i < q$ and $j < q$ have a colour. We now give a colour to the arcs of type $v_i v_q$ for $i < q$.

There are k_e arcs entering v_q in D_e which are already coloured. So it remains to give a colour to $k_a \leq k - k_e$ arcs. Each uncoloured arc may be assigned a colour in a list of size at least $\lceil \frac{k}{n} + m.g(m,k) \rceil$. This gives a choice between $n.\lceil \frac{k}{n} + m.g(m,k) \rceil$ different types. k_e types are forbidden by the entering arcs in D_e while at most $m.g(m,k)$ types are forbidden by the leaving arcs in D_e. Then it remains at least $n.\lceil \frac{k}{n} + m.g(m,k) \rceil - k_e - m.g(m,k) \geq k_a$ types for each entering arcs of D_a. So one can assign distinct available colours to each of the k_a arcs entering v_q.

We then build the $C_i(v_q)$ as we did for v_1.

This process finished, we obtain an n-fiber colouring of D using $\lceil \frac{mk}{n^2} \rceil + \lceil \frac{k}{n} \rceil + 2m\frac{g(m,k)}{n}$ colours. □

Theorem 29 gives an upper bound on $\lambda_n(m,k)$ when $m \geq n$. We now give one when $m < n$.

Proposition 30 *If $m < n$ then $\lambda_n(m,k) \leq \left\lceil \frac{k}{n-m} \right\rceil$.*

Proof. Let D be an m-labelled digraph with $\Delta^- \leq k$. For each vertex v, we give to its entering arcs a colour such that none of them is used more than $n - m$ times. This is possible as there are at most $k \leq (n-m)\left\lceil \frac{k}{n-m} \right\rceil$ arcs entering v. Then we have $in(v,\lambda) \leq n - m$. Moreover each arc vw is given a colour by w. Since D is m-labelled, a colour λ can be used to colour an arc of at most m different labels, i.e. $out(v,\lambda) \leq m$. Consequently $in(v,\lambda) + out(v,\lambda) \leq n$. This gives a proper n-fiber colouring. □

References

[1] I. Algor and N. Alon, The star arboricity of graphs, *Discrete Mathematics* **75** (1989), 11–22.

[2] N. Alon, C. McDiarmid, B. Reed, Star arboricity, *Combinatorica* **12** (1992), 375–380.

[3] R. Brandt, *Multicasting using WDM in Multifiber Optical Star Networks*, Thesis, UCSB, Sep 2003.

[4] R. Brandt and T. F. Gonzalez, Wavelength assignment in multifiber optical star networks under the multicasting communication mode, *Journal of Interconnection Networks* **6** (2005), 383–405.

[5] A. Frank, Covering branchings, *Acta Sci. Math. (Szeged)* **41** (1979), 77–81.

[6] B. Guiduli, On incidence coloring and star arboricity of graphs, *Discrete Mathematics* **163** (1997), 275–278.

[7] A. Lardies, R. Gupta and R. Patterson. Traffic grooming in a multi-layer network. *Opt. Netw. Mag.*, **2** (2001), 91–99.

[8] E. Modiano and P.J. Lin. Traffic grooming in WDM networks. *IEEE Communications Magazine*, **39**(7), (2001), 124–129.

[9] A. Pinlou and E. Sopena, The acircuitic directed star arboricity of subcubic graph is at most four, *Discrete Mathematics*, to appear.

[10] A. Schrijver, Combinatorial optimization. Polyhedra and efficiency. *Algorithms and Combinatorics* 24, Springer-Verlag, Berlin, 2003.

7 Appendix

7.1 Alternative proof of Theorem 14

In this subsection, we give an alternative proof of Theorem 14 using Theorem 11.

Proof. As every subcubic digraph is the subgraph of a cubic digraph, its suffices to prove the result for cubic digraph. So let D be a cubic digraph by Theorem 11, it admits a directed star 3-colouring c.

We will now select one arc e_C per bicoloured circuit C and recolour it with 4. If C is dominating (no arcs enters C) or is dominated (no arcs leaves C) let e_C be any arc of C, otherwise let e_C be an arc whose tail has indegree 2 and head outdegree 2.

It is easy to see that the set of arcs now coloured 4 is a matching. Note moreover that the arcs incident to an arc uv coloured 4 have their colours in the set of two colours $\{1, 2, 3\} \setminus \{c\}$ where c is the colour initially assigned to uv.

The way we have selected the recoloured arcs assures us that we have a 4-directed star colouring. Moreover, there is no circuits bicoloured with two colours in $\{1, 2, 3\}$. However, we may have bicoloured circuits with arcs coloured 4, so we need to do an extra recolouring. Note that the arcs coloured 4 of such circuits were not selected for dominating or dominated circuits.

An arc uv coloured 4 may be in at most 2 bicoloured circuit and if it is in two such circuits, let say C_1 bicoloured 4 and c_1 and C_2 bicoloured 4 and c_2 with $c_1 \neq c_2$, the two circuits enter u and leave v with different arcs. Now for each bicoloured circuit C, we *choose* an arc f_C coloured 4 in order to maximize the number of arcs chosen by two circuits.

Recolour each arc uv that has been chosen by two circuits, let say C_1 bicoloured 4 and c_1 and C_2 bicoloured 4 and c_2, with the colour $c_3 \in \{1, 2, 3\} \setminus \{c_1, c_2\}$. Doing so we still have a directed star colouring and the circuit C_1 and C_2 are no more bicoloured. Moreover, we do not create any new bicoloured circuit: indeed, a circuit C containing the arc uv must go through one of the two arcs vw_1 of C_1 and vw_2 of C_2, which are coloured c_1 and c_2 respectively. Wlog it uses uw_1. But w_1 has outdegree one and the arc leaving w_1 is in both C and C_1 and so is coloured 4. Hence the three colours c_3, c_i and 4 appears on C.

We will now recolour an arc of each remaining bicoloured circuit C one after another. Such a recolouring will make C no more bicoloured and create no new bicoloured circuits. Hence at the end, we will have a 4-acircuitic directed star colouring of D.

Let us consider such a circuit C. Its chosen arc uv has been chosen only once. Note that uv does not belong to any other such circuit since the number of arc chosen twice is maximized. In particular, all the arcs incident to uv have not been recoloured. Let tu be the arc preceding uv in C. Recolour tu with the colour c_3 in $\{1, 2, 3\} \setminus \{c_1, c_2\}$. This is valid since u is the head of an arc coloured 4 and thus has outdegree 2. Moreover, it does not create any bicoloured circuit since all the circuits containing tu must contain one of the arcs leaving v which are coloured using a colour of $\{c_1, c_2\}$.

□

Muticast Routing and Wavelength Assignment in Passive Optical Access Networks

B. Jaumard*, A. Houle‡
A. Shaikh§, D. Coudert†, and F. Huc†
* CIRRELT, GERAD, CIISE, Concordia University, Montréal, Canada
† Mascotte project, I3S(CNRS/UNSA)/INRIA, Sophia-Antipolis, France
‡ ECE, Université de Sherbrooke, Canada
§ ECE, Concordia University, Canada

September 9, 2008

***Abstract*— Passive optical networks (PONs) have evolved to provide much higher bandwidth in the access network. A PON is a point-to-multipoint optical network, where an optical line terminal (OLT) at the central office is connected to many optical network units (ONUs) at remote nodes through one or multiple 1:N optical splitters. The network between the OLT and the ONUs is passive, i.e., it does not require any power supply. The objective of this paper is to explore how to provision efficiently a given WDM-PON architecture with its set of splitters in order to best serve multiuser applications such as, e.g., video conference or video on demand applications. It amounts to implement efficiently multicast routing so that it is profitable both in terms of bandwidth efficiency use and network costs. We propose an efficient and scalable optimization model for tree and mesh topologies for both grooming and non grooming traffic. For a given topology and set of wavelengths and ONUs, simulation shows the expected grade of service one can expect for multicast traffic under different traffic patterns.**

Keywords: WDM Network; Network Dimensioning; Multicast; MC-RWA Problem; Column Generation; Optimal Solution.

CONTENTS

I. INTRODUCTION

The access network, also known as the first mile network or the local loop connects the service provider central offices (COs) to residential and business subscribers. The bandwidth demand in the access network has been rapidly increasing over the recent years. Residential subscribers demand is characterized by high bandwidth requirements and correspond to various media services. On the other hand, corporate users demand broadband infrastructure through which they can connect their local-area networks to the Internet backbone. While the dominant deployed broadband access solutions today are digital subscriber (DSL) (or very-high-rate DSL (VDSL)) and community antenna television (CATV) (cable

TV) based networks, both technologies have limitations as they were initially developed for carrying voice and TV signals, respectively. Moreover, VDSL encounters severe distance limitations, and CATV networks which were mainly build for delivering broadcast service, do not fit well for the bidirectional communication of a data network.

Passive optical networks (PONs) have evolved to provide much higher bandwidth in the access network. A PON is a point-to-multipoint optical network, where an optical line terminal (OLT) at the CO is connected to many optical network units (ONUs) at remote nodes through one or multiple 1:N optical splitters. The network between the OLT and the ONU is passive, i.e., it does not require any power supply. PONs use a single wavelength in each of the two directions - downstream (CO to end users) and upstream (end users to CO) - and the wavelengths are multiplexed on the same fiber through coarse WDM (CDWM). Various blends of the PONs have emerged in recent years: the ATM Passive Optical Network (APON), the first Passive optical network standard; the Ethernet PON (EPON) which is an IEEE/EFM standard for using Ethernet for packet data; the Broadband PON (BPON), a standard based on APON; and the Giga PON (GPON), an evolution of the BPON standard, which has higher rates, enhanced security, and choice of Layer 2 protocol (ATM, GEM, Ethernet).

Although the PON provides higher bandwidth than traditional copper-based access networks, there exists the need for further increasing the bandwidth of the PON by employing wavelength-division multiplexing (WDM) so that multiple wavelengths may be supported in either or both upstream and downstream directions.Such a PON is known as a WDM-PON. However, WDM-PONs networks have not yet been highly developed in North America due to immature or costly device technologies and a lack of suitable network protocols to support the architecture. Now that more mature, but still costly technologies have emerged, one the key issue is to develop those WDM-PON networks at minimum cost and maximum grade of services. Hence, in this paper, we develop optimization models for the design of WDM-PON networks where we maximize the number of granted requests while we minimize the network cost or maximize the revenue of the service provider. We consider four different models, two static ones, and two dynamic ones assuming the use of the TDM protocol, with either a tree or a light mesh topology.

The paper is organized as follows. In the next section, we present the notation and the model that we will use. In Sections III and IV, we present the different models, according to two different objectives. And finally, in Section V, we present the scenario that will be used for the experiments.

II. Generalities

A. Notations

Let us consider an optical access WDM network represented by an undirected graph $G = (V, E)$ with node set $V = \{\text{OLT}, \text{ONU}_1, \ldots, \text{ONU}_n\}$ where each node of V is associated with a node of the physical network, and with link set $L = \{\ell_1, \ell_2, \ldots, \ell_m\}$ where each link is associated with a physical fiber link of the physical network. The signal is a downstream one when it goes from the OLT to an ONU, it is an upstream one when it goes in the other direction. Let $\mathcal{O} = \{\text{ONU}_1, \ldots, \text{ONU}_n\}$ be the overall set of ONUs. Let $\omega(v)$ be the set of fiber links at node v. Given $\ell \in \omega(\text{OLT})$, we call $\mathcal{O}^\ell$ the set of ONUs that can be reached by a path starting at the OLT with link ℓ. In the first models (Sections III-A, III-B, IV-A and IV-B without time division multiplexing), the capacity of each link ℓ corresponds to the number of available wavelengths as the capacity of each request is assumed equal to the wavelength transport capacity. In the two following models (Sections III-C, III-D, IV-C and IV-D with time division multiplexing), transport capacity of the wavelength can vary from one wavelength to the next and belongs to the set of values $B = \{\text{OC}_{48}, \text{OC}_{192}\}$. Let B_ℓ be the transport capacity on fiber link ℓ. The set of available wavelengths is denoted by $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_W\}$ with $W = |\Lambda|$.

The traffic, which is supposed asymmetrical, is defined by a matrix $T = (T_{sD})$ where T_{sD} defines the number of requested connections from a source s to a given set of destination D. Let $R = \{r = (s_r, D_r) \in V \times \mathcal{P}(V) : T_{sD_r} > 0\}$ be the overall set of requests, where $\mathcal{P}(V)$ denotes the set of subsets of V. We associate a cost COST_r to each request which, e.g., can be equal to the revenue for the service provider when it grants r. The set R can be divided into two request subsets:

R^{CORE} the set of requests of the type (OLT, a subset of ONUs);

R^{ACCESS} the set of requests of the type (an ONU, a subset of ONUs).

For a request r originating at the OLT, it can be subdivided into sub-requests as follows. For each link $\ell \in \omega(\text{OLT})$, we define a sub-request r^ℓ from the OLT toward $D_r^\ell = D_r \cap \mathcal{O}^\ell$.

In passive optical access networks, only single-hop connections are used, i.e., given a request r and a path from a given source s_r to its destination or to one node of its destination node set $d \in D_r$, the path is routed on a unique wavelength. The MC-RWA problem can then be formally stated as follows: given a graph G associated with a WDM optical access network and a set of requests R, find suitable light-trees (t, λ) for each (accepted) connection where t is a tree on a subset of nodes and λ a wavelength, so that no two trees sharing an arc of G are assigned the same wavelength. We study several variants of the MC-RWA problem with the objective of minimizing the blocking rate, i.e., maximizing the number of granted connections. We consider first a basic case where the bandwidth requirement of a request is equal to the transport capacity of the wavelengths, both on a tree and a light mesh topology. We then extend those first models to dynamic traffic on the two types of topologies. Considering light mesh topologies is motivated by the need of provisioning additional bandwidth to some particular ONUs and also more importantly to add some reliability.

B. Equipment

We suppose that the network is equipped with bidirectional fibers. Also, the intermediate nodes, i.e. the nodes which are neither an ONU nor the OLT, are equipped with splitters. The

size of a splitter correspond to the number k of its output ports and may vary: a beam splitter can divide an input signal (beam light) into two or more output signals ($1:2$ or 1×2 splitter) or more (only power of 2, up to 32, i.e., 1:4, 1:8, 1:16, or 1:32 splitters exist). Through loss is the amount of attenuation the signal receives as it passes from input to output. A typical two-way splitter has a through loss of about 3 or 3.5 dB from the input to each output. Four-and eight-way splitters are also common, having typical through losses of 6-7 and 9-11 dB typically. If a $1:k$ splitter is installed at a given node with k outgoing fibers, then all signals on any wavelength, on any of these k outgoing fibers, are associated with a split signal and suffer from the same through loss. In addition, one has of course to take into account the optical attenuation, which is of the order of 0.2 to 0.25 dB/km. The topology we consider for the network is further discussed in Sections III-A and III-B.

III. MC-RWA Problem Models

The objective is to maximize the grade of service, i.e., to maximize the number of granted requests:

$$\max \sum_{r\in R} \text{COST}_r x_r.$$

A. MC-RWA on a Tree Topology

Given a network with a tree topology rooted at an OLT and where the ONUs are leaves. We have $\mathcal{O}^\ell \cap \mathcal{O}^{\ell'} = \emptyset$ for two different links ℓ and ℓ' of $\omega(\text{OLT})$ which implies that the routes are unique for every $r \in R$.

We suppose that the tree has no multiple edges, i.e., the links are single bidirectional fibers, each of them with W wavelengths. Also, as the topology of the access network is given, we suppose that it satisfies the attenuation constraints, in other words, the intensity of the signal at the ONUs is always high enough.

One decision vector x: $x_r = 1$ if request r is granted, and 0 otherwise for all $r \in R$.

$$x_r = \begin{cases} 1 & \text{if request } r \text{ is granted} \\ 0 & \text{otherwise} \end{cases}$$

Capacity constraints on the outgoing fiber links of the OLT: we must consider the requests of type (OLT, D_r), and both the downstream components and the upstream component of the requests of type (ONU_i, D_r).

$$\sum_{r\in R: D_r\cap \mathcal{O}^\ell \neq \emptyset} x_r + \sum_{r\in R^{\text{ACCESS}}: s_r \in \mathcal{O}^\ell} x_r \leq W \qquad \ell \in \omega(\text{OLT}) \quad (1)$$

Each ONU can read only one wavelength:

$$\sum_{r\in R: \text{ONU}_i \in D_r} x_r \leq 1 \qquad \text{ONU}_i \in \mathcal{O}. \quad (2)$$

Each ONU can send on only one wavelength:

$$\sum_{r\in R^{\text{ACCESS}}: s_r = \text{ONU}_i} x_r \leq 1 \qquad \text{ONU}_i \in \mathcal{O}. \quad (3)$$

We do not need to add explicitly the constraints stating that we need two different wavelengths at a given ONU, one for the downstream signal, another one for the upstream signal due to the following result.

Theorem 1: Given a solution to the MC-RWA_T problem, one can always associate wavelengths to the granted requests using a greedy algorithm.

Proof: The set of used wavelengths can be split between the wavelengths for downstream requests and those for upstream requests. On link ℓ, there are $W_\ell^D = \sum_{r\in R: s_r \in \mathcal{O}^\ell} x_r$ downstream wavelengths, and $W_\ell^U = \sum_{r\in R: D_r \cap \mathcal{O}^\ell \neq \emptyset} x_r$ upstream wavelengths. Equation (1) guarantees that we do not exceed the number of available wavelengths, i.e., W.

On link ℓ, we may assign the first W_ℓ^D wavelengths to the downstream traffic, so that the j^{th} granted core request r_j on link $\ell \in \omega(\text{OLT})$ is assigned wavelength λ_j and the remaining wavelengths to the upstream traffic. This means that if $\text{ONU}_i \in D_{r_j}$, then ONU_i receives signal on λ_j. Similarly, again on link ℓ, the j^{th} granted access request r_j on link $\ell \in \omega(\text{OLT})$ is assigned wavelength $\lambda_{j+W_\ell^D}$, and $\text{ONU}_{s_{r_j}}$ sends signal on $\lambda_{j+W_\ell^D}$. Equations (2) and (3) ensure that each ONU receives signal on at most one wavelength for downstream traffic and sends signal on at most one wavelength for upstream traffic. ■

B. MC-RWA on a Light Mesh Topology

Let us now consider the MC-RWA_M problem corresponding to the MC-RWA problem on a WDM-PON light mesh topology. Such a topology may be of interest in order to increase the grade of service on the one hand, and in order to provide some sort of resiliency in case of link failure. However, at least for nowadays, it would be too costly to design a WDM-PON network which would be protected against all link failures. Therefore, we remain with the assumptions that each ONU is equipped with a single transmitter and a single receiver, meaning that it can read and send on a single wavelength.

We suppose given a mesh network which is obtained from a tree rooted at an OLT of which some leaves have been merged and where the ONUs are leaves or merged leaves. Since the devices in PON are passive and that the objective is to get low cost access network the routing has to be simple. We suppose that a message sent by the OLT on some link always go downstream, i.e., once arrived at an ONU it does not leave it to go to an other ONU.

The main difference between a tree network and a mesh network is that we do not have any more $\mathcal{O}^\ell \cap \mathcal{O}^{\ell'} = \emptyset$ for two different links ℓ and ℓ' of $\omega(\text{OLT})$. It means that there may have several routes from the OLT to the ONU, however we have few of them (at most one per link $l \in \omega(\text{OLT})$).

We also suppose that the links of the network are single bidirectional fiber, each of them with W wavelength.

1) Variables: In order to set the MC-RWA_M model, we use a first decision vector $x = (x_r)_{r\in R}$ such that $x_r = 1$ if request r is granted, and 0 otherwise. We also also two sets of decision vectors in order to check whether a given link belongs

to the route of a given request, and if it is the case, whether the route traverses the link in the upstream or the downstream direction. Let $x_\ell^D = (x_{r\ell}^D)_{r\in R}$ and $x_\ell^D = (x_{r\ell}^U)_{r\in R}$, for each link $\ell \in \omega(\text{OLT})$ be such that, for a given ℓ, $x_{r\ell}^D$ (resp. $x_{r\ell}^U$) is equal to 1 if request r is routed through ℓ in the downstream (resp. upstream) direction, and 0 otherwise.

$$x_{r\ell}^D = \begin{cases} 1 & \text{if request } r \text{ is routed through } \ell \text{ downstream} \\ 0 & \text{otherwise} \end{cases}$$

$$x_{r\ell}^U = \begin{cases} 1 & \text{if request } r \text{ is routed through } \ell \text{ upstream} \\ 0 & \text{otherwise} \end{cases}$$

One other vectors with binary variables, such that for each request r, and ech ONU_i, source or destination, each component x_{ri}^ℓ is used in order to check whether ONU_i^ℓ, with respect to request r on link ℓ, is served or not. If $x_{ri}^\ell = 1$, it means that ONU_i receives request r on link ℓ or send it if $\text{ONU}_i = s_r$.

$$x_{ri}^\ell = \begin{cases} 1 & \text{if request } r \text{ is satisfied for } \text{ONU}_i \text{ via } \ell. \\ 0 & \text{otherwise} \end{cases}$$

2) Constraints: Capacity constraints on the outgoing fiber links of the OLT: we must consider the requests of type (OLT, D_r), and both the downstream components and the upstream components of the requests of type (ONU_i, D_r):

$$\sum_{r\in R: D_r \cap O^\ell \neq \emptyset} x_{r\ell}^D + \sum_{r\in R^{\text{ACCESS}}: s_r \in \mathcal{O}^\ell} x_{r\ell}^U \leq W \qquad \ell \in \omega(\text{OLT}).$$

Each destination ONU has to receive the request if the request is accepted. Conversely, if none of the $\text{ONU}_i \in D_r$ is served, then the request r is denied.

$$x_r \leq \sum_{\ell: \text{ONU}_i \in \mathcal{O}^\ell \cap D_r} x_{ri}^\ell \qquad r \in R, \text{ONU}_i \in D_r.$$

The requests have to be sent:

$$x_r \leq \sum_{\ell: s_r \in \mathcal{O}^\ell} x_{rs_r}^\ell \qquad r \in R^{\text{ACCESS}}. \tag{4}$$

Each ONU can read only one wavelength at a time on all incoming fibers:

$$\sum_{r\in R: D_r \cap \text{ONU}_i \neq \emptyset} \sum_{\ell \in \omega(\text{OLT}): \text{ONU}_i \in \mathcal{O}^\ell} x_{ri}^\ell \leq 1 \qquad \text{ONU}_i \in O. \tag{5}$$

Each ONU can send on only one wavelength on all potential outgoing fibers:

$$\sum_{r\in R: s_r = \text{ONU}_i} \sum_{\ell \in \omega(\text{OLT}): \text{ONU}_i \in \mathcal{O}^\ell} x_{rs_r}^\ell \leq 1 \qquad \text{ONU}_i \in O. \tag{6}$$

$$x_{ri}^\ell \leq x_{r\ell}^D \qquad r \in R, \text{ONU}_i \in D_r, l : \text{ONU}_i \in O^\ell \tag{7}$$

$$x_{rs_r}^\ell \leq x_{r\ell}^U \qquad r \in R^{ACCESS}, l : s_r \in O^\ell \tag{8}$$

Again, wavelengths are not formally associated with each granted request as Theorem 1 still applies.

Theorem 2: We can always assign wavelengths

C. MC-GRWA *on a Tree Topology*

1) Parameters: Let $\mathcal{C}$ be the overall set of configurations. A configuration $C \in \mathcal{C}$ corresponds to a set of granted components (either the uplink component of a request, or one of its downlink component, i.e., a subset of served ONUs) of requests assigned on the same wavelength: the master problem will make sure that all components of a request can be provisioned in order to grant the (whole) request.

$$C_r^U = \begin{cases} 1 & \text{if the uplink path component of } r \in R^{\text{ACCESS}} \\ & \text{can be provisioned in configuration } C \\ 0 & \text{otherwise} \end{cases}$$
$$\text{for } r \in R^{\text{ACCESS}},$$
$$C_{ri} = \begin{cases} 1 & \text{if } \text{ONU}_i \text{ is served with respect to request } r \\ & \text{in configuration } C \\ 0 & \text{otherwise} \end{cases}$$
$$\text{for } \text{ONU}_i \in D_r, r \in R, \tag{9}$$

i.e., $C \in \{0,1\}^{|R^{\text{ACCESS}}| + \sum_{r\in R} |D_r|}$.

2) Variables: We use a variable x^C, $C \in \mathcal{C}$, to indicate if a configuration is selected for some request provisioning on a given wavelength λ.

$$x^C = \begin{cases} 1 & \text{if } C \text{ is selected} \\ 0 & \text{otherwise} \end{cases} \qquad C \in \mathcal{C}.$$

We also use two other sets of variables $(x_r)_{r\in R}$ and $(x_{ri})_{\text{ONU}_i \in D_r, r\in R}$ defined as follows:
For $r \in R$,

$$x_r = \begin{cases} 1 & \text{if request } r \text{ is granted,} \\ 0 & \text{otherwise, i.e., request } r \text{ is denied.} \end{cases}$$

For $r \in R$ and $\text{ONU}_i \in D_r$,

$$x_{ri} = \begin{cases} 1 & \text{if } \text{ONU}_i \text{ is served with respect to request } r \\ & \text{assuming request } r \text{ is granted,} \\ 0 & \text{otherwise, which implies that} \\ & \text{request } r \text{ cannot then be granted.} \end{cases}$$

3) Master Problem:

a) Objective: $\max \sum_{r\in R} \text{COST}_r \, x_r$.

b) Constraints: No more configurations than the number of wavelengths, as each configuration is associated with a wavelength:

$$\sum_{C\in\mathcal{C}} x^C \leq W. \tag{10}$$

Each ONU can read at most one wavelength:

$$\sum_{C\in\mathcal{C}} \left(\max_{r\in R} C_{ri} \right) x^C \leq 1 \qquad \text{ONU}_i \in \mathcal{O}. \tag{11}$$

where $\max_{r\in R} C_{ri} = 1$ if at least one request r is granted in order to serve ONU_i in configuration C, and 0 otherwise.

Each ONU can send its requests on at most one wavelength:

$$\sum_{C\in\mathcal{C}} \left(\max_{r\in R^{\text{ACCESS}}:s_r=\text{ONU}_i} C_r^U \right) x^C \leq 1 \qquad \text{ONU}_i \in \mathcal{O} \quad (12)$$

where $\max_{r\in R^{\text{ACCESS}}} C_r^U = 1$ if at least one request $r \in R^{\text{ACCESS}}$ such that $s_r = \text{ONU}_i$ is granted in configuration C and therefore uses the signal emitted at ONU_i, and 0 otherwise.

In order for a request to be granted, all of its ONU must be served, possibly with a different configuration (i.e., wavelength):

$$\sum_{C\in\mathcal{C}} C_{ri}\, x^C \geq x_{ri} \qquad \text{ONU}_i \in D_r, r \in R \quad (13)$$

$$x_r \leq x_{ri} \qquad \text{ONU}_i \in D_r, r \in R \quad (14)$$

$$\sum_{C\in\mathcal{C}} C_r^U\, x^C \geq x_r \qquad r \in R^{\text{ACCESS}}. \quad (15)$$

Let us first consider a request $r \in R^{\text{CORE}}$. Request r can be granted if and only if all its destinations can be served, i.e., $x_r = 1$ if and only if $x_{ir} = 1$ for all $\text{ONU}_i \in D_r$. Note that if at least one $\text{ONU}_i \in D_r$ is not served ($x_{ir} = 0$), then, using (14) r cannot be granted (i.e., $x_r = 0$). In addition, $x_{ri} = 1$ only if there is at least one selected configuration C (i.e., such that $x^C = 1$) such that $C_{ir} = 1$, see equation (13). On the other hand, if, for a given r, all x_{ri} for $i : \text{ONU}_i \in D_r$ are equal to 1, x_r will be set to 1 due to the fact that the objective is $\max \sum_{r\in R} \text{COST}_r x_r$.

Let us next consider a request $r \in R^{\text{ACCESS}}$. Request r can be granted if and only if *(i)* all its destinations can be served and *(ii)* its uplink component can be provisioned, i.e., $x_r = 1$ if and only if *(i)* $x_{ir} = 1$ for all $\text{ONU}_i \in D_r$ and *(ii)* there is at least one selected configuration C (i.e., such that $x^C = 1$) such that $C_r^U = 1$. Note that if either one $\text{ONU}_i \in D_r$ is not served ($x_{ir} = 0$) or the uplink component cannot be provisioned (i.e., all selected configurations C are such that $C_r^U = 0$), using (14) or (15), r cannot be granted (i.e., $x_r = 0$). On the other hand, if, for a given r, all x_{ri} for $i : \text{ONU}_i \in D_r$ are equal to 1 and there is at least one selected configuration C such that $C_r^U = 1$, x_r will be set to 1 due to the fact that the objective is $\max \sum_{r\in R} \text{COST}_r x_r$.

4) Pricing Problem:

a) Objective: Each pricing problem corresponds to the generation of a configuration. The objective is to maximize the reduced cost, i.e., $\overline{c}^C$, or $\overline{c}$ for short, where

$$\overline{c} = -u_0 - \sum_{i:\text{ONU}_i\in\mathcal{O}} u_i^D \left(\max_{r\in R} C_{ri} \right) - \sum_{i:\text{ONU}_i\in\mathcal{O}} u_i^U \left(\max_{r\in R^{\text{ACCESS}}:s_r=\text{ONU}_i} C_r^U \right) + \sum_{r\in R} \sum_{i:\text{ONU}_i\in D_r} v_{ri}^D C_{ri} + \sum_{r\in R^{\text{ACCESS}}} v_r^U C_r^U$$

where u_0 is the dual variable associated with constraint (10), $u_i^D \geq 0$ with (11), $u_i^U \geq 0$ with (12), $v_{ri}^D \leq 0$ with (13), and $v_r^U \leq 0$ with (15).

In order to eliminate the nonlinearities, let us introduce the two binary variables C_i^D and C_i^U. In order to have:

$$C_i^D = \max_{r\in R} C_{ri} \quad (16)$$

$$C_i^U = \max_{r\in R^{\text{ACCESS}}:s_r=\text{ONU}_i} C_r^U \quad (17)$$

we add the constraints :

$$C_i^D \geq C_{ri} \qquad r \in R, \text{ONU}_i \in D_r \quad (18)$$

$$C_i^U \geq C_r^U \qquad r \in R^{\text{ACCESS}}, i : \text{ONU}_i = s_r, \quad (19)$$

$$C_i^D \leq \sum_{r\in R} C_{ri} \qquad i : \text{ONU}_i \in D_r, r \in R \quad (20)$$

$$C_i^U \leq \sum_{r\in R^{\text{ACCESS}}:\text{ONU}_i=s_r} C_r^U \qquad r \in R^{\text{ACCESS}}, i : \text{ONU}_i = s_r. \quad (21)$$

for $i : \text{ONU}_i \in \mathcal{O}$. Observe that $C_i^D = 1$ if and only if ONU_i is served for at least one request, and 0 otherwise, and that $C_i^U = 1$ if and only if the path $(\text{ONU}_i, \text{OLT})$ is provisioned for at least one request originating at ONU_i, and 0 otherwise.

The objective of the pricing problem can then be rewritten:

$$\overline{c} = -u_0 - \sum_{i:\text{ONU}_i\in\mathcal{O}} u_i^D C_i^D - \sum_{i:\text{ONU}_i\in\mathcal{O}} u_i^U C_i^U + \sum_{r\in R} \sum_{i:\text{ONU}_i\in D_r} v_{ri}^D C_{ri} + \sum_{r\in R^{\text{ACCESS}}} v_r^U C_r^U.$$

b) Variables: We use two decision vectors y^D and y^U such that each component y_ℓ^D (resp. y_ℓ^U) indicates whether the wavelength associated with the configuration is used downstream (resp. upstream) or not: y_ℓ^D gets value 1 if it used downstream, 0 otherwise (i.e., it is either unused or used upstream); y_ℓ^D gets value 1 if it used upstream, 0 otherwise (i.e., it is either unused or used downstream).

$$y_\ell^U = \begin{cases} 1 & \text{if the wavelength associated with the configuration} \\ & \text{is used downstream on } \ell \\ 0 & \text{otherwise} \end{cases}$$

$$y_\ell^D = \begin{cases} 1 & \text{if the wavelength associated with the configuration} \\ & \text{is used downstream on } \ell \\ 0 & \text{otherwise} \end{cases}$$

Decision vector C with components defined as in III-C.

$$C_r^U = \begin{cases} 1 & \text{if the uplink path component of } r \in R^{\text{ACCESS}} \\ & \text{can be provisioned in configuration } C \\ 0 & \text{otherwise} \end{cases}$$

for $r \in R^{\text{ACCESS}}$,

$$C_{ri} = \begin{cases} 1 & \text{if } \text{ONU}_i \text{ is served with respect to request } r \\ & \text{in configuration } C \\ 0 & \text{otherwise} \end{cases}$$

for $\text{ONU}_i \in D_r, r \in R$.

c) Constraints: A given link ℓ cannot be used simultaneously used upstream and downstream:

$$y_\ell^D + y_\ell^U \leq 1 \qquad \ell \in \omega(\text{OLT}). \tag{22}$$

Capacity constraints on each link of the OLT when it is used downstream:

$$\sum_{r \in R:\ D_r \cap \mathcal{O}^\ell \neq \emptyset} b_r \left(\max_{i:\text{ONU}_i \in D_r \cap O^\ell} C_{ri} \right) \leq B_\ell\, y_\ell^D \qquad \ell \in \omega(\text{OLT}). \tag{23}$$

Capacity constraints on each link of the OLT when it is used upstream:

$$\sum_{r \in R^{\text{ACCESS}}: s_r \in \mathcal{O}^\ell} b_r C_r^U \leq B_\ell\, y_\ell^U \qquad \ell \in \omega(\text{OLT}).$$

In order to linearize constraint (23), we define $C_{r\ell}^D$ for the core requests:

$$C_{r\ell}^D = \begin{cases} 1 & \text{if } r \in R^{\text{CORE}} \text{ is granted in} \\ & \text{configuration } C, \text{ which entails} \\ & \text{the servicing of all } \text{ONU}_i \in D_r, \\ 0 & \text{otherwise} \end{cases}$$
for $r \in R^{\text{CORE}}$.

To linearize constraint (23), we want $C_{r\ell}^D = \max_{i:\text{ONU}_i \in D_r \cap \mathcal{O}^\ell} C_{ri}$. This is done by adding the following equations:

$$\sum_{r \in R:\ D_r \cap \mathcal{O}^\ell \neq \emptyset} b_r C_{r\ell}^D \leq B_\ell\, y_\ell^D \qquad \ell \in \omega(\text{OLT}) \tag{24}$$
$$C_{r\ell}^D \geq C_{ri} \qquad i : \text{ONU}_i \in D_r, r \in R^{\text{CORE}} \tag{25}$$
$$C_{r\ell}^D \leq \sum_{i:\text{ONU}_i \in D_r} C_{ri} \qquad r \in R^{\text{CORE}}. \tag{26}$$

d) Recapitulation: Pricing Problem:

$$\max \overline{c}$$

where

$$\overline{c} = -u_0 - \sum_{i:\text{ONU}_i \in \mathcal{O}} u_i^D C_i^D - \sum_{i:\text{ONU}_i \in \mathcal{O}} u_i^U C_i^U + \sum_{r \in R} \sum_{i:\text{ONU}_i \in D_r} v_{ri}^D C_{ri} + \sum_{r \in R^{\text{ACCESS}}} v_r^U C_r^U$$

subject to the following set of constraints:

$$C_i^D \geq C_{ri} \qquad r \in R, \text{ONU}_i \in D_r \tag{18}$$
$$C_i^U \geq C_r^U \qquad r \in R^{\text{ACCESS}}, \text{ONU}_i = s_r \tag{19}$$
$$y_\ell^D + y_\ell^U \leq 1 \qquad \ell \in \omega(\text{OLT}) \tag{22}$$
$$\sum_{r \in R^{\text{ACCESS}}: s_r \in \mathcal{O}^\ell} b_r C_r^U \leq B_\ell\, y_\ell^U \qquad \ell \in \omega(\text{OLT}) \tag{27}$$
$$\sum_{r \in R:\ D_r \cap \mathcal{O}^\ell \neq \emptyset} b_r C_{r\ell}^D \leq B_\ell\, y_\ell^D \qquad \ell \in \omega(\text{OLT}) \tag{24}$$
$$C_{r\ell}^D \geq C_{ri} \qquad i : \text{ONU}_i \in D_r, r \in R^{\text{CORE}} \tag{25}$$
$$C_{r\ell}^D \leq \sum_{i:\text{ONU}_i \in D_r} C_{ri} \qquad r \in R^{\text{CORE}} \tag{26}$$
$$y_\ell^D, y_\ell^U \in \{0,1\} \qquad \ell \in \omega(\text{OLT})$$
$$C_r^D, C_r^U \in \{0,1\} \qquad r \in R$$
$$C_i^D, C_i^U \in \{0,1\} \qquad \text{ONU}_i \in \mathcal{O}$$
$$C_{ri} \in \{0,1\} \qquad i : \text{ONU}_i \in D_r, r \in R.$$

D. MC-GRWA *on a Light Mesh Topology*

We make the same assumptions as in Section III-B. Unlike the tree network topology, the light mesh topology has some ONUs with more than one paths. In the light mesh network topology, ONUs with multiple paths are considered in the pricing problem. Consequently, there is no change in the master problem and we use the one described in Section III-C. For detail of variables, parameters, objective and constraints, refer to this section.

1) Parameters,Variables,Master Problem: Same as for the tree topology.

2) Pricing Problem:

a) Objective: Same as for the tree topology.

b) Variables: We need two additional sets of variables $(C_{ri}^\ell)_{ri\ell}$ and $(C_r^\ell)_{r\ell}$ defined as follows:

$$C_{ri}^\ell = \begin{cases} 1 & \text{ONU}_i \text{ is served with respect to request } r \\ & \text{using the resource of link } \ell, \\ 0 & \text{otherwise} \end{cases}$$

$$C_r^\ell = \begin{cases} 1 & \text{uplink component of (granted) request } r \\ & \text{uses resource of link } \ell, \\ 0 & \text{otherwise.} \end{cases}$$

c) Constraints: The constraints are those of the problem on the tree topology slightly modified to adapt to the mesh network.

$$C_i^D \geq C_{ri} \qquad \text{ONU}_i \in D_r, r \in R \tag{28}$$
$$C_{s_r}^U \geq C_r^U \qquad r \in R^{\text{ACCESS}} \tag{29}$$
$$C_i^D \leq \sum_{r \in R} C_{ri} \qquad \text{ONU}_i \in O \tag{30}$$
$$C_i^U \leq \sum_{r \in R^{\text{ACCESS}}: \text{ONU}_i = s_r} C_r^U \qquad \text{ONU}_i \in O \tag{31}$$

The four previous equation are here to linearise the equations, they are the same as equations (18-21).

The following equations are specific to the mesh topology. They express the possibility to serve an ONU using different fibers ℓ.

$$C^{\ell}_{ri} \leq C_{ri} \qquad \text{ONU}_i \in D_r, \ell : \text{ONU}_i \in O^{\ell}, r \in R \tag{32}$$

$$C^{\ell}_{r} \leq C^{U}_{r} \qquad r \in R^{\text{ACCESS}}, \ell : s_r \in O^{\ell} \tag{33}$$

$$C_{ri} \leq \sum_{\ell:\text{ONU}_i \in O^{\ell}} C^{\ell}_{ri} \qquad \text{ONU}_i \in D_r, r \in R \tag{34}$$

$$C^{U}_{r} \leq \sum_{\ell:s_r \in O^{\ell}} C^{\ell}_{r} \qquad r \in R^{\text{ACCESS}} \tag{35}$$

Then we have equations to verify that no fibers and no wavelength are over-used :

$$y^{D}_{\ell} + y^{U}_{\ell} \leq 1 \qquad \ell \in \omega(\text{OLT}) \tag{36}$$

$$\sum_{r \in R} b_r C^{\ell}_{ri} \leq B_{\ell} y^{D}_{\ell} \qquad \ell \in \omega(\text{OLT}) \tag{37}$$

$$\sum_{r \in R^{\text{ACCESS}}:s_r \in O^{\ell}} b_r C^{\ell}_{r} \leq B_{\ell} y^{U}_{\ell} \, \ell \in \omega(\text{OLT}) \tag{38}$$

Finally, the following equations allow to determine which ONU receives which request in the configuration we are computing.

$$C^{l}_{ri} \leq C^{D}_{r\ell} \quad r \in R, \text{ONU}_i \in D_r, l : \text{ONU}_i \in O^{l} \tag{39}$$

$$C^{l}_{r} \leq C^{U}_{r\ell} \quad r \in R, l : s_r \in O^{l} \tag{40}$$

$$\sum_{\ell:\text{ONU}_i \in O^{\ell}} C^{\ell}_{ri} \leq 1 \quad \text{ONU}_i \in D_r, r \in R \tag{41}$$

$$\sum_{\ell:s_r \in O^{\ell}} C^{\ell}_{r} \leq 1 \qquad r \in R^{\text{ACCESS}} \tag{42}$$

IV. Maximizing the Number of Served ONUs

In this section, we define the mathematical models for the tree and the light mesh topologies with/without traffic-grooming, where the objective is to maximize the weighted number of served ONUs subject to the available resources in the networks. A partially granted request is defined as a request that is granted to serve a few (at least one) destination ONUs. In this section, while maximizing the number of served ONUs we may grant partially requests and/or whole requests. In a weighted version, we can associate a cost to each served ONU that would represent weight of ONU.

A. MC-RWA *on* WDM-PON *Tree Topology*

A new optimization model for the tree network without traffic-grooming, with the objective of maximizing the weighted number of served ONUs, is proposed as follows.

1) Objective: We need the variables x_{ri}, which for each request r and each ONU, destination or source of r, indicates if r is satisfied for this ONU.

$$x_{ri} = \begin{cases} 1 & \text{if request } r \text{ is satisfied for ONU}_i \\ 0 & \text{otherwise} \end{cases}$$

The objective is to maximize the number of served ONUs:

$$\max \sum_{r \in R} \sum_{i:\text{ONU}_i \in D_r} \text{COST}_{r_i} x_{ri}.$$

where COST_{r_i} is the cost associated with ONU_i with respect to request r.

2) Constraints: We use the same variables as in previous formulation, exept x_r which is replaced by x^{ℓ}_r. It indicates if request r is sent on fiber ℓ or not.

$$x_r = \begin{cases} 1 & \text{if request } r \text{ is granted} \\ 0 & \text{otherwise} \end{cases}$$

Capacity constraints on the fiber link of the OLT: Upstream & downstream wavelengths of the ONUs can't be exceeds from available wavelength on each associated link. x^{ℓ}_r gets value 1 if request r is served on link ℓ, 0 otherwise.

$$x^{\ell}_r = \begin{cases} 1 & \text{if request } r \text{ is granted on link } \ell \\ 0 & \text{otherwise} \end{cases}$$

$$\sum_{r \in R: D_r \cap \mathcal{O}^{\ell} \neq \emptyset} x^{\ell}_r + \sum_{r \in R^{\text{ACCESS}}:s_r \in \mathcal{O}^{\ell}} x^{\ell}_r \leq W \quad \ell \in \omega(\text{OLT}). \tag{43}$$

Each ONU can read only one wavelength:

$$\sum_{r \in R:\text{ONU}_i \in D_r} x_{ri} \leq 1 \qquad \text{ONU}_i \in \mathcal{O}. \tag{44}$$

Each ONU can send only one wavelength:

$$\sum_{r \in R^{\text{ACCESS}}:\text{ONU}_i = s_r} x_{ri} \leq 1 \qquad \text{ONU}_i \in \mathcal{O}. \tag{45}$$

In order to grant the request r on link ℓ, at least one ONU in the destination of request r in O^{ℓ} must be accepted:

$$x^{\ell}_r \leq \sum_{i:\text{ONU}_i \in D_r \cap O^{\ell}} x_{ri} \qquad r \in R, l \in \omega(\text{OLT}). \tag{46}$$

In order to grant the request r, source ONU (if it is not OLT) of request r must be accepted:

$$x^{\ell}_r \leq x_{rs_r} \qquad r \in R^{\text{ACCESS}}, l \in \omega(\text{OLT}). \tag{47}$$

The request r must be granted in order to grant the destination ONU(s) : If none of the ONU_i is served, the x_r denied

$$x^{\ell}_r \geq x_{ri} \qquad \ell \in \omega(\text{OLT}), \text{ONU}_i \in D_r \cap O^{\ell}. \tag{48}$$

B. MC-RWA *on* WDM-PON *Light Mesh Topology*

This section describes the MC-RWA (without traffic-grooming) problem on WDM-PON light mesh network topology with the objective of maximizing the weighted number of served ONUs.

1) Objective: Maximize the number of served ONUs:

$$\max \sum_{r \in R} \sum_{i:\text{ONU}_i \in D_r} \sum_{\ell:\text{ONU}_i \in O^{\ell}} \text{COST}_{ri} x^{\ell}_{ri}$$

2) Constraints: Concerning the capacity of the optical fibers, it has a limited number of wavelengths and we can not exceed this number of available wavelengths on each link. Note that, as before, wavelengths are bidirectional, so they can be use either for downstream or upstream direction and that we assume that we are free to use the wavelengths downstream and upstream. The following set of constraints assures that wavelengths used in the downstream and upstream direction do no exceed the total number of available wavelength on each link outgoing the OLT. First recall the variables that are used:

$$x^D_{r\ell} = \begin{cases} 1 & \text{if request } r \text{ is routed through } \ell \text{ downstream} \\ 0 & \text{otherwise} \end{cases}$$

$$x^U_{r\ell} = \begin{cases} 1 & \text{if request } r \text{ is routed through } \ell \text{ upstream} \\ 0 & \text{otherwise} \end{cases}$$

$$x^\ell_{ri} = \begin{cases} 1 & \text{if request } r \text{ is satisfied for ONU}_i \text{ via } \ell. \\ 0 & \text{otherwise} \end{cases}$$

$$x_r = \begin{cases} 1 & \text{if request } r \text{ is granted} \\ 0 & \text{otherwise} \end{cases}$$

$$\sum_{r\in R: D_r \cap \mathcal{O}^\ell \neq \emptyset} x^D_{r\ell} + \sum_{r\in R^{\text{ACCESS}}: s_r \in \mathcal{O}^\ell} x^U_{r\ell} \leq W \qquad \ell \in \omega(\text{OLT}). \quad (49)$$

Each ONU can read only one wavelength:

$$\sum_{r\in R: \text{ONU}_i \in D_r} \sum_{\ell: \text{ONU}_i \in O^\ell} x^\ell_{ri} \leq 1 \qquad \text{ONU}_i \in \mathcal{O}. \quad (50)$$

Each ONU can send only on one wavelength:

$$\sum_{r\in R^{\text{ACCESS}}} \sum_{\ell: s_r \in O^\ell} x^\ell_{rs_r} \leq 1 \qquad s_r \in \mathcal{O}. \quad (51)$$

In order to grant the request r, at least one ONU in the destination of request r must be accepted:

$$x_r \geq \sum_{\ell: \text{ONU}_i \in \mathcal{O}^\ell} x^\ell_{ri} \qquad \text{ONU}_i \in D_r, r \in R. \quad (52)$$

In order to grant the request r, Source ONU (if it is not OLT) of request r must be accepted:

$$x_r \leq \sum_{\ell: s_r \in \mathcal{O}^\ell} x^\ell_{rs_r} \qquad r \in R^{\text{ACCESS}}. \quad (53)$$

If none of the ONU_i is served, the x_r denied:

$$x_r \leq \sum_{i: \text{ONU}_i \in D_r} \sum_{\ell: \text{ONU}_i \in \mathcal{O}^\ell} x^\ell_{ri} \qquad r \in R. \quad (54)$$

If link ℓ accepts the request r for downstream, then all the destination ONU(s) of r on ℓ can be accepted:

$$x^D_{r\ell} \leq \sum_{i: \text{ONU}_i \in \mathcal{O}^\ell} x^\ell_{ri} \qquad r \in R, \text{ONU}_i \in \mathcal{D}_r \cap \mathcal{O}^\ell. \quad (55)$$

If link ℓ accepts the request r for upstream, then source ONU of r on ℓ can be accepted:

$$x^U_{r\ell} \leq x^\ell_{rs_r} \qquad r \in R^{\text{ACCESS}}, \ell : s_r \in O^\ell. \quad (56)$$

If link ℓ accepts the request r for downstream, then destination ONU_i of r on ℓ can be accepted:

$$x^D_{r\ell} \geq x^\ell_{ri} \qquad r \in R, \ell : \text{ONU}_i \in D_r \cap O^\ell. \quad (57)$$

C. MC-GRWA *on a Tree Topology*

1) Master Problem: The master problem is quite similar to the one we discussed in Section III-D, except that, now, our objective is to maximize the weighted number of destination ONUs. In this problem, the parameter vector has as components C_{ri} and C_r for $\text{ONU}_i \in D_r, r \in R$ and the variable vectors are x_C, x_{ri} as defined in Section III-C and IV-A.

$$x^C = \begin{cases} 1 & \text{if } C \text{ is selected} \\ 0 & \text{otherwise} \end{cases} \qquad C \in \mathcal{C}.$$

$$x_{ri} = \begin{cases} 1 & \text{if request } r \text{ is satisfied for ONU}_i \\ 0 & \text{otherwise} \end{cases}$$

Note that, in this problem we do not use the vector with components x_r, instead, we use a new vector whose components are x_{s_r}, defined as follows:

$$x_{s_r} = \begin{cases} 1 & \text{if source } s_r = \text{ONU}_i \text{ of a request } r \text{ is served in} \\ & \text{the upstream direction, for } r \in R^{\text{ACCESS}} \\ 0 & \text{otherwise} \end{cases}$$

a) Objective:

$$\max \sum_{r\in\mathcal{R}} \text{COST}_{ri} x_{ri}.$$

b) Constraints: We have the same capacity constraints as in III-C, Equations (10-12).

In order to serve destination ONU_i with respect to request r in the downstream direction,one of its associated configuration C must be granted, this is Equation 13. Equation 13 deals with the upstream direction.

We also need the two equations:

$$x_{ri} \leq x_{s_r} \qquad \text{ONU}_i \in \mathcal{D}_r, r \in R^{\text{ACCESS}} \quad (58)$$

$$x_{s_r} \leq \sum_{\text{ONU}_i \in \mathcal{D}_r} x_{ri} r \in R^{\text{ACCESS}} \quad (59)$$

2) Pricing Problem: The pricing problem is same as the pricing problem for the tree when the objective is to maximize the weighted number of granted requests in Section III-C. As mentioned previously, the number of granted requests/ONUs are considered in the master problem and the configuration is generated by the pricing problem, so that there is no change in the pricing problem while generating the configuration C. Consequently, the pricing problem has same reduced cost objective as well as constrains and descriptions of their parameters, variables.

D. MC-GRWA on a Mesh Topology

The light mesh traffic-grooming model, like other traffic-grooming models, is divided into two subproblems called master problem and pricing problem, discussed in the following sections. The characterisitic of a configuration C are those described in Section III-D.

1) Master Problem: The master problem is the same than the master problem for the tree network topology, described in Section IV-C, with the same objective. The difference is that in a light mesh network, a few or all ONUs have multiple paths to the OLT. This multiple paths constraints are considered in the pricing problem.

2) Pricing Problem: Similarly, the pricing problem is the same as the pricing problem for the light mesh network topology with the objective of maximizing the weighted number of granted requests, presented in Section III-D. As mentioned previously, the pricing problem provide a configuration to the master problem, which selects them. There is no change in the pricing problem.

V. Solutions of the MC-RWA Models

A. An Overview on Column Generation Modeling

Column generation techniques offer solution methods for linear programs with a very large number of variables (e.g., exponential) where constraints can be expressed implicitly. They rely on a decomposition of the initial linear program into the *master problem* and the *pricing problem*. The master problem corresponds to a linear program associated with a restricted constraint matrix, with respect to the number of variables (or columns) of the initial constraint matrix. The pricing problem is defined by the optimization of the so-called reduced cost (refer to [1] if not familiar with linear programing) subject to the implicit constraints expressed by the coefficients of the constraint matrix of the master problem.

The column generation solution scheme is similar to that of the simplex algorithm: it is an iterative process where, at each step, we attempt to add one or more columns to the constraint matrix of the master problem in order to improve the value of its objective function. The search for such columns is made through the solution of the pricing problem. If its outcome corresponds to one or more columns with a negative reduced cost (assuming we deal with minimization), then it entails an improvement of the value of the master objective function; otherwise, if no solution of the pricing problem can be identified with a negative cost, we then conclude that the current solution is indeed optimal.

Column generation can be combined with branch-and-bound techniques for solving integer linear programs with a large number of variables, see [2] for a nice overview. Branching rules have to be devised properly in order to avoid generating a huge number of subproblems in the search tree associated with the branch-and-bound, either by branching on the variables of the master problem using cuts, or by branching on the variables of the pricing problem using classical branching schemes or cuts.

B. Solution of the MC-RWA and MC-GRWA Models

First we solve the linear relaxation using a column generation method where the master problem is solved using the LP CPLEX package, and where the pricing problems are solved using the ILP CPLEX package. Once we have obtained the optimal solution of the LP relaxation, we use the ILP CPLEX package again in order to get an ILP solution for the master problem. This does not lead necessarily to an optimal solution of the corresponding MC-RWA and MC-GRWA model. However, bounds can be obtained on the quality of the ILP solution obtained this way. Moreover, in order to speed the process of getting an ILP solution for the master problems, we also use a rounding off procedure that works as follows.

Objective 2: Maximizing the number of served ONUs.
Consider the configuration which contributes the most to satisfying ONUs and such that x_C^λ is fractional.
Round off x_C^λ to 1.
Re-optimize the LP relaxation assuming $x_C^\lambda = 1$.
Reiterate.

VI. Computational Results

We plan simulations for both the MC-RWA and the MC-GRWA problems on tree and mesh topologies which are described in next section. These experiments will be done under two different scenarios as described in the last section.

1) Network Instances: We consider a network with a single OLT and eight outgoing bidirectional fibers. For each fiber ℓ, there is 32 ONUs (leaf nodes independently of the tree or mesh structure). Each fiber have height or sixteen wavelengths.

2) Traffic Instances: In the first scenario, we simply consider OLT-ONUs requests while in the second one we consider both OLT-ONUs and ONU-ONUs requests. When grooming is not considered, we assume that each request use 100% of the capacity of a wavelength, otherwise it is specified according to the type of the request. Here is the description of the two scenarios we consider.

a) First scenario.: There is three types of requests: requests corresponding to customers viewing video or TV. They represent 60% of the requests and each request use between 50 and 90% of the bandwidth B available in a wavelength. These requests are multicast requests with 1 to 5 destinations. The second type of requests corresponds to radio transmission. We suppose that it represent 20% of the traffic and that each request use from 20 to 50% of the bandwith of a wavelength. The third request corresponds to users surfing on the net. It represents 20% of the requests and they use only 10% of the available bandwith.

b) Second scenario.: There is a fourth type of requests: It correspond to neighbors playing together on-line games. It is represented by a multicast originating at an ONU with destination set of size from 4 to 10. The request use from 30 to 60% of the bandwith of a wavelength. This type of request repesent 50% of the requests in the second scenario. In this scenario, the proportion of the three previous types of requests are respectively 30%, 10% and 10%.

A. Evaluation Parameters

We want to compare the two objectives different objectives:

- maximizing the grade of service
- maximizing the number of granted ONUs

We will compare the bandwidth usage and the wasted bandwidth.

VII. Conclusions

An interesting evolution is to consider a mix of both objectives: we should require some request to be fully satisfied n order to be counted in the objective, while for other request each served ONU will be counted independantly.

Acknowledgment

Work of B. Jaumard has been supported by a Concordia Research Chair on the Optimization of Communication Networks and by a NSERC grant. Work of D. Coudert has been partially supported by ANR-JC OSERA, INRIA, European project IST FET AEOLUS, COST 293 and Région Provence Alpes Côte d'Azur PACA. The research done is in this paper was initiated during visits of B. Jaumard and D. Coudert in France and Québec, supported by a FQRNT-INRIA grant.

References

[1] M. Maier, M. Herzog, and M. Reisslein, "STARGATE: The Next Evolutionary Step toward Unleashing the Potential of WDM EPONs," IEEE COMMUNICATIONS MAGAZINE, vol. 45, no. 5, p. 50, 2007.

[2] C. Barnhart, E. Johnson, G. Nemhauser, M. Savelsbergh, and P. Vance, "Branch-and-Price: Column Generation for Solving Huge Integer Programs," OPERATIONS RESEARCH-BALTIMORE-, vol. 46, pp. 316–329, 1998.

Annexe D

Fiabilité et tolérance aux pannes

D.1 Reliability of Connections in Multilayer Networks under Shared Risk Groups and Costs Constraints

Article présenté lors d'ICC08.

D.2 Overlapping Segment for an Efficient Protection of WDM Networks

Article en préparation.

Reliability of Connections in Multilayer Networks under Shared Risk Groups and Costs Constraints

David Coudert, Florian Huc, Fabrice Peix, Marie-Emilie Voge
MASCOTTE, INRIA-I3S(CNRS/UNSA), Sophia-Antipolis, France
{firstname.lastname}@sophia.inria.fr

***Abstract*— The notion of Shared Risk Resource Groups (SRRG) has been introduced to capture survivability issues when a set of resources may fail simultaneously. Applied to Wavelength Division Multiplexing Network (WDM), it expresses that some links and nodes may fail simultaneously. The reliability of a connection therefore depends on the number of SRRGs through which it is routed. Consequently, this number has to be minimized. This problem has been proved NP-complete and hard to approximate in general, even when routing a single request. Some heuristics using shortest paths have already been designed, however the cost (the usual routing cost, not in term of SRRG) was not part of the objective. In this paper we study the problem of minimizing a linear combination of the average number of SRRG per paths and the cost of the routing. The main result of our work is a column generation formulation that allows to solve efficiently the problem of maximizing the reliability of a set of connection requests in MPLS/WDM mesh networks with SRRGs while keeping the cost of the routing low.**

Keywords: Reliability, Shared Risk Resource Groups, Integer Linear Programming, Column Generation.

I. INTRODUCTION

A challenging issue for service providers operating connection oriented networks, such as MPLS/WDM mesh networks, is to provide reliable routes for accepted requests. This has been fostered by the growing use of virtual and overlay networks which are embedded, in a transparent way, on an underlying network. Therefore, a great deal of researchers efforts have been concentrated on the design of efficient protection mechanisms at the backbone level (see for example [1]–[4]). At this level, the reliability of a single connection in the network might be measured as a function of the number of links it uses, each single link having its own probability of failure. However, in a physical network, several fibers, or links, may be packed together and hence physically cut simultaneously (earthquake, fire, malicious behaviour etc). Consequently, a single cause of breakdown induces several link failures. The concept of Shared Risk Resource Group (SRRG), or Shared Risk Group (SRG) for short, has been introduced to capture network survivability issues in this situation, i.e. when a set of resources may fail simultaneously. This concept can be extended to various kind of correlated failures.

A typical example of SRG is given by MPLS multilayer networks as described in [5]. This is illustrated by Fig. 1 where links AH and EI of the virtual network are both routed into the underlying network through link FG and so belong to a same SRG, the one associated to FG. Indeed a failure on the physical link FG induce the disruption of both virtual links AH and EI. In such multilayer networks with SRGs, the reliability of a connection is measured as a function of the SRGs through which it is routed.

Given a network and a set of requests, the problem of finding a routing minimizing the average number of SRG through which the route of a request goes has already been studied. It has been proved to be NP-hard [1], hard to approximate [6], and heuristic algorithms have been designed. For example [1] studied a tabu search heuristic algorithm for routing requests with different requirements. However, if we consider a network with edge cost, such an objective may lead to a costly routing. To avoid this, the objective to be minimized has to take into account two cost criteria. First we want to minimize the average failure probability, that is the average number of risks of failure on which a connection is dependent, as previously. Secondly, since an operator is interested in both the safety of the connections and their operating costs, the objective has to take into account the cost of the edges used along the route of the connections.

Hence, we concentrate on the problem of computing under capacity constraints a set of paths minimizing a linear combination of their edge costs and the number of SRGs through which they are routed. To our knowledge, this is the first time in the SRG context that such an objective, considering both the quality of the service offered to customers and the cost of the services, is proposed. Our main result is the first integer linear program (ILP) with column generation for the Minimum Average Color Flow problem, even when we consider only the objective of minimizing the average number of SRGs.

In this paper we also discuss a way to generate "realistic" virtual networks with SRRG. Since the simultaneous failures of links are correlated, a random assignment of SRRG to virtual links does not necessarily correspond to a feasible routing of the virtual topology into a real physical network, as was pointed out in [7]. The solution we propose avoid this problem and hence the experiments are run on graphs presenting more characteristics of real instances.

Our paper is organized as follows. In the next Section, we present the notations and network models used in the rest of the paper. Sec. III is dedicated to the Minimum Average Color Flow problem: once the problem is defined, we study its complexity and present two ILP formulations. In Sec. IV-A we remind some basics about column generation before applying it to Minimum Average Color Flow. We then detail in Sec. V the generation of our instances and the implementation of

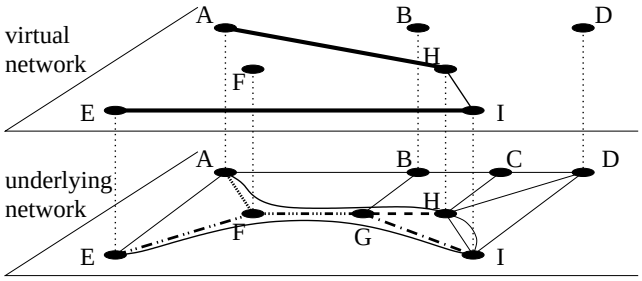


Fig. 1. Example of SRLG: virtual links AH and EI share the same risk FG.

our solving process, and finally we analyze the computational results before concluding.

II. Network model and notations

The multilayer networks we consider are composed of two layers, a physical or underlying network and a virtual one.

The physical layer is modelled by a digraph $G_{\rm PN} = (V_{\rm PN}, L_{\rm PN})$, where $V_{\rm PN}$ is the set of nodes, and $L_{\rm PN} = \{e_1, \dots e_m\}$ is the set of links (i.e. unidirectional fibers). Similarly, the virtual network embedded in the physical one is modelled by the digraph $G_{\rm VN} = (V_{\rm VN}, L_{\rm VN})$. Each arc $e \in L_{\rm VN}$ has a cost γ_e per flow unit and a capacity u_e. The construction of the virtual networks used in our simulations will be described more precisely in Sec. V.

Each arc e_i of the physical network G_{PN} gives in the virtual layer of a single risk group that we represent by a color c_i. More precisely, color c_i is the subset of arcs of the virtual network routed through arc e_i on the underlying layer, that is the set of arcs sharing the risk of failure of arc e_i. The set $\mathcal{C} = \{c_1, \dots, c_m\}$ is the set of colors (i.e. SRG) of the network. Note at that point that an arc of the virtual network may belong to several SRGs (= colors). However, following [1], [2], [5], we apply the simple transformation on $G_{\rm VN}$ described in [6], [8] : each multi-colored arc is replaced with a chain of mono-colored arcs, so that each arc now belongs to exactly one color. For simplicity purpose we will denote also by $G_{\rm VN}$ the modified graph. Such graphs with SRG are refered to in the literature as *colored graphs*. Interested readers may refer to [9] for a model with multiple colors per edges, and to [8] for a preliminary comparison of both models.

Let K be a set of connection requests over $G_{\rm VN}$, such that for each request $k \in K$, we are given its source s_k, its destination t_k and its integral bandwidth requirement d_k. We assume that each request can be split into unitary requests to be served independently. Consequently, in the rest of the paper all the requests are unitary. For each request $k \in K$, $\mathcal{P}^k$ is the set of all the $s_k t_k$-directed paths in $G_{\rm VN}$ and $\mathcal{P} = \cup_{k \in K} \mathcal{P}^k$

Finally, we denote respectively by $\Gamma^-(v)$ and $\Gamma^+(v)$ the set of incoming and outgoing links at a node v.

III. Minimum Average Color Flow Problem (MACF)

A. Problem definition

An instance of the Minimum Average Color Flow problem consists in a digraph $G_{\rm VN} = (V_{\rm VN}, L_{\rm VN})$, representing the virtual network, with arc cost γ_e per flow unit and capacity u_e for each arc $e \in L_{\rm VN}$. A set $\mathcal{C} = \{c_1, \dots, c_m\}$ of colors -or SRG- partitioning the arc set $L_{\rm VN}$ is given, as well as a set of unitary connection requests K and a real $\alpha \in [0, 1]$.

A solution is a routing of all the requests satisfying all the capacity constraints.

The objective to be minimized is a convex combination of two cost criteria whose respective contributions can be adjusted through the parameter $\alpha \in [0, 1]$. The first criterion, related to the operating cost, is the classical flow cost depending on the arc costs γ_e divided by the total flow request ($= |K|$). The second criterion, related to the safety of the connections, is the average number of colors - SRGs - to which belongs a path routing one unit of flow, that is the sum of the numbers of colors on which is routed each request divided by $|K|$. The objective function is then $\frac{\alpha}{|K|} \times$ (classical flow cost) $+ \frac{(1-\alpha)}{|K|} \times$ (SRG flow cost).

Note that when $\alpha = 0$ the problem becomes the minimisation of the average number of SRG or in other word the average probability of failure for a connection, whereas if $\alpha = 1$ the problem is reduced to a classical integral multicommodity flow.

B. Complexity

We now study the complexity of MACF under various hypothesis on the colored graph. We deduce these results for MACF from one of its subproblems: Minimum Color Path with Edge Costs problem (MCPwEC) which is the MACF problem with a single request, i.e. $|K| = 1$, and unitary arc capacities. Thus, it consists in finding a path from a source s to a sink t minimizing a convex combination of the number of colors used by the path and of the classical path cost.

1) Null arc costs: With null arc costs, this problem is the Minimum Color Path problem (MCP) which has been proved NP-hard in [2] and hard to approximate within a factor $2^{\log^{1-\delta} |\mathcal{C}|^{\frac{1}{2}}}$ in [6], with $\delta = (\log \log |\mathcal{C}|^{\frac{1}{2}})^{-\varepsilon}$ and $\varepsilon < \frac{1}{2}$. These results were proved for undirected graphs, but their adaptation to the directed case is straightforward and induces the same factor for MACF.

2) Span 1: MCP can be solved in polynomial time when some conditions are satisfied on the *span* of the colors of the graph [6], [8], [10]. The span of a color is the number of connected components in the subgraph induced by the edges of that color. The influence of this parameter on the MCP complexity has been studied in [6], [11]. However, we show that the span has no such an influence on the hardness of MCPwEC.

Theorem 1: [6] When all colors have span 1, MCP is polynomial.

The proof is based on the fact that within a connected component of a color c, each vertex is reachable from any other vertex of the component using only edges of color c.

The span definition and Theorem 1 can easily be extended to the *color symmetric digraph* settings, i.e. when an arc and its reverse arc belong to the same color: the span is simply the

number of strongly connected components and the property is still verified.

Let us now prove that the adjunction of edge costs deeply modify the complexity properties of MCP since MCPwEC is as hard to approximate as in the general directed case in the aforesaid symmetric settings.

Theorem 2: The MCPwEC problem on a color symmetric digraph is NP-hard and hard to approximate within a factor $2^{\log^{1-\delta}|\mathcal{C}|^{\frac{1}{2}}}$ with $\delta = (\log\log|\mathcal{C}|^{\frac{1}{2}})^{-\varepsilon}$ and $\varepsilon < \frac{1}{2}$ even when all colors have span 1.

Proof: The proof is a reduction from MCP to MCPwEC. Let G be a color symmetric digraph on color set $\mathcal{C}$ with unspecified span and two vertices s and t defining an instance of MCP. We construct an instance of MCPwEC on the digraph G' with convex coefficient α in the objective function. G' is the graph G with two additional symmetric arcs of color c and cost $X > \frac{1-\alpha}{\alpha}|\mathcal{C}|$ between each pair of strongly connected components of c. The cost of arcs from G in G' is null and all the capacities are unitary.

An optimal st-path for the MCPwEC instance uses none of the added arcs since using only ones cost $\alpha X > (1-\alpha)|\mathcal{C}|$, which is greater than the cost of using the whole arc and color sets of G. Therefore, an optimal path in G' for MCPwEC is an optimal path in G for MCP too. Indeed the two costs of a same path for MCPwEC and MCP differ only by the multiplicative constant $1-\alpha$. This last point concludes the proof. ■

From Theorem 2 we can deduce that in this special case MACF is as hard to approximate as its subproblem MCPwEC.

3) Color reduced to a single arc: Unlike MCPwEC and MCP, MACF remains NP-hard and hard to approximate when all colors are reduced to a single arc. Indeed in such a case, MCPwEC and MCP are merely polynomial shortest path problems while MACF still contains the Minimum Edge Cost Flow problem (MECF) as a subproblem. The MECF problem consists in routing a single integral request in a digraph with capacities and fixed arc costs, that is the cost of using an arc is the same whatever the flow value is on it, as long as it is positive. This problem is NP-hard and hard approximate within a factor $2^{\log^{1-\varepsilon} n}$, $\forall\varepsilon > 0$ [12].

Theorem 3: The MACF problem is hard to approximate within a factor $\approx 2^{\log^{1-\varepsilon} n}$ $\forall\varepsilon > 0$ even when all colors are reduced to a single arc, unless $NP \subseteq DTIME(n^{\text{polylog}n})$.

C. ILP Formulations

In this section we present two ILP formulations for MACF based on the two classical multiflow formulations: the compact node-arc formulation and the exponential size arc-path one (see [13]). Although exponential, the arc-path formulation is more efficient than the compact one thanks to the possible use of column generation techniques.

1) Node-Arc Formulation: The binary flow variable z_e^k for $e \in L_{\text{VN}}$ and $k \in K$ gets value 1 if the virtual link e is used to route the unitary request k and 0 otherwise. The binary color variable χ_k^c, for $k \in K$ and $c \in \mathcal{C}$, gets values 1 if the unitary request k is routed through an arc belonging to the risk group represented by color c.

$$\min \frac{1}{|K|}\sum_{k\in K}\left(\alpha\sum_{c\in\mathcal{C}}\chi_k^c + (1-\alpha)\sum_{e\in A}\gamma_e z_e^k\right)$$

$$\sum_{e\in\Gamma^+(v)} z_e^k - \sum_{e\in\Gamma^-(v)} z_e^k = \begin{cases} 1 & \text{if } v = s_k \\ -1 & \text{if } v = t_k \\ 0 & \text{otherwise} \end{cases} \quad \forall k \in K \qquad (1)$$

$$\sum_{k\in K} z_e^k \le u_e \qquad \forall e \in L_{\text{VN}} \qquad (2)$$

$$z_e^k \le \chi_k^{c(e)} \qquad \forall k, \forall e \in L_{\text{VN}} \qquad (3)$$

Constraints (1) and (2) are the classical flow conservation and capacity constraints of a node-arc multiflow formulation. Constraint (3) ensures that whenever the flow for a request k on an arc e is positive, the color $c(e)$ of this arc is counted as used for this request, i.e. the value of $\chi_k^{c(e)}$ is forced to 1, where $c(e)$ is the color of arc e. There are $|K|(|L_{\text{VN}}| + |\mathcal{C}|)$ variables and $|K|(|V_{\text{VN}}| + |L_{\text{VN}}|) + |L_{\text{VN}}|$ constraints.

The objective to be minimized is the average on all requests of a convex combination of the number of colors used by a request and of the classical cost of the route of a request.

2) Arc-Path Formulation: We now use binary variable z_p^k which gets value 1 if the request $k \in K$ is routed on the path $p \in \mathcal{P}$, and 0 otherwise. As there might be a large number (exponential) of valid paths for each request, there might be a large number of variables in this family. As in the node-arc formulation, the binary variable χ_k^c has value 1 if the path used by request k uses an arc of color c and 0 otherwise.

$$\min \frac{1}{|K|}\sum_{k}\left(\alpha\sum_{c\in\mathcal{C}}\chi_k^{c(e)} + (1-\alpha)\sum_{e\in L_{\text{VN}}}\sum_{P\in\mathcal{P}^k|e\in P}\gamma_e z_P^k\right)$$

$$\sum_{P\in\mathcal{P}^k} z_P^k = 1 \qquad \forall k \in K \qquad (\sigma_k) \qquad (4)$$

$$\sum_{k}\sum_{P\in\mathcal{P}^k|e\in P} z_P^k \le u_e \quad \forall e \in L_{\text{VN}} \qquad (\omega_e) \qquad (5)$$

$$\sum_{P\in\mathcal{P}^k|e\in P} z_P^k \le \chi_k^{c(e)} \qquad \forall k \in K, \forall e \in L_{\text{VN}} \quad (\pi_e^k) \qquad (6)$$

Constraints (4) ensure the satisfaction of all requests: for a given request k there must be a path used to route it; we call σ_k the associated dual variable. The capacity constraint (5) on each arc expresses that there can not be more paths using an arc e than there is bandwidth to fit them; we call ω_e the associated dual variables. As in the node-arc formulation, constraints (6) take into account the colors used to serve each request. The objective function is also similar. There are $|K|(|L_{\text{VN}}| + 1) + |L_{\text{VN}}|$ constraints and empirically at most $10|K|$ variables, which is a lot smaller (for both variables and constraints) than in the node-arc formulation.

Note that to apply the column generation technique, or more precisely to define the auxiliary problem, we need the dual constraint associated to the large size family of variables z_p^k.

IV. Solving Minimum Average Color Flow

A. Principle of column generation

A column generation formulation is a transformation of a linear program containing a very large size variable family (e.g., exponential) into two subproblems: the *master problem* and the *auxiliary problem*. This decomposition relies on the Dantzig-Wolf decomposition [13]. It allows to consider only a subset of variables at a time.

Basically, starting with an initial reduced set of variables, we solve the master problem and obtain dual variable values that will be used by the auxiliary problem. Then, we solve the auxiliary problem to obtain new variables for the master problem, and so on. We repeat this loop until the auxiliary problem finds no more variables for the master problem. The solution of the master problem is then optimal (see [14]).

Column generation can be combined with branch-and-bound techniques to solve integral linear programs with a large number of variables, see [15] for a nice overview.

B. Column generation applied to MACF

1) Master Problem: The master problem corresponds to the original arc-path formulation in which only a subset of the variables from the large size families $\{z_p^k\}_{p\in\mathcal{P}^k}$ is considered, all other variables of these families simply do not exist.

2) Auxiliary Problem: The auxiliary problem is deduced from the dual constraints of the master problem associated to the large size family of variable z_p^k:

$$\sigma_k \leq \sum_{e\in P}\left(\frac{(1-\alpha)}{|K|}\gamma_e + \omega_e + \pi_e^k\right) \quad \forall k, \forall P \in \mathcal{P}^k$$

A solution of the auxiliary problem is a path P violating this constraint. In our case the auxiliary problem is a shortest path problem on G_{VN} with arc cost $\frac{(1-\alpha)}{|K|}\gamma_e + \omega_e + \pi_e^k$. We have to solve the auxiliary problem for each request k. Notice that the arc costs depend on the request, indeed the *color cost* ($\chi_k^{c(e)}$) is not the same for each request.

3) Generation of the initial solution: The initial set of variables of the master program has to contain the non null variables of a feasible solution otherwise the column generation process can not start. To find this initial subset of variables we solve a classical minimum cost multiflow problem: from MACF we keep only the arc costs and capacities and we try to satisfy all the requests under the capacity constraints (see [13] for references on the minimum cost multiflow problem).

C. Obtaining integral solutions: Branching

We first solve with column generation a relaxation of the problem in which the variables z_P^k and χ_k^c are fractional. Then we do branching to obtain an integral solution. During the branching a lot of computational time can be saved by considering the χ_k^c as fractional variables. Indeed, since all requests are unitary, once the flow variables are integral, the constraints force the χ_k^c variables to be integral too.

V. Experiments

We have used Cplex with a Concert Java interface and the open source mascopt library [16] to implement our model.

A. Generation of instances

To give a realistic significance to the colors, we assigned them to the virtual links according to a real routing of the virtual layer on a physical network as follows.

We used two networks as physical layers : NSFnet (14 nodes and 21 links) and Brazil (27 nodes and 70 links) [17]. Each physical link and node is associated a distinct color. if the network is already precolored with meaningful groups of risk, we may use them as well.

The virtual layer was generated using the following steps.

1) We generate an undirected graph on the physical vertex set using a *pseudo random* graph generator, implemented in mascopt, which accepts a set of constraints on the resulting graph. We choose the following ones: diameter at most 3, minimum cut at least 2, and number of links around 3 times the number of nodes.
2) We replace each edge by two symmetrical arcs.
3) We randomly generate unitary requests: 140 requests for NSFnet and 200 for Brazil.
4) We compute a *double multiflow*: the demands are routed on the generated graph whose arcs are themselves routed on the physical network simultaneously.
5) Each arc of the generated graph corresponds to a set of parallel virtual links : an arc is routed on a set of paths, each one giving a distinct virtual link. To each virtual link we assign the colors of the physical resources through which is routed its associated path.
6) The routing of the requests on the virtual network gives a weight to each virtual link. Thus, the capacity given to each virtual link v_l is the minimum over each physical link p_l of the proportion of utilisation of p_l by v_l among all virtual links multiplied by the capacity of p_l, that is: $\min_{p_l \ni v_l} \frac{\text{weight}_{(v_l)}\text{capacity}_{(p_l)}}{\sum_{v'_l\in v_p}\text{weight}_{(v'_l)}}$
7) We transform the networks given by the virtual links into a mono-colored graph [6] since at that point it may be multi-colored.

B. Implementation

We now briefly describe our implementation to allow a better understanding of next section V-C.

1) Columns generation: As explained in IV-A, the column generation is based on two problems (*master* and *auxiliary*). In our implementation both problems are solved using Cplex. During the process, we only add columns to the master problem without removing any.

2) Obtaining an integral solution: We first implemented a branch-and-price algorithm using pseudo-cost to order the variables [15] and depth-first-search to visit the branch-and-price tree. However, our implementation is not yet efficient enough due to the lack of significant cuts. Therefore, we decided to obtain an integral solution via Cplex using only the set of variables selected during the column generation process. The obtained solution might not be optimal, but close to it in our experimentation. We asked Cplex to guarantee a gap with

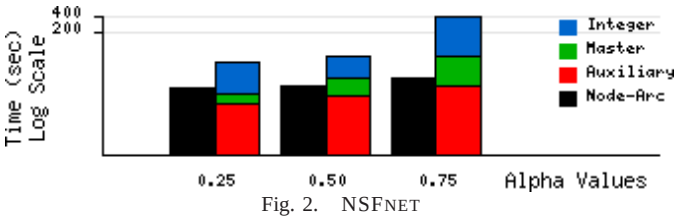


Fig. 2. NSFNET

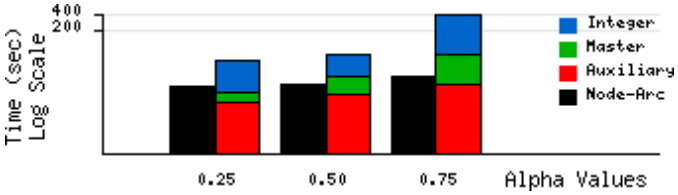


Fig. 3. BRAZIL

optimality of less than 5%, and in practice the gap was less than 5‰.

C. Computational results

We made all our tests on a *Intel* Core 2 2.4 GHz with 4Mb of L1 cache and 2 Gb of memory. We present our results in two graphics, one for NSFNET and one for BRAZIL. We solved each problem ten times using both formulations asking for an integer solution at at most 5% of the optimal (except when $\alpha = 0.75$, where we ask for 10%). The high of the rectangle represents the total time spent to find the solution. For the arc path formulation , the rectangle is splited accordingly to the time spent to solve the master problems, the auxiliary problems and to find an integral solution.

1) MACF: column generation vs node-arc formulation:

a) Computational time: Surprisingly, the node-arc formulation is always faster than the column generation formulation whereas there is a lot more variables in the first one. This can be partly explained by our implementation (we used a Java interface to call CPLEX which slows down the process) but also by the constraints on the color variables. Indeed in the node-arc formulation, a color constraint involves only two variables, while it may involve several ones in the column generation formulation. The color variables being the most numerous, it has a great impact on the resolution time.

In addition in the column generation formulation, each time we solve the master problem, we have to solve an auxiliary problem for each request (cf IV-B.2).

b) Size of the instances solved: The column generation formulation allows to solve larger instances than the node-arc formulation, where the number of variables is too large.

c) Influence of the colors: The resolution time increases with α for both formulations due to the increasing importance of the color variables, which are 0-1 variables and numerous. It is worth noting that in the column generation formulation, the quality of the initial solution decreases when α increases since it does not take the colors into account at all.

VI. CONCLUSION

In this paper we have investigated the problem of maximizing a linear combination of the reliability and the cost of a set of connection requests in mesh networks with SRRG. We have shown that MACF is NP-hard and hard to approximate, and we have proposed a column generation formulation to solve it that allows to obtain optimal or near optimal solutions in reasonable time. We will now pursue our implementations and extend our work to the multiple-path problem where each connection request is protected by a SRRG-disjoint path and to other classical protection schemes. Note that SRRG-disjoint path in multilayer networks and with a different objective function currently under investigation in [18].

ACKNOWLEDGMENTS

This work was supported by ANR JC OSERA, région PACA, European projects IST FET AEOLUS and COST 293 Graal.

REFERENCES

[1] L. Shen, X. Yang, and B. Ramamurthy, "Shared risk link group (SRLG)-diverse path provisioning under hybrid service level agreements in wavelength-routed optical mesh networks," *IEEE/ACM Transactions on Networking*, vol. 13, no. 4, pp. 918–931, 2005.

[2] S. Yuan, S. Varma, and J. Jue, "Minimum-color path problems for reliability in mesh networks," in *IEEE INFOCOM*, vol. 4, 2005, pp. 2658–2669.

[3] C. Liu and L. Ruan, "p-cycle design in survivable WDM networks with shared risk link groups (SRLGs)," *Photonic Network Communications*, vol. 11, no. 3, pp. 301–311, May 2006.

[4] S. Ramamurthy and B. Mukherjee, "Survivable WDM mesh networks, part 1: Protection," in *IEEE INFOCOM*, 1999, pp. 744–751.

[5] D. Papadimitriou, F. Poppe, J. Jones, S. Venkatachalam, S. Dharanikota, R. Jain, R. Hartani, and D. Griffith, "Inference of shared risk link groups," IETF Draft, OIF Contribution, OIF 2001-066.

[6] D. Coudert, P. Datta, S. Perennes, H. Rivano, and M.-E. Voge, "Shared risk resource group: Complexity and approximability issues," *Parallel Processing Letters*, vol. 17, no. 2, pp. 169–184, Jun. 2007.

[7] D. Coudert, S. Perennes, H. Rivano, and M.-E. Voge, "Shared risk resource groups and colored graph: Polynomial cases and transformation issues," HAL, Tech. Rep. inria-00175143, Sep. 2007. [Online]. Available: http://hal.inria.fr/inria-00175143/fr/

[8] M.-E. Voge, "How to transform a multilayer network into a colored graph," in *IEEE-LEOS ICTON/COST 293 GRAAL*, vol. 3, Jun. 2006, pp. 116–119.

[9] A. Faragó, "A graph theoretic model for complex network failure scenarios," in *Proceedings of the Eighth INFORMS Telecommunications Conference*, Dallas, Texas, March 2006.

[10] P. Datta and A. Somani, "Diverse routing for shared risk resource groups (SRRG) failures in WDM optical networks," in *IEEE BroadNets*, Oct. 2004, pp. 120– 129.

[11] D. Coudert, S. Perennes, H. Rivano, and M.-E. Voge, "Shared risk resource groups and survivability in multilayer networks," in *IEEE-LEOS ICTON / COST 293 GRAAL*, vol. 3, Jun. 2006, pp. 235–238.

[12] G. Even, G. Kortsarz, and W. Slany, "On network design problems: fixed cost flows and the covering steiner problem," *ACM Trans. Algorithms*, vol. 1, no. 1, pp. 74–101, 2005.

[13] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network flows: theory, algorithms, and applications*. Prentice Hall,, 1993.

[14] V. Chvatal, *Linear Programming*. Freeman, 1983.

[15] C. Barnhart, E. Johnson, G. Nemhauser, M. Savelsbergh, and P. Vance, "Branch-and-price: Column generation for solving huge integer programs," *Operations Research*, vol. 46, no. 3, pp. 316–329, 1998.

[16] J.-F. Lalande, M. Syska, and Y. Verhoeven, "Mascopt - a network optimization library: Graph manipulation," INRIA Sophia Antipolis, Tech. Rep. RT-0293, 2004. [Online]. Available: http://www-sop.inria.fr/mascotte/mascopt

[17] T. Noronha and C. Ribeiro, "Routing and wavelength assignment by partition coloring," *European Journal of Operational Research*, vol. 1713, no. 3, pp. 797–810, Jun. 2006.

[18] S. Orlowski and M. Pioro, "Primal and dual formulations of network reconfiguration problems and related path generation issues," 2007, manuscript.

A New Framework for Efficient Shared Segment Protection Scheme for WDM Networks

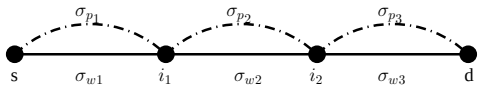
Brigitte Jaumard[†], Florian Huc[*], Nazmum Bhuiyan[‡], David Coudert[*]
[*] Mascotte project, I3S(CNRS/UNSA)/INRIA, Sophia-Antipolis, France
[†] GERAD, CIRRELT, and CIISE, Concordia University, Montréal, Canada
[‡] ECE, Concordia University, Montréal, Canada

***Abstract*— This work introduces a new shared segment protection scheme that fully exploits the hop endpoints of the primary logical hops (induced by the routing) without adding extra ones for protection and ensure both node and link protection in an efficient manner both in terms of cost and bandwidth. Contrary to the link or path protection scheme, the segment protection one has been less studied although it offers an interesting compromise between those two protection schemes, attempting to encompass all their advantages. We investigate two different Shared Segment Protection (SSP) schemes: Basic Shared Segment Protection (BSSP) and Shared Segment Protection with segment Overlap (SSPO). The latter one is compared with the Short Leap Shared Protection (SLSP) scheme introduced by Ho and Mouftah (2002) in terms of architecture design and signaling mechanisms. The key difference lies in the overlapping of working segments in SLSP and of protection segments in SSPO. For both BSSP and SSPO schemes, we propose two novel efficient ILP formulations, based on a column generation mathematical modeling. While (SSPO) offers the advantage over (BSSP) to ensure both node and link protection, it is not necessarily more costly. Indeed, depending on the network topology and the traffic instances, it can be shown that none of the two SSP schemes dominates the other one. Therefore, the SSPO protection scheme should be favored as it offers more protection, i.e., it adds the node protection to the link protection.**

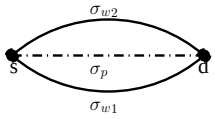
I. Introduction

Wavelength Division Multiplexing (WDM) is a well known technology that allows an efficient use of the high bandwidth offered by optical networks [1], [2]. Under WDM, laser beams are used to implement fixed end-to-end connections in the network - called lightpaths - under the constraint that two lightpaths cannot share the same wavelength over the same fiber. The Routing and Wavelength Assignment (RWA) problem consists thus in assigning a route in the network and a wavelength to each lightpath. This NP-hard problem has been widely studied during the last 15 years (see, e.g., [3], [4]) and optimal or near-optimal solutions can now be obtained in reasonable time using proper integer linear programming formulation ([5], [6]).

Since the bandwidth requirement of a request is usually smaller than the bandwidth offered by a single wavelength (Mbits versus Gbits), several requests may be groomed on a same wavelength in a Time Division Multiplexing (TDM) manner as it is the case in SONET/SDH networks. The problem of determining which requests to groom and which route to associate with the groomed requests corresponds to the so-called Grooming, Routing and Wavelength Assignment (GRWA) problem ([7], [8], [9], [10], [11]). To add a request onto a wavelength, that is to groom it with other request, or to extract it from a wavelength, we use SONET Add-Drop Multiplexers (ADM) that convert optical signals into electronic ones and vice versa in order to conduct the add/drop operations. Requests can be added or dropped from the traffic stream when they are in their electronic form. A segment is a bundle of requests groomed together on a given wavelength λ between two nodes and that are not encountering any electronic conversions, i.e., between two nodes that are equipped with a λ ADM. Therefore any working/protection segment uses two ports of ADMs devices, an output port at the origin of the segment where the signal is converted from electronic to optical in order to add one or more requests, and an input port at the destination of the segment where the signal is converted from optical to electronic in order to drop one or several requests. ADMs consists in a set of blades, where each blade is made of one output and one input port. The number of blades used to route the traffic is a key factor in the overall cost of the network and its optimization is a problem in itself (see for example [12], [9]). We will use it as an estimation of the protection provisioning cost.



(a) BSSP



(b) shared protection

Fig. 1. SSP Protection Scheme

In the current study, we focus on *Shared Segment Protection* schemes in the context of a sequential framework where the working segments are first defined (using any given GRWA algorithm) and then the protection scheme is defined. We

therefore assume that we are given a set of working paths, where each working path is either single-hop or multi-hop, i.e., made of one or several (up to 3) working segments. Transport blades are installed at each endpoint of a working segment. This induces a natural segmentation of the lightpath that can be used as a base for the protection scheme in order to save on the network cost.

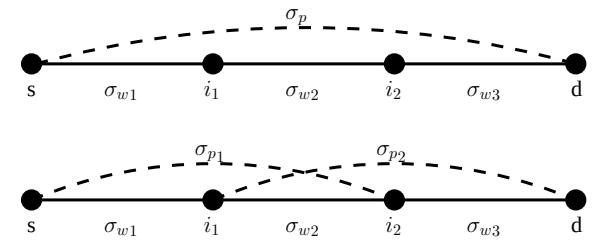
A first segment protection, called BSSP, is such that a protection is defined for each working segment, and therefore such that both working and protection segments have the same endpoints. A second segment protection, called SSPO, is such that for 3-hop working paths, we allow protection segments to encompass two working segments in order to reduce the bandwidth cost, but also and firstly to ensure node protection. This entails some overlapping of the protection segments. In the next section, we will compare SSPO with the SLSP protection scheme of Mouftah and Ho [13] where the overlapping is at the level of the working segments. In both BSSP and SSPO, protection paths can be shared by several node and link disjoint working segments, it is illustrated in Figure I(b) where σ_p protects two disjoint working segments.

The paper is organized as follows. In the next section, we have a qualitative comparison of the BSSP, SSPO and SLSP protection schemes. In the following sections, we investigate efficient Integer Linear Programming (ILP) formulations to solve models for designing minimum cost protection schemes using first the BSSP scheme (Section IV) and then the SSPO scheme (Section V). Both ILP formulations rely on column generation methods which have now been shown to be extremely efficient for solving highly combinatorial problems, see, e.g., Barnhart *et al.* [14]. Conclusions are drawn in the last section.

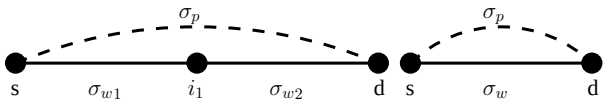
II. Shared Segment Protection

BSSP protection has already been studied by Bouffard [15]. Although less greedy in terms of bandwidth than shared link protection, and with a nice compromise for recovery time between link and path protection, it lacks a full protection against node failures (i.e., failure of a device, as ADM, located at a node: in Figure I(a) neither node i_1 nor i_2 are protected).

Ho and Mouftah introduced in [13] the Short Leap Shared Protection (SLSP) scheme as an extension of BSSP, simultaneously protecting against node failure and fiber cut. In SLSP the working path is subdivided into several equal length and overlapped segments, each assigned (by the source node) a protection domain after the working path is selected. In this paper we consider a new protection scheme also protecting against both fiber cut and node failure based on the segmentation induced by the routing. The *Shared Segment Protection with Overlap* (SSPO) scheme consists in protecting each working segment simultaneously with at least an other working segment (except for a request routed on a single working segment, in this case the SSPO associate an end to end protection to the working segment). Since there are ADMs at each endpoint of a working segment, it means that at any node lying between two working segments, there is an optical/electronic or electronic/optical conversion. If we use those nodes as endpoints for the protection segments, whether they are source or destination, we use only one input/output of an ADM in order to put the protection segments in place. In the SSPO scheme, protection segments overlap as they are designed to protect several working segments simultaneously and not only a single working segment as in the SLSP scheme. This overlap between adjacent protection segments aims at ensuring node protection for all nodes, except for the source and destination nodes. All different types of protection segments in the SSPO scheme, are shown on Figure 2. Note also that a protection segment can be shared with several working segments as long as they are pairwise disjoint.



(a) A request with 3 working segments



(b) A request with 2 working segments

(c) A request with 1 working segments

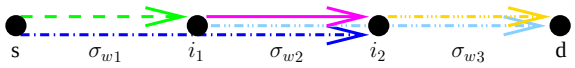
Fig. 2. SSPO Protection Scheme

From BSSP and SSPO, several variants can be studied. For example if delay is not a main issue, the protection segment may be divided into several segments so that to increase segment sharing and so to decrease the bandwidth needed for protection.

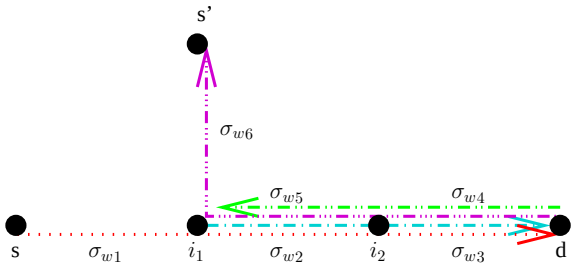
Finally, note that shared risk group constraints can also be taken into account by modeling two working segments belonging to the same risk group as conflicting working segments (see, e.g., [16]).

A. BSSP vs. SSPO

Depending on the network topology and the set of working segments (that depend in turn on the set of requests and the GRWA algorithm used to define these segments), there is no dominance of either the BSSP or the SSPO protection scheme in terms of both bandwidth and cost as evaluated through the number of transport blades. Therefore, at equal or similar cost, SSPO should be favored over BSSP as it offers a better

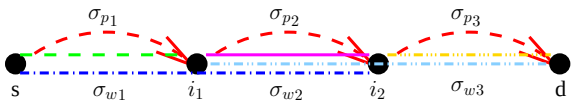


(a) First set of working segments in a grid

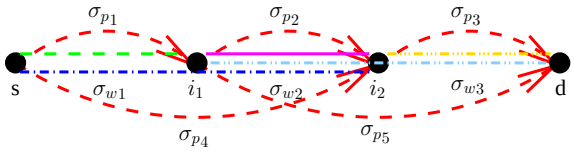


(b) Second set of working segments

Fig. 3. Two possible types of interaction between working segments



(a) Without overlap: 3 protection segments and 4 blades.



(b) With overlap: 5 protection segments and 8 blades.

Fig. 4. BSSP/SSPO protections for the example of Figure 3(a).

protection scheme, i.e., node and link failure vs. link failure only.

Let us examine the two examples depicted in Figure 3. Note that both examples are quite generic patterns that could be easily encountered in a given network and traffic instance. The first example (Figure 3(a)) is associated with a set of 5 requests, $\{k_1, k_2, k_3, k_4, k_5\}$ such that: $k_1 : s \longmapsto i_1$ on 1 segment, $k_2 : s \longmapsto i_2$ on 2 segments, $k_3 : i_1 \longmapsto i_2$ on one segment, $k_4 : i_1 \longmapsto d$ on two segments, and $k_5 : i_1 \longmapsto d$ on one segment. k_1 and k_2 are groomed from s to i_1 to form σ_{w1}, k_2, k_3 and k_4 are groomed from i_1 to i_2 to form σ_{w2}, k_4 and k_5 are groomed from i_2 to d to form σ_{w3}.

The second example is associated with a set of 4 requests, $\{k_1, k_2, k_3, k_4\}$ such that $k_1 : s \longmapsto d$, $k_2 : d \longmapsto s'$, $k_3 : i_1 \longmapsto d$, $k_4 : d \longmapsto i_1$, Let us assume that there are routed on wavelengths using the following working segments: $\sigma_{w_1} : s \longmapsto i_1$, $\sigma_{w_2} : i_1 \longmapsto i_2$, $\sigma_{w_3} : i_2 \longmapsto d$ for request k_1; $\sigma_{w_4} : d \longmapsto i_2$, $\sigma_{w_5} : i_2 \longmapsto i_1$, $\sigma_{w_6} : i_1 \longmapsto s'$ for request k_2; $\sigma_{w_2} : i_1 \longmapsto i_2$, $\sigma_{w_3} : i_2 \longmapsto d$ (groomed with k_1 on both of them) for request k_3; $\sigma_{w_4} : d \longmapsto i_2$, $\sigma_{w_5} : i_2 \longmapsto i_1$ (groomed with k_2 on both of them) for request k_4.

For the first example, a SSPO protection requires 5 protection segments and 8 blades (2 per node) and is thus more expensive than a BSSP protection that uses only 3 protection segments and 4 blades (1 per node) as shown in Figure 4. On the opposite, SSPO protection is more economic than BSSP for the second example as shown in Figure 5 (For readability purpose, we do not represent on this figures the requests which are represented on Figure 3(b)). The protection requires 5 blades (1 per node) vs. 7 blades (2 blades at nodes i_1 and i_2 and 1 blade at each of the other nodes).

Note also that depending again on the network topology and on the definition of the working segments, while it may not be possible for one of the protection scheme to define a protection for all requests (e.g. lack of available wavelengths), it may be possible for the other one, and vice-versa.

III. Notation and Definitions

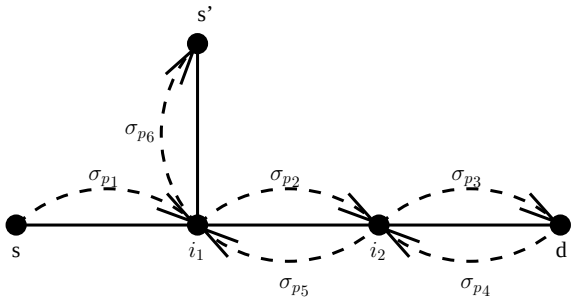
Consider a WDM network represented by a directed graph $G = (V, L)$ where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes in one to one correspondence with the set of network nodes, and $L = \{\ell_1, \ell_2, \ldots, \ell_m\}$ is the set of arcs, each arc being associated with a fiber link. Given $v \in V$, we denote the set of incoming and outgoing arcs of v by respectively $\omega^-(v)$ and $\omega^+(v)$.

The traffic is a set K of requests, such that for each request $k \in K$, we are given : its source s_k and its destination d_k, its bandwidth requirement b_k and its working path represented by its set of no more than three working segments, $\mathcal{S}_k^W$. Let $\mathcal{S}^W = \bigcup_{k \in K} \mathcal{S}_k^W$ be the set of all working segments. Note that each working segment σ_w is associated with a lightpath made of a path from the source node of σ_w, denoted by $v_s(\sigma_w)$, to its destination node $v_d(\sigma_w)$ and a wavelength λ. It follows that each working segment is associated with all requests groomed from $v_s(\sigma_w)$ to $v_d(\sigma_w)$ on λ on this path, on a unique wavelength.

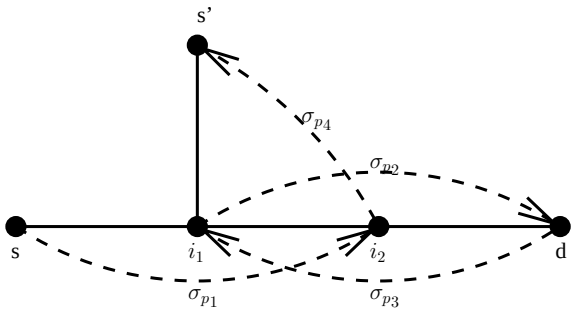
Let K^i be the set of requests with a working path using i segments, $i = 1, 2, 3$.

The shared segment protection (SSP) problem is expressed as follows: *for each $k \in K$, associate a protection segment σ_p to each of its working segment σ_w*. To ensure protection, working and protection segments must be edge disjoint, for SSPO protection they also need to be node disjoint. For BSSP, we call $\mathcal{S}_w^P$ the set of suitable protection segments for a working segment σ_w. For SSPO we call $\mathcal{S}_{w,k}^P$ the set of suitable protection segments for a working segment σ_w and a request k that uses it. We call $\mathcal{S}^P = \bigcup_{\sigma_w \in \mathcal{S}^W} \mathcal{S}_w^P$ or $\mathcal{S}^P = \bigcup_{\sigma_w \in \mathcal{S}^W} \mathcal{S}_{w,k}^P$ the set of protection paths. Note that $\mathcal{S}^P$ depends of the type (overlapping or non overlapping) of protection.

The objective is to *minimize the number of transport blades*. Note that both ports of a blade have the same transport



(a) Without overlap: 6 protection segments and 7 blades.



(b) With overlap: 4 protection segments and 5 blades.

Fig. 5. BSSP/SSPO protections for the example Figure 3(b).

capacity, but not necessarily the same wavelength. Moreover, a transport blade cannot be such that one of its port is used in a working path, and the other one in a protection path.

Two protection segments are in conflict if they use the same wavelength on the same fiber link since we can not use them simultaneously. Two working segments can be protected by two conflicting protection segments if and only if they do not share any fiber link. For SSPO protection, we add the condition that they do not share any node except for their endpoints. Indeed, if a fiber link shared by two working segments is cut, we need to reroute each pair of working segments on two different alternative paths. We use the following parameter:

$$\delta_{ww'} = \begin{cases} 1 & \text{if } \sigma_w \text{ and } \sigma_{w'} \text{ can be protected by the same protection segment} \\ 0 & \text{otherwise.} \end{cases}$$

IV. BSSP Protection Scheme

In this section, we will restrict our attention to the BSSP protection. We propose to investigate a column generation formulation in order to find optimal protection design with the BSSP scheme. We will outline the main features and advantages of column generation formulations in the next paragraph, and then detail the proposed model in the following paragraphs.

A. An Overview on Column Generation Models

Column generation techniques offer solution methods for linear programs with a very large number of variables (e.g., exponential) where constraints can be expressed implicitly. They rely on a decomposition of the initial linear program into the *master problem* and the *pricing problem*. The master problem corresponds to a linear program associated with a restricted constraint matrix, with respect to the number of variables (or columns) of the initial constraint matrix. The pricing problem is defined by the optimization of the so-called reduced cost subject to the implicit constraints expressed by the coefficients of the constraint matrix of the master problem. In some cases, there may be several pricing problems if, e.g., there are several types of columns.

The column generation solution scheme is similar to that of the simplex algorithm: it is an iterative process where, at each step, we attempt to add one or more columns to the constraint matrix of the master problem in order to improve the value of its objective function. The search for such columns is made through the solution of the pricing problem. If its outcome corresponds to one or more columns with a negative reduced cost (assuming we deal with minimization), then it entails an improvement of the value of the master objective function; otherwise, if no solution of the pricing problem can be identified with a negative cost, we then conclude that the current solution is indeed optimal.

Column generation can be combined with branch-and-bound techniques for solving integer linear programs with a large number of variables, see [14] for a nice overview. Branching rules have to be devised properly in order to avoid generating a huge number of subproblems in the search tree associated with the branch-and-bound, either by branching on the variables of the master problem using cuts, or by branching on the variables of the pricing problem using classical branching schemes or cuts.

B. The CG-BSSP Column Generation Model with Wavelength BSSP Protection Configurations

We propose a column generation model, denoted by CG-BSSP, based on *wavelength BSSP protection configurations*. For each wavelength λ, we define a *wavelength BSSP protection configuration*, as a set of protection segments following the BSSP protection scheme, all routed on λ, which protect a given set of working segments that are not necessarily routed on the wavelength λ. In order to reduce the number of configurations to be explored, we only consider the maximal ones, i.e., those are not embedded in larger ones with identical cost.

Such a column generation model leads to a decomposition where the master problem takes care of selecting the best configurations, one for each wavelength, i.e., the set of configurations that minimizes the cost as evaluated by the number of transport blades. The pricing problem identifies the best possible configurations, and therefore handles the constraints associated with the definition of a protection segment, i.e., no link sharing between a working segment and its protection,

protection sharing whenever it helps to reduce the cost. Note that due the strength of column generation, only a very small number of configurations will indeed need to be generated using the expression of the reduced cost in order to identify the most promising configurations, or to conclude that there exists no more configurations than those already included in the constraint set of the master problem will be able to improve the objective of the master problem, i.e., to reduce the cost of the transport blades, see, e.g., some previous work where the strength of column generation has allowed an efficient solution of highly combinatorial network design or provisioning problems [17].

1) Master Problem: In the CG-BSSP model, a column (or wavelength protection configuration) is associated with a wavelength λ and corresponds to a set of segments routed on λ, which can protect a given set of working segments, the largest one for the minimum transport blade cost. Note that, since two protection paths routed on two different wavelengths are not in conflict, we can select the columns independently one from the others. The pricing problem will identify eligible configurations for each wavelength. Solving the master will lead to a solution, i.e., a BSSP protection scheme, defined by a selection of configurations, one for each wavelength.

Each variable z_C of the master problem is associated with a configuration $C \in \mathcal{C}^\lambda$ for a given wavelength λ: $z_C = 1$ if the C configuration, $C \in \mathcal{C}^\lambda$, is selected on wavelength λ, and 0 otherwise. Let $\mathcal{C}^{\text{BSSP}}$, or $\mathcal{C}$ for short when there is no confusion, be the overall set of wavelength BSSP protection configurations, to any configuration C is associated a cost B_C. Although $\mathcal{C}$ is a huge set following its definition, in practice only a small number of its elements (e.g., few hundred for large network and traffic instances) will need to be listed in order to get an optimal or a near optimal solution, see again [17]. Let a^C be the column of the constraint matrix associated with the variable z_C. Each working segment is associated with a component of a^C: $a^C_{\sigma_w} = 1$ if σ_w can be protected under this configuration, and 0 otherwise. We then get the following mathematical model for the master problem:

$$\min \sum_{\lambda \in \Lambda} \sum_{C \in \mathcal{C}^\lambda} B_C z_C$$

subject to:

$$\sum_{C \in \mathcal{C}^\lambda} z_C \leq 1 \qquad \lambda \in \Lambda \qquad (u_0^\lambda) \qquad (1)$$

$$\sum_{\lambda \in \Lambda} \sum_{C \in \mathcal{C}^\lambda} a^C_{\sigma_w} z_C \geq 1 \qquad \sigma_w \in \mathcal{S}^W \qquad (u_{\sigma_w}) \qquad (2)$$

$$z_C \in \{0,1\} \qquad C \in \mathcal{C}, \qquad (3)$$

where u_0^λ and u_{σ_w} denote the dual variables associated respectively with constraints (1-λ) and (2-σ_w) (we will use those variables in the definition of the pricing problem in the next section). Constraints (1) express that we can use at most one configuration per wavelength. Constraints (2) translate the condition that each working segment must be protected at least once. Note that, in some cases, due to the search for maximal wavelength protection configurations, we may end up with "over" protection of some working segments: not only it does not affect the transport blade cost, but we could argue that in case of higher order failure, it might be useful.

2) Pricing Problems: There are as many pricing problems as the number of wavelengths in order to take into account the wavelengths assigned to the working segments. Consider the auxiliary graph $G_\lambda = (V, L_\lambda)$ where $L_\lambda = \{e \in L : (e,\lambda) \notin \sigma_w \text{ for } \sigma_w \in \mathcal{S}^W\}$. In order to define protection segments, we use a flow modeling formulation where each segment σ_p that protects a given working segment σ_w is associated with a unit flow from $v_s(\sigma_w)$ to $v_d(\sigma_w)$ where $v_s(\sigma_w)$ and $v_d(\sigma_w)$ denotes respectively the source and the destination nodes of the σ_w segment. We therefore introduce flow variables $\varphi^{\sigma_w}_{e\lambda}$ such that $\varphi^{\sigma_w}_{e\lambda} = 1$ if e supports a segment with wavelength λ in order to protect σ_w, and 0 otherwise. Note that in case of a protection segment σ_p shared by two node (except for the endpoints) and link disjoint working segments σ_w and $\sigma_{w'}$ with the same endpoints v_s and v_d, there will be an overall flow of value 2 (i.e., $\varphi^{\sigma_w}_{e\lambda} + \varphi^{\sigma_{w'}}_{e\lambda}$) from v_s to v_d on all links e of σ_p. Let us now study the mathematical formulation of the pricing problem for a given wavelength λ. In order to alleviate the notations, let us denote the flow variables by $\varphi^{\sigma_w}_e$.

a) Objective Function: The objective function of the λ pricing problem corresponds to the minimization of the reduced cost that is defined by:

$$\begin{aligned} \overline{B}_{C_\lambda} &= B_{C_\lambda} - u \cdot a^{C_\lambda} + u_0^\lambda \\ &= B_{C_\lambda} + u_0^\lambda - \sum_{\sigma_w \in \mathcal{S}^W} u_{\sigma_w} a^{C_\lambda}_{\sigma_w} \end{aligned} \qquad (4)$$

where $a^{C_\lambda}_{\sigma_w} = \sum_{e \in \omega^+(v_s(\sigma_w))} \varphi^{\sigma_w}_e$, and u_0^λ and u_{σ_w} are the dual variables associated respectively with constraints (1-λ) and (2-σ_w) of the master problem, see the previous section.

The number of transport blades used at v, B^λ_v, is the maximum number of transport ports between

- the number of protection segments $\sigma_p = (v, v')$ routed on λ and used as the first segment of a protection path originating at v and
- the number of protection segments $\sigma_p = (v', v)$ routed on λ and used as the last segment of a protection path terminating at v.

Those protection segments can be identified using the flow variables as follows:

$$B^{C_\lambda}_v \geq \sum_{e \in \omega^+(v)} \varphi^{\sigma_w}_e \qquad (5)$$

$$B^{C_\lambda}_v \geq \sum_{e \in \omega^-(v)} \varphi^{\sigma_w}_e. \qquad (6)$$

Note that the right-hand side of (5) (resp. (6)) evaluates the number of output (resp. input) ports at node v. The number of transport blades is then (over) estimated as follows: $B_{C_\lambda} = \sum_{v \in V} B^{C_\lambda}_v$. Note that $\sum_{C_\lambda \in \mathcal{C}} B_{C_\lambda} z_{C_\lambda}$ is only an over estimation of the number of blades as not all blades will be fully utilized on each wavelength: taking into account that an input port of

a blade is not used on λ_1 and that an output port is not used on λ_2 can result in the saving of one transport blade. Note that this issue cannot be solved by minimizing the number of ports instead of the number of blades. We will see in paragraph IV-B.3 how to modify the model in order to obtain an exact computation of the number of transport blades.

b) Constraints: Constraints of the pricing problem deal with the constraints associated with the definition of a proper wavelength protection configuration, they are as follows:

$$\sum_{e\in\omega^-(v)} \varphi_e^{\sigma_w} = \sum_{e\in\omega^+(v)} \varphi_e^{\sigma_w} \qquad \sigma_w \in \mathcal{S}^W, \; v \in V : v \notin \{v_s(\sigma_w), v_d(\sigma_w)\} \quad (7)$$

$$\sum_{e\in\omega^+(v_s(\sigma_w))} \varphi_e^{\sigma_w} = \sum_{e\in\omega^-(v_d(\sigma_w))} \varphi_e^{\sigma_w} \leq 1 \quad \sigma_w \in \mathcal{S}^W \quad (8)$$

$$\sum_{e\in\omega^-(v_s(\sigma_w))} \varphi_e^{\sigma_w} = \sum_{e\in\omega^+(v_d(\sigma_w))} \varphi_e^{\sigma_w} = 0 \quad \sigma_w \in \mathcal{S}^W \quad (9)$$

$$\varphi_e^{\sigma_w} + \varphi_e^{\sigma_{w'}} \leq 1 + \delta_{ww'} \qquad e \in L_\lambda; \sigma_w, \sigma_{w'} \in \mathcal{S}^W \quad (10)$$

$$\varphi_e^{\sigma_w} \in \{0,1\} \qquad e \in L_\lambda, e \notin \sigma_w, \sigma_w \in \mathcal{S}^W \quad (11)$$

$$\varphi_e^{\sigma_w} = 0 \qquad e \in (L \setminus L_\lambda) \cup \sigma_w, \sigma_w \in \mathcal{S}^W \quad (12)$$

Equation (7) corresponds to the flow conservation at intermediate nodes. Equation (8) expresses that the flow starting at $v_{s(\sigma_w)}$ finishes at $v_{d(\sigma_w)}$ while (9) means that no flow arrives at $v_{s(\sigma_w)}$ and none leaves from $v_{d(\sigma_w)}$. Equation (10) prevents two working segments in conflict (i.e., they share at least one fiber link) to be protected by the same protection segment. Equation (12) prevents a given link to be used both in a working segment and in its protection. Moreover, it forbids to use link e with the λ wavelength assignment in the definition of σ_p if (e, λ) is already included in a working segment.

3) Exact Computation of the Number of Transport Blades: Let us study how to modify the mathematical formulations of the last two paragraphs in order to obtain an exact computation of the number of transport blades. Recall that different wavelengths can be used on the input and output ports of a transport blade. So, in order to to calculate exactly the number of blades, we need to consider the maximum of the number of input and output ports. The pricing problem will identify potential configurations for each wavelength. Solving the master will lead to a solution, i.e., a BSSP protection scheme, defined by a selection of configurations, one for each wavelength.

We then get the following mathematical model for the master problem:

$$\min \sum_{v\in V} B_v$$

subject to:

$$\sum_{C\in\mathcal{C}^\lambda} z_C \leq 1 \qquad \lambda \in \Lambda \qquad (u_0^\lambda) \quad (13)$$

$$B_v \geq \sum_{C\in\mathcal{C}} B_v^{C,\text{OUT}} z_C \qquad v \in V \qquad (u_v^{\text{OUT}}) \quad (14)$$

$$B_v \geq \sum_{C\in\mathcal{C}} B_v^{C,\text{IN}} z_C \qquad v \in V \qquad (u_v^{\text{IN}}) \quad (15)$$

$$\sum_{\lambda\in\Lambda} \sum_{C\in\mathcal{C}^\lambda} a_{\sigma_w}^C z_C \geq 1 \qquad \sigma_w \in \mathcal{S}^W \qquad (u_{\sigma_w}) \quad (16)$$

$$z_C \in \{0,1\} \qquad C \in \mathcal{C}. \quad (17)$$

a) Objective Function - Pricing Problems: Again, there are as many pricing problems as the number of wavelengths. The objective function of the λ pricing problem corresponds to the minimization of the reduced cost:

$$\overline{B}_{C_\lambda} = \sum_{v\in V} (B_v^{C_\lambda^{\text{OUT}}} u_v^{\text{OUT}} + B_v^{C_\lambda^{\text{IN}}} u_v^{\text{IN}}) + u_0^\lambda - \sum_{\sigma_w\in\mathcal{S}^W} u_{\sigma_w} a_{\sigma_w}^{C_\lambda} \quad (18)$$

We differentiate the number of incoming ports $B_v^{C_\lambda^{\text{IN}}}$ from the number of outgoing ports $B_v^{C_\lambda^{\text{OUT}}}$ used at v in the λ pricing problem:

$$B_v^{C_\lambda^{\text{OUT}}} = \sum_{e\in\omega^+(v)} \varphi_e^{\sigma_w} \quad (19)$$

$$B_v^{C_\lambda^{\text{IN}}} = \sum_{e\in\omega^-(v)} \varphi_e^{\sigma_w}. \quad (20)$$

The other constraints are identical.

C. Solution of the BSSP Model

A key feature of column generation methods is that we do not need to solve exactly the pricing problem as long as we are able to design an efficient heuristic that quickly exhibits a column with a negative reduced cost, even though it is not the most negative one, it is enough in order to be able to iterate. Next, we do not suggest to solve exactly the column generation model that has been defined in the previous section, but to use it to design an efficient *global search* heuristic as in [18], although the model can be solved exactly for small to medium instances and therefore used to estimate the quality of the heuristic solutions.

Note also that each pricing problem is λ dependent, and therefore only a limited number of $\varphi_e^{\sigma_w}$ variables appear, i.e., $\varphi_e^{\sigma_w} = 0$ for all $e \in \sigma_w$ such that σ_w is supported on the λ wavelength. Therefore a possible direction in order to solve the pricing problem is to use an ILP package with the *lazy* constraint option (as in, e.g., CPLEX™) in order to introduce the working segments only as needed.

In order to obtain an integer solution for the master problem, one can easily get one using a rounding off method where after a variable has been fixed to an integer value, the optimal

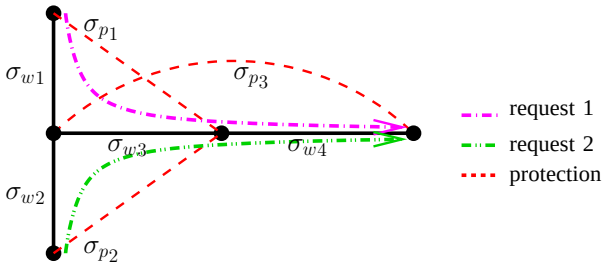


Fig. 6. protection by overlapping segments of a set of two requests

solution of the linear relaxation is updated using column generation. Various rounding off schemes should be investigated in order to identify the most successful one, i.e., selecting a variable either with respect to its fractional value or with respect to its contribution to the protection.

V. SSPO Protection Scheme

In this section we consider the SSPO protection scheme where a protection segment may span more than one working segment, see Figure 6 for an illustration.

A. The CG-SSPO Column Generation Model with Wavelength Configuration

1) Master problem: The master problem has a similar mathematical expression than for the BSSP protection scheme, except that the definition of the wavelength protection configurations differs. We will use *wavelength SSPO protection configurations*. Again, for a given wavelength λ, it is defined by a set of protection segments, all routed on λ, which protects a given set of working segments that are not necessarily routed on λ. The difference lies in the definition of the protection segments. For single hop working paths, they are the same as in the BSSP protection scheme: their endpoints coincide with those of the working segment, while they cannot share any link of the working segments they protect. Protection sharing is allowed, and encouraged as long as it helps to reduce the transport blade cost. For 2-hop working working paths, a SSPO protection is made of a single protection segment which has its two endpoints in common with those of the working path. For 3-hop working paths, a SSPO protection is either made of a single protection path (no difference with path protection) or of two protection segments that overlap over the second working segments, i.e., if the working path of request k is made of three segments $(\sigma_w, \sigma_{w'}, \sigma_{w''})$, the first protection segment may start at $v_s(\sigma_w)$ and end at $v_d(\sigma_{w'})$, while the second protection segment would start at $v_s(\sigma_{w'})$ and end at $v_d(\sigma_{w''})$. Notice that for both 2-hop and 3-hop working paths the working segment $\sigma_{w'}$ is automatically protected if the others are. It is why we do not have to add specific constraints for them.

The SSPO protection obliges to consider the protection of a working segment for a given request as shown on Figure 5(b), indeed the protection path and the working path no longer have the same endpoints. This induces the following modification in constraint (2):

$$\sum_{\lambda\in\Lambda}\sum_{C_\lambda\in\mathcal{C}} a^{C_\lambda}_{\sigma_w,k} z_{C_\lambda} \geq 1 \qquad k\in K, \sigma_w \in \mathcal{S}^k \qquad (21)$$

2) Pricing Problem:

a) Objective: Minimize the cost of the column :

$$\overline{B}_{C_\lambda} = B_{C_\lambda} - \sum_{k\in K}\sum_{\sigma_w\in\mathcal{S}^k} u_{\sigma_w,k} a^{C_\lambda}_{\sigma_w,k} + u^\lambda_0 \qquad (22)$$

where $u_{\sigma_w,k}$ is the dual variable associated with constraint (21-(k,σ_w)).

For 1 hop:

$$a^{C_\lambda}_{\sigma_w,k} = \sum_{e\in\omega^+(v_s(\sigma_w))} \varphi^{\sigma_w,k}_e$$

where $v_s(\sigma_w) = v^k_s$.

For 2 hops:

$$a^{C_\lambda}_{\sigma_w,k} = \sum_{e\in\omega^+(v_s(\sigma_w))} \varphi^{\sigma_w,k}_e \quad \text{if } v_s(\sigma_w) = v^k_s$$
$$a^{C_\lambda}_{\sigma_w,k} = \sum_{e\in\omega^-(v_d(\sigma_w))} \varphi^{\sigma_w,k}_e \quad \text{if } v_d(\sigma_w) = v^k_d.$$

For 3 hops $(\sigma_w, \sigma_{w'}, \sigma_{w''})$:

$$a^{C_\lambda}_{\sigma_w,k} = \sum_{e\in\omega^+(v_s(\sigma_w))} \varphi^{\sigma_w,k}_e \quad \text{if } v_s(\sigma_w) = v^k_s$$
$$a^{C_\lambda}_{\sigma_{w'},k} = a^{C_\lambda}_{\sigma_{w''},k}$$
$$a^{C_\lambda}_{\sigma_w,k} = \sum_{e\in\omega^-(v_d(\sigma_w))} \varphi^{\sigma_w,k}_e \quad \text{if } v_d(\sigma_w) = v^k_d.$$

Let

$$\mathcal{S}^W = \mathcal{S}^{W_1} \cup \mathcal{S}^{W_2} \cup \mathcal{S}^{W_3},$$

where $\mathcal{S}^{W_i} = \cup_{k\in K^i}(\mathcal{S}^W \cap \mathcal{S}^W_k)$ is the set of working segments of requests with i working segments (i.e., hops), $i = 1, 2, 3$ and $\mathcal{S}^W_k$ is the set of working segments for request k. Denote by v^k_s and v^k_d respectively the source and the destination node of the working path of request k.

b) Constraints: Let us now describe the set of constraints. As in the previous model, the protection segment associated with a working segment will be defined by an unit flow following the SSPO protection scheme. We therefore need to specify the connection index together with the working segment to be protected as there might be different protection segments associated with a given working segment depending on the requests. More formally, the protection of the working segment σ_w, with respect to request k (and such that σ_w is not the second working segment of request k), will be defined by the path described by the flow variables $\varphi^{\sigma_w,k}_e = 1$, for all $e \in L_\lambda$, where it is forbidden for the path to go through nodes which belong to any working segment of k, except for the endpoints of the working segments.

Consider the following example with two requests and six nodes. Let us assume that request k_1 is routed on a 3-hop path with segments σ_{w_1} from v_1 to v_2, σ_{w_2} from v_2 to v_4, σ_{w_3} from v_4 to v_5, and that request k_2 is routed on a 3-hop path with segments σ_{w_4} from v_3 to v_2, σ_{w_2} from v_2 to v_4, σ_{w_5} from v_4

to v_6. Following the SSPO scheme, we need four protection segments: σ_{p_1} from v_1 to v_4 and σ_{p_2} from v_2 to v_5 to ensure protection for k_1; σ_{p_3} from v_3 to v_4 and σ_{p_4} from v_2 to v_6 to ensure protection for k_2. We therefore have two flows going out of v_2, one for each request, both ensuring the protection of the segment σ_{w_2} and the protection of an additional segment. Same remark for the incoming flows at v_4.

$$\sum_{e\in\omega^-(v)} \varphi_e^{\sigma_w,k} = \sum_{e\in\omega^+(v)} \varphi_e^{\sigma_w,k} \quad v \in V \setminus \bigcup_{\sigma\in\mathcal{S}_k^W} \sigma, \sigma_w \in \mathcal{S}_k^W, k \in K \quad (23)$$

$$\sum_{e\in\omega^-(v)} \varphi_e^{\sigma_w,k} = \sum_{e\in\omega^+(v)} \varphi_e^{\sigma_w,k} = 0 \quad v \in \bigcup_{\sigma\in\mathcal{S}_k^W} \sigma : v \notin \bigcup_{\sigma\in\mathcal{S}_k^W} \{v_s(\sigma), v_d(\sigma)\}, \sigma_w \in \mathcal{S}_k^W, k \in K \quad (24)$$

$$\sum_{e\in\omega^+(v_s(\sigma_w))} \varphi_e^{\sigma_w,k} = \sum_{e\in\omega^-(v_d(\sigma_w))} \varphi_e^{\sigma_w,k} \leq 1 \quad \sigma_w \in K^1 \quad (25)$$

$$\sum_{e\in\omega^+(v_s(\sigma_w))} \varphi_e^{\sigma_w,k} = \sum_{e\in\omega^-(v_d(\sigma_{w'}))} \varphi_e^{\sigma_w,k} \leq 1 \quad (26)$$

$$\sum_{e\in\omega^-(v_d(\sigma_w))} \varphi_e^{\sigma_w,k} = \sum_{e\in\omega^+(v_s(\sigma_{w'}))} \varphi_e^{\sigma_w,k} = 0 \quad (27)$$

$$v_s(\sigma_w) = v_s^k, \{\sigma_w, \sigma_{w'}\} = \mathcal{S}_k^W, k \in K^2 \quad (28)$$

For 3-hop working paths, constraints (29) express that the protection segment starts at $v_s(\sigma_w)$ and ends at either $v_d(\sigma_{w'})$ or $v_d(\sigma_{w''})$, where $\sigma_{w'}$ is the second working segment and $\sigma_{w''}$ the third.

$$\sum_{e\in\omega^+(v_s(\sigma_w))} \varphi_e^{\sigma_w,k} = \sum_{e\in\omega^-(v_d(\sigma_{w'}))} \varphi_e^{\sigma_w,k} + \sum_{e\in\omega^-(v_d(\sigma_{w''}))} \varphi_e^{\sigma_w,k} \leq 1 \quad v_s(\sigma_w) = v_s^k, v_d(\sigma_{w''}) = v_d^k (\sigma_w, \sigma_{w'}, \sigma_{w''}) = \mathcal{S}_k^W, k \in K^3 \quad (29)$$

Constraints (30) applies for the third working segment $\sigma_{w''}$, expressing that the protection segment starts at either $v_s(\sigma_w)$ or $v_s(\sigma_{w'})$ and ends at $v_d(\sigma_{w''})$.

$$\sum_{e\in\omega^+(v_s(\sigma_w))} \varphi_e^{\sigma_w,k} + \sum_{e\in\omega^+(v_s(\sigma_{w'}))} \varphi_e^{\sigma_w,k} = \sum_{e\in\omega^-(v_d(\sigma_{w''}))} \varphi_e^{\sigma_w,k} \leq 1 \quad v_s(\sigma_w) = v_s^k, v_d(\sigma_{w''}) = v_d^k (\sigma_w, \sigma_{w'}, \sigma_{w''}) = \mathcal{S}_k^W, k \in K^3 \quad (30)$$

We do not need constraints for defining a protection for the second working segment of a 3-hop request, as it is protected by the same protection segment than the first or the last working segment.

Constraints (31) prevent from an useless use of intermediate nodes.

$$\sum_{e\in\omega^-(v_s(\sigma_w))} \varphi_e^{\sigma_w,k} = \sum_{e\in\omega^+(v_d(\sigma_{w''}))} \varphi_e^{\sigma_w,k} = 0 \quad v_s(\sigma_w) = v_s^k, v_d(\sigma_{w''}) = v_d^k (\sigma_w, \sigma_{w'}, \sigma_{w''}) = \mathcal{S}_k^W, k \in K^3 \quad (31)$$

$$\sum_{e\in\omega^-(v_s^k)} \varphi_e^{\sigma_w,k} = 0 \quad \sigma_w \in \mathcal{S}_k^W, k \in K \quad (32)$$

$$\sum_{e\in\omega^+(v_d^k)} \varphi_e^{\sigma_w,k} = 0 \quad \sigma_w \in \mathcal{S}_k^W, k \in K \quad (33)$$

$$\varphi_e^{\sigma_w,k} + \varphi_e^{\sigma_{w'},k'} \leq 1 + \delta_{ww'} \quad e \in L_\lambda; \sigma_w, \sigma_{w'} \in \mathcal{S}^W, k, k' \in K \quad (10')$$

$$\varphi_e^{\sigma_w,k} \in \{0,1\} \quad e \in L_\lambda \setminus (\bigcup_{\sigma_w\in\mathcal{S}_k^W} \sigma_w), \sigma_w \in \mathcal{S}^W, k \in K \quad (11')$$

$$\varphi_e^{\sigma_w,k} = 0 \quad e \in (L \setminus L_\lambda) \bigcup_{\sigma_w\in\mathcal{S}_k^W} \sigma_w, \sigma_w \in \mathcal{S}^W, k \in K \quad (12')$$

Constraints (23) correspond to the classical flow conservation constraints.

For single-hop working paths, constraints are the same than in the CG-BSSP model, i.e, corresponds to constraints (23) - (25), (32) - (33), (10'), (11') and (12') with the addition of constraints (24) in order to prevent the protection path to use a node of the working path.

For 2-hop working paths, the only possibility for a protection segment is to go from the source to the destination of the request, without going any node or link already involved in one of the working segments of the working path. Constraints (26) apply for the first working segment σ_w, expressing that the protection segment starts at $v_s(\sigma_w)$ and ends at $v_d(\sigma_{w'})$, where $\sigma_{w'}$ is the second working segment. Equation (27) forbid the use of the end node of the first working segment, to ensure its protection. We do not need constraints for the second working segment of a 2-hop request, as its is protected by the same protection segment than the first working segment.

Constraints (10'), (11') and (12 bid) are similar than in the pricing problem of the CG-BSSP column generation model, except that σ_w take value in all $\mathcal{S}_k^W$.

B. Solution of the SSPO Model

We can use the same techniques than those proposed for the BSSP model.

VI. Conclusion

In this paper we investigated a new segment protection scheme in WDM networks in the context of traffic grooming. The cost of the protection is measured through the overall number of ADMs required for the protection segments. We investigated two protection schemes assuming a sequential approach, where we define the protection framework once the working paths, made of a set of 1 to 3 working segments, has been defined using a given GRWA algorithm. The first protection scheme, called (BSSP), is such that each working segment is protected individually, while in the second one, called (SSPO), each working segment is protected simultaneously with others. Advantages of the SSPO scheme lie in that it protects the network against both node failure and fiber cut. We showed, using generic examples that none of the two protection technique dominates the other and we gave for both of them an ILP formulation using column generation. We are currently working on the implementation of these models techniques and performance results will be reported soon to better assess the efficiency of the SSPO scheme. Therefore, as equal or similar cost, the SSPO protection framework should be favored as it offers a better protection, i.e., offers protection against both node and link failure.

Acknowledgment

Work of B. Jaumard has been supported by a Concordia Research Chair on the Optimization of Communication Networks and by a NSERC grant. Work of F. Huc and D. Coudert has been partially supported by ANR-JC OSERA, INRIA, European project IST FET AEOLUS, COST 293 and Région Provence Alpes Côte d'Azur PACA.

References

[1] B. Mukherjee, *Optical WDM Networks*. Springer, 2006.

[2] R. Ramaswami and K. Sivarajan, *Optical Networks - A Practical Perspective*, 2nd ed. Morgan Kaufmann, 2002.

[3] R. Dutta and G. Rouskas, "A survey of virtual topology design algorithms for wavelength routed optical networks," *Optical Networks Magazine*, vol. 1, no. 1, pp. 73–89, January 2000.

[4] H. Zang, J. P. Jue, and B. Mukherjee, "A review of routing and wavelength assignment approaches for wavelength-routed optical WDM networks," *Optical Networks Magazine*, pp. 47–60, January 2000.

[5] B. Jaumard, C. Meyer, and B. Thiongane, "Decomposition methods for the RWA problem," submitted for Publication.

[6] ——, "ILP formulations for the RWA problem for symmetrical systems," in *Handbook for Optimization in Telecommunications*, P. Pardalos and M. Resende, Eds. Kluwer, 2006, ch. 23, pp. 637–678.

[7] R. Dutta and G. Rouskas, "Traffic grooming in WDM networks: Past and future," *IEEE Network*, vol. 16, no. 6, pp. 46–56, November/December 2002.

[8] E. Modiano and P. Lin, "Traffic grooming in WDM networks," *IEEE Communications Magazine*, vol. 39, no. 7, pp. 124–129, Jul. 2001.

[9] A. Somani, *Survivability and Traffic Grooming in WDM Optical Networks*. Cambridge Univwersity Press, Jan. 2006.

[10] K. Zhu and B. Mukherjee, "A review of traffic grooming in WDM optical networks: Architectures and challenges," *Optical Networks Magazine*, vol. 4, no. 2, pp. 55–64, March/April 2003.

[11] J. Hu and B. Leida, "Traffic grooming, routing, and wavelength assignment in optical WDM mesh networks," in *IEEE InfoCom*, 2004.

[12] S. Huang, R. Dutta, and G. N. Rouskas, "Traffic grooming in path, star, and tree networks: Complexity, bounds, and algorithms," *IEEE Journal on Selected Areas in Communications*, 2006.

[13] P.-H. Ho and H. T. Mouftah, "A framework for service-guaranteed shared protection in wdm mesh networks," *IEEE Communications Magazine*, pp. 97–103, February 2002.

[14] C. Barnhart, E. Johnson, G. Nemhauser, M. Savelsbergh, and P. Vance, "Branch-and-price: Column generation for solving huge integer programs," *Operations Research*, vol. 46, no. 3, pp. 316–329, May-June 1998.

[15] A. Bouffard, "Dimensionnement GRWA et protection par segment dans les rÃ©seaux optiques WDM," Master's thesis, Université de Montréal, Canada, 2005.

[16] L. Shen, X. Yang, and B. Ramamurthy, "Shared risk link group (srlg)-diverse path provisioning under hybrid service level agreements in wavelength-routed optical mesh networks," *IEEE/ACM Transactions on Networking, vol 13, pp. 918*, August 2005.

[17] T. Hemazro, B. Jaumard, and O. Marcotte, "A column generation and branch-and-cut algorithm for the channel assignment problem," *to appear in Computers and Operations Research*, 2007.

[18] B. Jaumard, B. Vignac, and F. Vanderbeck, "An efficient global search heuristic for the grwa proble," *submitted for publication*, 2007.

Annexe E

Reroutage de connexions et Pathwidth

E.1 A distributed algorithm for computing and updating the process number of a forest

Rapport de recherche, présenté à ALGOTEL08 puis à DISC08 en version étendue.

E.2 Pathwidth of Outerplanar Graphs

Article publié dans JGT.

E.3 On the Pathwidth of Planar Graphs

Article accepté pour publication dans Journal of Discrete Maths.

inria-00288304, version 3 - 27 Jun 2008

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A distributed algorithm for computing and updating the process number of a forest

David Coudert — Florian Huc — Dorian Mazauric

N° 6560

June 2008

Thème COM

Rapport de recherche

ISSN 0249-6399 ISRN INRIA/RR--6560--FR+ENG

inria-00288304, version 3 - 27 Jun 2008

inria-00288304, version 3 - 27 Jun 2008

A distributed algorithm for computing and updating the process number of a forest

David Coudert* , Florian Huc* , Dorian Mazauric*

Thème COM — Systèmes communicants
Projet MASCOTTE

Rapport de recherche n° 6560 — June 2008 — 16 pages

Abstract: In this paper, we present a distributed algorithm to compute various parameters of a tree such as the process number, the edge search number or the node search number and so the pathwidth. This algorithm requires n steps, an overall computation time of $O(n \log n)$, and n messages of size $\log_3 n + 3$. We then propose a distributed algorithm to update the process number (or the node search number, or the edge search number) of each component of a forest after adding or deleting an edge. This second algorithm requires $O(D)$ steps, an overall computation time of $O(D \log n)$, and $O(D)$ messages of size $\log_3 n + 3$, where D is the diameter of the modified connected component. Finally, we show how to extend our algorithms to trees and forests of unknown size using messages of less than $2\alpha + 4 + \varepsilon$ bits, where α is the parameter to be determined and $\varepsilon = 1$ for updates algorithms.

Key-words: pathwidth, process number, search number, distributed algorithm.

MASCOTTE, INRIA, I3S(CNRS/UNSA), Sophia Antipolis, France.
{firstname.lastname@sophia.inria.fr}

* This work was partially funded by the European projects IST FET AEOLUS and COST 293 GRAAL, and done within the CRC CORSO with France Telecom R&D.

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

inria-00288304, version 3 - 27 Jun 2008

Un algorithme distribué pour le calcul et la mise à jour du process number d'une forêt

Résumé : Dans cet article, nous présentons un algorithme distribué permettant de calculer divers paramètres d'un arbre tel le process number, la pathwidth et l'edge search number. Cet algorithme nécessite n étapes, a un temps d'exécution de $O(n \log n)$ et génère n messages de taille $\log_3 n + 3$. Nous montrons ensuite comment il peut servir a mettre à jour le process number (ou la pathwidth ou l'edge search number) de chaque composante d'un forêt après l'ajout ou la suppression d'une arête. En fin on montre que cela peut être fait même si la taille de la forêt est inconnue.

Mots-clés : pathwidth, process number, search number, algorithme distribué

inria-00288304, version 3 - 27 Jun 2008

1 Introduction

Treewidth and pathwidth have been introduced by Robertson and Seymour [11] as part of the graph minor project. By definition, the treewidth of a tree is one, but its pathwidth might be up to $\log n$. A linear time centralized algorithms to compute the pathwidth of a tree has been proposed in [5, 12, 13], but so far no dynamic algorithm exists.

The algorithmic counter part of the notion of pathwidth is the node searching problem [8]. It consists in finding an invisible and fast fugitive in a graph using the smallest set of agents. The minimun number of agents needed gives the pathwidth. Other graph invariants closely related to the notion of pathwidth have been proposed such as the process number [2, 3] and the edge search number [9]. For this two invariants it is not known if they are strictly equivalent to the pathwidth or not.

In this paper, we propose a dynamic algorithm to compute those different parameters on trees and to update them in a forest after the addition or deletion of an edge. We also show that no distributed algorithm can always transmit a number of bits linear in n and give a characterisation of the trees whose process number and edge search number equals their pathwidth. To present our results, we concentrate on the process number.

As mentioned before the process number of a (di)graph has been introduced to model a routing reconfiguration problem in WDM or WiFi networks in [2, 3]. The graph represents a set of tasks that have to be realized. A *process strategy* is a serie of actions in order to realize all the tasks represented by the graph. It finishes when all the nodes of the graph are *processed*. In order to process the graph, the three actions we can do are:

(1) put an agent on a node.

(2) remove an agent from a node if all its neighbors are either processed or occupied by an agent. The node is now processed.

(3) process a node if all its neighbors are occupied by an agent (the node is surrounded).

A *p-process strategy* is a strategy which process the graph using p agents. The *process number* of a graph G, $\mathrm{pn}(G)$, is the smallest p such that a p-process strategy exists. For example, a star has process number 1 (we place an agent on its center), a path of length at least 4 has process number 2, a cycle of size 5 or more has process number 3, and a $n \times n$ grid has process number $n+1$. Moreover, it has been proved in [2, 3] that $\mathrm{pw}(G) \leq \mathrm{pn}(G) \leq \mathrm{pw}(G)+1$, where $\mathrm{pw}(G)$ is the *pathwidth* of G [11].

The node search number [8], $\mathrm{ns}(G)$, can be defined similarly except that we only use rules (1) and (2). It was proved by Ellis *et al.* [5] that $\mathrm{ns}(G) = \mathrm{pw}(G)+1$, and by Kinnersley [7] that $\mathrm{pw}(G) = \mathrm{vs}(G)$, where $\mathrm{vs}(G)$ is the *vertex separation* of G. Those results show that the vertex separation, the node search number and the pathwidth are equivalent. Please refer to recent surveys [6, 4] for more information.

The following Theorem gives a construction which enforces each parameter to grow by 1, which implies that for any tree $\mathrm{ns}(T)$, $\mathrm{es}(T)$, $\mathrm{pw}(T)$, $\mathrm{vs}(T)$, and $\mathrm{pn}(T)$ are less than $\log_3(n)$.

Theorem 1 ([2] and [10]) *Let G_1, G_2 and G_3 be three connected graphs such that* $\mathrm{vs}(G_i) = vs$, $\mathrm{ns}(G_i) = ns$ *and* $\mathrm{pn}(\mathrm{G_i}) = \mathrm{p}$, $1 \leq i \leq 3$. *We construct the graph G by putting one copy of each of the G_i, and we add one node v that has exactly one neighbour in each of the G_i, $1 \leq i \leq 3$. Then* $\mathrm{vs}(G) = vs + 1$, $\mathrm{ns}(G) = ns + 1$ *and* $\mathrm{pn}(\mathrm{G}) = \mathrm{p} + 1$.

The algorithm we propose is based on the decomposition of a tree into subtrees forming a *hierarchical decomposition*. It is fully distributed, can be executed in an asynchronous environment and the construction of the hierarchical decomposition requires only a small amount of information.

It uses ideas similar to the ones used by Ellis *et al.* [5] to design an algorithm which computes the node search number in linear time. However their algorithm is centralized and the distributed version uses $O(n \log n)$ operations and transmit a total of $O(n \log n \log(\log n))$ bits. We improve the distributed version as our algorithm also requires $O(n \log n)$ operations but transmit at most $n(\log_3 n + 3)$ bits. We also prove that it is optimal in the sense that for any $k \in \mathbb{N}$, no dynamic algorithm, such that the vertex at which the edge addition/deletion is done, can only simultaneously sends one message to its neighbours, can always transmit less than $\frac{k-1}{k} n(\log_3(n))$ bits. Furthermore, with a small increase in the amount of transmitted information, we extend our algorithm to a fully dynamic algorithm allowing to add and remove edges even if the total size of the tree is unknown.

Finally we explain how to adapt our algorithm to compute the node search number and the edge search number of a tree. It should also certainly be adapted to compute the mixed search number and other similar parameters.

This paper start with the presentation of the hierarchical decomposition of a tree in Section 2. Then in Section 3 we present an algorithm to compute the process number of a tree and analyze its complexity. In Section 4 we show how to update efficiently the process number of each component of a forest after the addition or the deletion of any tree edge, thus resulting in a dynamic algorithm. Section 5 concludes this paper with several improvements including extensions of our algorithm to trees of unknown size and to compute other parameters.

All along this paper, we assume that each node u knows the set of its neighbours which we note $\Gamma(u)$. However, the size of the tree is not needed as explained in Section 5.

2 Tools for the algorithm

The algorithm is initialized at the leaves. Each leaf sends a message to its only neighbor which becomes its *father*. Then, a node v which has received messages from all its neighbors but one process them and sends a message to its last neighbor, its *father*. We say that this node has been *visited*. Finally, the last node, w, receives a message from all its neighbours and computes the process number of T: $\mathrm{pn}(\mathrm{T})$. w is called the *root* of T.

Notice that our algorithm is fully distributed, that it can be executed in an asynchronous environment (we assume that each node knows its neighbors) and that there are as many steps as nodes in the tree.

At each step, the goal of the message sent by a node v to its father v_0 is to describe, in a synthetic way, the structure of the *subtree T_v rooted at v*, that is the connected component of T minus the edge vv_0, $(T - vv_0)$, containing v (see Figure 1).

inria-00288304, version 3 - 27 Jun 2008

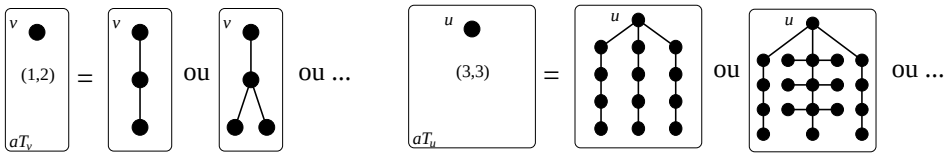


Figure 2: Example of trees whose associated vector are $vect(v) = (1,2)$ and $vect(u) = (3,3)$.

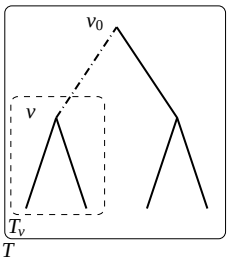


Figure 1: The subtree T_v

In fact a message describes a decomposition of T_v into a set of smaller disjoint trees. The trees of this decomposition are indexed by their roots; we note R_v the set of roots of the trees of this decomposition. Through the algorithm, given a node w, an unique tree with root w will be computed, i.e. if in two different decompositions there is a tree rooted at w, it will be the same. We call a tree of a decomposition with root w an *associated-tree* and note it aT^w.

An associated-tree, and more generally any tree, can be of two types: *stable* or *unstable*. Intuitively, the process number of a stable tree will not be affected if we add a component of same process number whereas the process number of an unstable tree will increase in this case.

Definition 1 *Let T be a tree with root r. T is said* stable *if there is an optimal process strategy such that the last (or equivalently first) node to have an agent is r or if there is a (≤ 2)-process strategy finishing with r. Otherwise T is* unstable. *The node r is said stable or unstable accordingly to T.*

Remark We consider a tree of process number one as stable (even if an optimal process strategy finishing at its root needs two agents) for technical reason.

From Definition 1, we give two values to describe if an associated-tree aT^w rooted at w is stable or unstable and to give its process number: pn its process number, and pn^+ the minimun number of agents used in a process strategy such that the last (or first) node to have an agent is w. They together formed the vector associated to aT^w: $vect(w) = (\text{pn}, \text{pn}^+)$. By extension we associate $vect(w)$ to w. Remark that they are unique for a given associated-tree but several associated-trees can have the same values, also they depend on the root of the associated-tree (see Figure 2). Remark also that to store this vector it is sufficient to store $(\text{pn}, \text{pn}^+ - \text{pn})$, which is an integer (pn) and a bit since $\text{pn} \leq \text{pn}^+ \leq \text{pn} + 1$.

Back to our algorithm, each associated-tree aT^w of the decomposition of T_v will be described by its vector $vect(w)$, and the message sent by a node v to its father v_0 contains the vector of all associated-trees of the decomposition. However if the decomposition does not verify some specific properties, this information is not sufficient to compute the process number of T_v. It is why we need the notion of hierarchical decomposition.

inria-00288304, version 3 - 27 Jun 2008

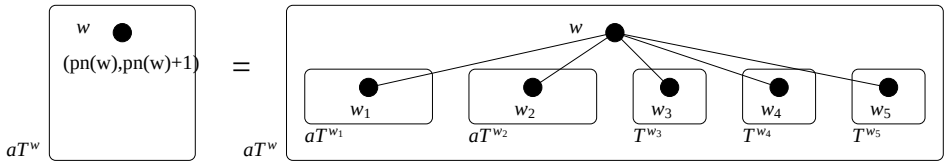


Figure 4: Structure of an unstable associated-tree aT^w. $vect(w_1) = vect(w_2) = (\mathrm{pn(w)}, \mathrm{pn(w)})$ and $\forall i \in [3,5], \mathrm{pn}(T^{w_i}) < \mathrm{pn(w)}$.

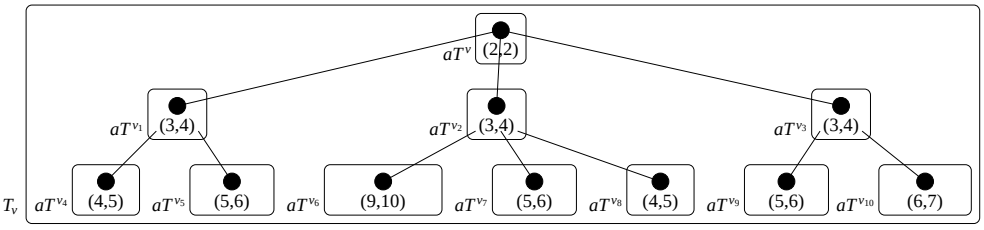


Figure 5: Example of a hierarchical decomposition of a tree T_v with process number 9.

2.1 Hierarchical decomposition

In a *hierarchical decomposition* of T_v, we impose that an associated-tree aT^w has a process number higher than the associated-tree aT^x containing the father of w, as illustrated in Figure 3. We also impose that a hierarchical decomposition has at most one stable associated-tree and if there is one it has to be minimal according to this order. Finally we impose that all unstable associated-trees satisfies Property 1. Figure 5 gives an example of a hierarchical decomposition of a tree with process number 9.

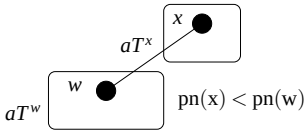


Figure 3: $aT^x < aT^w$.

Property 1 (c.f. Figure 4) *Given a node w, its associated-tree aT^w, the subtree T_w rooted at w, and $\Gamma(w) \cap T_w = \{w_1, \ldots, w_k\}$, if aT^w, and so w, is unstable it has the following structure: w has two neighbours $w_1, w_2 \in \Gamma(w) \cap T_w$ which are stables and such that* $\mathrm{pn}(w_1) = \mathrm{pn}(w_2) = \mathrm{pn}(w)$. *Furthermore aT^w is formed by its root w, the two stable associated-trees aT^{w_1} and aT^{w_2} and of $l \leq k-2$ other subtrees $T^{w_3}, \ldots, T^{w_{l+2}}$ whose roots are visited neighbours and whose process number is at most* $\mathrm{pn}(w) - 1$. *Notice that the subtrees $T^{w_3}, \ldots, T^{w_{l+2}}$ are not necessarily the associated-trees $aT^{w_3}, \ldots, aT^{w_{l+2}}$.*

To describe a given hierarchical decomposition, a node v stores a vector and a table encoding the shape of the associated-trees aT^v. We will see with Theorem 2 that it is sufficient to compute the process number of T_v. More precisely v stores:

- The vector of the stable associated-tree of the decomposition if there is one, $(-1,-1)$ otherwise;

inria-00288304, version 3 - 27 Jun 2008

- A table t_v of length $L(t_v) = \max_{w \in R_v}(\mathrm{pn(w)})$ which in cell i, noted $t_v[i]$, contains the number of unstable associated-trees whose vector is $(i, i+1)$ in the decomposition. (Remember that (1,2) is considered as stable, hence the first cell always contains 0).

For example in Figure 5, v and v_1 store respectively:

$HD(v)$: $t_v =$ | 0 | 0 | 3 | 2 | 3 | 1 | 0 | 0 | 1 | and $(\mathrm{pn(v)}, \mathrm{pn^+(v)}) = (2,2)$

$HD(v_1)$: $t_{v_1} =$ | 0 | 0 | 1 | 1 | 1 | and $(\mathrm{pn}(\mathrm{v}_1), \mathrm{pn}^+(\mathrm{v}_1)) = (-1,-1)$

Lemma 1 *Let $T = (V,E)$ be a tree rooted at r and aT^w, $r \notin aT^w$, an unstable associated-tree rooted at $w \in V$ in a hierarchical decomposition. If* $\mathrm{pn(aT^w)} = \mathrm{p}$, $\mathrm{pn(T)} = \mathrm{p}$ *iff* $\mathrm{pn(T \setminus aT^w)} \leq \mathrm{p}-1$.
Furthermore if $\mathrm{pn(T)} = \mathrm{p}$, *$T$ is unstable.*

Proof If there is a tree aT^x in the hierarchical decomposition with $\mathrm{pn(aT^x)} > \mathrm{p}$ then $\mathrm{pn(T \setminus aT^w)} > \mathrm{p}$. From now on we assume that for all aT^x of the hierarchical decomposition, $\mathrm{pn(aT^x)} \leq \mathrm{p}$. Using the properties of a hierarchical decomposition, it implies that w is the only node through which aT^w is connected to the rest of T.

By Property 1, aT^w is formed by its root w, two stable subtrees T^{w_1} and T^{w_2} with process number p and some other subtrees with process number less than $p-1$.

If $T \setminus aT^w$ has process number at least p then w is a node with three branches having process number at least p. Hence, by Theorem 1, T has process number at least $p+1$.

Otherwise $\mathrm{pn(T \setminus aT^w)} < \mathrm{p}$ and we describe a p-process strategy. We start by an optimal process strategy the stable associated-tree aT^{w_1}. It uses p agents and finishes with w_1 occupied by an agent. Then we place an agent on w and process w_1. We continue with an optimal process strategy of $T^w \setminus aT^{w_2}$, it uses at most $p-1$ extra agents.

Now, since $\mathrm{pn(T \setminus aT^w)} < \mathrm{p}$, we continue with a $(p-1)$-process strategy of $T \setminus aT^w$. We then place an agent on w_2 and process w. It now only remains to process aT^{w_2} starting at w_2 which can be done with p agents by assumption.

T is clearly unstable since it contains an unstable subtree aT^w with same process number which does not contain the root of T. □

Theorem 2 *Given a rooted tree T, a table t and a vector $vect = (\mathrm{pn}, \mathrm{pn}^+)$, if there is a hierarchical decomposition of T described by $(vect, t)$, we can compute* $\mathrm{pn(T)}$. *More precisely:*

a) $\mathrm{pn(T)} = \mathrm{L(t)} \Leftrightarrow \exists \mathrm{i} \in [1..\mathrm{L(t)}]$ *such that* $t[i] = 0$ *and* $\forall j \in [i+1..L(t)]\ t[j] = 1$. *Furthermore T is unstable.*

b) If $\mathrm{pn(T)} \neq \mathrm{L(t)}$ *then* $\mathrm{pn(T)} = \max\{\mathrm{pn}, \mathrm{L(t)}+1\}$ *and T is stable.*

The Property a) means that if in the table t of a hierarchical decomposition there is a cell with a 0 followed only by cells full of 1, then the process number of a tree accepting such a hierarchical decomposition has process number $L(t)$.

Proof of Theorem 2 First remark that the process number is at most $L(t)+1$.
By induction on $L(t)$.

inria-00288304, version 3 - 27 Jun 2008

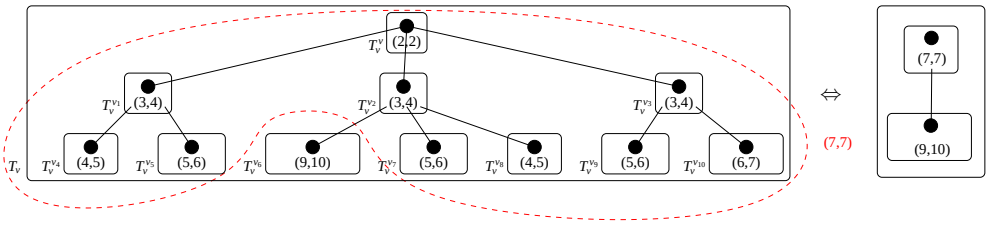


Figure 6: A simpler hierarchical decomposition of the example of Figure 5.

- If $L(t) = 0$, T is a single node and pn(T) = 0. If $L(t) = 1$, T is a stable tree with vector (1,1) or (1,2). In both case pn(T) = 1. If $L(t) = 2$ and $t[2] = 0$, T is a stable tree with vector (2,2) and pn(T) = 2. If $t[i] = 0$ for all $i \leq L(t)$, T is a stable tree with vector $(L(t), L(t))$ and pn(T) = L(t).
- When $L(t) \geq 2$ and $t[L(t)] = 1$. We call the associated-tree of the hierarchical decomposition having process number $L(t)$ aT^w and w its root. By Lemma 1, pn(T) = L(t) $\Leftrightarrow$ pn(T $\setminus$ aT^w) $\leq$ L(t) $-$ 1.
  - If $\exists i \in [1..L(t)]$ with $t[i] = 0$ and $\forall j \in [i+1..L(t)]$ $t[j] = 1$, we have pn(T $\setminus$ aT^w) $\leq$ L(t) $-$ 1.
    - Indeed, either $t[L(t)-1] = 1$ and pn(T $\setminus$ aT^w) = L(t) $-$ 1 by induction, so pn(T) = L(t).
    - Or $t[L(t)-1] = 0$. In this case either, we have a table with only 0 and we are at an initialisation case: pn(T $\setminus$ aT^w) = L(t) $-$ 1 or we can delete this last cell, the length of the table is then $L(t)-2$ and we are sure that pn(T $\setminus$ aT^w) $\leq$ L(t) $-$ 1 by the very first remark of the proof. In both cases we have once again pn(T) = L(t).
  - If in t there is a cell with a number bigger than one followed by cells full of one until the last cell, then, by induction, pn(T $\setminus$ aT^w) = L(t) and hence pn(T) = L(t) + 1.
- When $L(t) \geq 2$ and $t[L(t)] \geq 2$, we call one of the associated-tree of process number $L(t)$ aT^w and w its root. pn(T $\setminus$ aT^w) $\geq$ L(t), hence, from Lemma 1 pn(T) $>$ L(t) which means pn(T) = L(t) + 1 by the very first remark.

T stable or unstable follows from Lemma 1 and the process strategy we described. □

2.2 Minimal hierarchical decomposition

In the example of Figure 5, Theorem 2 directly says it has process number 9. If we now consider this example minus the subtree of vector $(9,10)$, then Theorem 2 says it has process number 7 and furthermore that it is stable. Hence, we can get another hierarchical decomposition as shown on Figure 6.

inria-00288304, version 3 - 27 Jun 2008

In fact we can generalize this simplification. Given a table t and an index $i \leq L(t)$, we note $t[1..i]$ the table composed of the i first cells of t. For a given hierarchical decomposition described by its vector and its table, $HD = (vect, t)$, we call $HD_i = (vect, t[1..i])$ a *i-restricted hierarchical decomposition*. Notice that if HD is a hierarchical decomposition of a tree T, then HD_i is a hierarchical decomposition of the subtree composed of the associated-trees having process number at most i.

A last definition, if a tree accepts several hierarchical decompositions, we say they are *equivalent*.

We now describe the simplification of a given hierarchical decomposition $HD = (vect, t)$ of a tree T. If there is $i \leq L(t)$ such that a tree T_i, whose hierarchical decomposition is described by $HD_i = (vect, t[1..i])$, has process number $i+1$, then HD is equivalent to a simpler hierarchical decomposition $HD' = ((i+1, i+1), t')$, where $L(t') = L(t)$, $t'[j] = 0$ for $j \leq i+1$, and $t'[j] = t[j]$ for $j > i+1$. If no such i exist, the hierarchical decomposition can not be simplified.

We call a hierarchical decomposition we can not simplify a *minimal hierarchical decomposition*. Our algorithm will compute such decompositions for each subtree T_v, $v \in V$. Furthermore we have:

Lemma 2 *Let $HD = ((\mathrm{pn}, \mathrm{pn}^+), \mathrm{t})$ be a minimal hierarchical decomposition. For all $i \in [2..L(t)]$, we have $t[i] \in \{0, 1\}$.*

3 Distributed algorithm for the process number

We can now describe precisely algorithm `algoHD`:

- The algorithm is initialized at the leaves. Each leaf sends the message $((0,0), [\,])$ (where $[\,]$ represents a table of length 0) to its only neighbour which becomes its father.
- A node v, which has received messages from all its neighbours but one, computes the minimal hierarchical decomposition of T_v using Algorithm 1. Then it sends $(\mathrm{pn}(\mathrm{T_v}), \mathrm{pn}^+(\mathrm{T_v})), \mathrm{t_v})$ to its last neighbour, its father.
- The last node w receives a message from all its neighbours, it computes the minimal hierarchical decomposition of $T_v = T$ and Theorem 2 gives the process number $\mathrm{pn}(\mathrm{T})$. w is called the root of T.

Remark It may happen that two adjacent nodes v and w receive a message from all their neighbors. It is the case when node v, after sending its message to its last neighbor w, receives a message from w. In this case, both v and w are potential candidates to be the root of the tree. There are two possibilities to solve this problem. If each node has a unique identifier (e.g. MAC address) known by its neighbors, then the one of v and w with the largest identifier becomes the root, otherwise, u and w send each other a random bit, repeat in case of equality, and the 1 win.

Lemma 3 *Given a tree $T = (V, E)$, with $|V| = n$, the time complexity of Algorithm 1 is $O(\log n)$.*

Proof All operations are linear in $L(t_v)$, and $L(t_v) \leq \mathrm{pn}(\mathrm{T}) \leq \log_3 \mathrm{n}$. □

inria-00288304, version 3 - 27 Jun 2008

Algorithm 1 Computation of the minimal hierarchical decomposition

Require: $v_1, ..., v_d$ the visited neighbours of v, and the corresponding minimal hierarchical decompositions $HD(v_i) = ((\mathrm{pn}(\mathrm{v_i}), \mathrm{pn}^+(\mathrm{v_i})), \mathrm{t_{v_i}})$

Require: t_v^{int}, a table such that $t_v^{int}[i] := t_{v_1}[i] + ... + t_{v_{d-1}}[i]$, $\forall i \in [2..\max_{1 \le j \le d} L(t_{v_j})]$.

Require: $M_v := \{v_i;\ \forall j \in [1..d-1],\ \mathrm{pn}(\mathrm{v_j}) \le \mathrm{pn}(\mathrm{v_i})\}$ {all v_i such that $\mathrm{pn}(\mathrm{v_i})$ is maximum}

Ensure: $vect(v)$ and t_v

{computation }

Let (p_v, p_v^+) be the vector of the associated-tree of v

if $\forall v_i \in M_v$, $\mathrm{pn}(\mathrm{v_i}) < 2$ **then** {Initial cases}

$$(p_v, p_v^+) := \begin{cases} (0,0) & \text{when } \forall v_i \in M_v,\ \mathrm{pn}(\mathrm{v_i}) = -1 \\ (1,1) & \text{when } \forall v_i \in M_v,\ \mathrm{pn}(\mathrm{v_i}) = 0 \\ (1,2) & \text{when } |M_v| = 1 \text{ and } vect(v_i) = (1,1) \\ (2,2) & \text{otherwise} \end{cases}$$

else {general cases}

  if $|M_v| = 2$ **then** {v is unstable}

    $(p_v, p_v^+) := (\mathrm{pn}(\mathrm{v_i}), \mathrm{pn}(\mathrm{v_i}) + 1)$, where $v_i \in M_v$

  else {v is stable}

    if $|M_v| > 2$ **then** {Theorem 1}

      $(p_v, p_v^+) := (\mathrm{pn}(\mathrm{v_i}) + 1, \mathrm{pn}(\mathrm{v_i}) + 1)$, where $v_i \in M_v$

    else

      $(p_v, p_v^+) := (\mathrm{pn}(\mathrm{v_i}), \mathrm{pn}(\mathrm{v_i}))$, where $v_i \in M_v$

{computation of the table}

$L(t_v) := \max\{L(t_v^{int}), p_v\}$

$t_v := t_v^{int}$

if $p_v < p_v^+$ and $p_v > 1$ **then**

  $t_v[p_v] := t_v[p_v] + 1$

  $t_v[j] := 0$, $\forall j \in [2..p_v - 1]$

  $(p_v, p_v^+) := (-1, -1)$ {Here, (p_v, p_v^+) is stable}

Let k be such that $t_v[k] > 1$ and $t_v[i] \le 1$, $\forall i \in [k+1..L(t_v)]$

Let k_1 be such that $t_v[k_1] = 0$ and $t_v[i] = 1$, $\forall i \in [k..k_1 - 1]$

if $t_v[p_v] = 0$ **then**

  $k_2 := p_v$

else

  Let k_2 be such that $t_v[k_2] = 0$ and $t_v[i] > 0$, $\forall i \in [p_v..k_2 - 1]$ {We assume that there exists a virtual cell $t_v[L(t_v) + 1] = 0$}

if k, k_1 and k_2 exist **then**

  $t_v[i] := 0$, $\forall i \in [2..\max(k_1, k_2)] := 0$

  $vect(v) := (\max(k_1, k_2), \max(k_1, k_2))$

else {the hierarchical decomposition is already minimal}

  $vect(v) := (p_v, p_v^+)$

inria-00288304, version 3 - 27 Jun 2008

Lemma 4 *Given a tree $T = (V,E)$, with $|V| = n$, algo HD computes* pn(T) *in n steps and overall $O(n \log n)$ operations.*

Proof Each node v of degree d_v has to compute M_v (the set of neighbors v_i with maximum pn(v_i)) which requires $O(d_v)$ operations, and t_v^{sum} (the sum of all received tables) that is $O(\sum_1^d L(t_v^{sum}))$ operations. Finally it applies Algorithm 1. As $\sum_{v \in V} d_v = 2(n-1)$, we have $\sum_{v \in V}(d_v + \log n + \sum_1^d L(t_v^{sum})) = O(n \log n)$. □

Lemma 5 *Given a tree $T = (V,E)$, with $|V| = n$, algoHD sends $n-1$ messages each of size* $\log_3 n + 2$.

Proof Node v sends its minimal hierarchical decomposition to its father, that is $HD_v = (vect(v), t_v)$, with $vect(v) = (\text{pn(v)}, \text{pn}^+\text{(v)})$. From Theorem 1 we know that $L(t_v) \leq \log_3 n$, from Lemma 2, t_v contains only 0 and 1's, hence we need only $\log_3 n$ bits to transmit t_v. Furthermore, if pn(v) ≥ 1, $t_v[\text{pn(v)}] = 0$ and $\forall i \leq \text{pn(v)}, \text{t}_\text{v}[\text{i}] = 0$. Hence we can add an artificial 1 to the cell of t_v with index pn(v) to indicate the value pn(v).

To summarize, we transmit a table t and two bits ab. $ab = 00$ means $vect(v) = (-1,-1)$, $ab = 01$ means $vect(v) = (0,0)$, 10 means $vect(v) = (\text{pn}, \text{pn})$ and 11 means $vect(v) = (\text{pn}, \text{pn}+1)$. When $a = 1$, pn is the index of the first 1 in the transmitted table and t_v is the transmitted table minus this 1. When $a = 0$, t_v is the transmitted table t. It is clear that in this coding, each message has size $\log_3 n + 2$. □

4 Dynamic and incremental algorithms

In this section, we propose a dynamic algorithm that allows to compute the process number of the tree resulting of the addition of an edge between two trees. It also allows to delete any edge. To do this efficiently, it uses one of the main advantage of the hierarchical decomposition: the possibility to change the root of the tree without additional information (Lemma 6). From that we design an incremental algorithm that computes the process number of a tree.

If we want to join two trees with an edge between their roots then it is easy to see that Algorithm 1 will do it. However if we do not join them through the root, a preprocessing to change the root of the trees needs to be done. In next Section we propose one. To apply this algorithm, each node needs to store the information received from each of its neighbors and a table which is the sum of the received tables: $\forall v_i \in \Gamma(v) \cap T_v : vect_{v_i}, t_{v_i}$ and t_v^{sum}. Recall that t_v^{sum} is defined as $t_v^{sum}[j] = \sum_{v_i \in \Gamma(v) \cap T_v} t_{v_i}[j]$ in the algorithm.

For a given tree T, we note $D(T)$ or D if there is no ambiguity the *diameter* of T.

We describe now three functions we will use in the dynamic version of our algorithm.

4.1 Functions for updating the process number

Lemma 6 (Change of the root) *Given a tree $T = (V,E)$ rooted at $r_1 \in V$ of diameter D, and its hierarchical decomposition, we can choose a new root $r_2 \in V$ and update accordingly the hierarchical decomposition in $O(D)$ steps of time complexity $O(\log n)$ each, using $O(D)$ messages of size* $\log n + 3$.

inria-00288304, version 3 - 27 Jun 2008

Proof We describe an algorithm to change the root from r_1 to r_2:

First, r_2 sends a message to r_1 through the unique path between r_1 and r_2, $r_2 = u_0, u_1, u_2, \ldots, u_k = r_1$, to notify the change. Then, r_1 computes its hierarchical decomposition, considering that u_{k-1} is its father. We assume that each node v stores the information received from its neighbours and t_v^{sum}. r_1 applies Algorithm 1 using all vectors stored but $vect_{u_{k-1}}$ and $t_v^{sum} - t_{v_{k-1}}$. Then it sends a message to u_{k-1}.

After, u_{k-1} computes its hierarchical decomposition, considering that u_{k-2} is its father, and sends a message to u_{k-2}. We repeat until r_2 receives a message from u_1. Finally, r_2 computes the process number of T and becomes the new root. We have a new hierarchical decomposition.

In this algorithm, u_i substracts the table $t_{u_{i-1}}$ from $t_{u_i}^{sum}$, and later adds $t_{u_{i+1}}$, computes M_{u_i} and finally applies Algorithm 1. Clearly, all computation requires $O(\log n)$ operations. The messages need one more bit than in the previous algorithm to indicate whether a table has to be added or substracted. □

Lemma 7 (Addition of an edge) *Given two trees $T_{r_1} = (V_1, E_1)$ and $T_{r_2} = (V_2, E_2)$ respectively rooted at r_1 and r_2, we can add the edge $(w_1, w_2), w_1 \in V_1$ and $w_2 \in V_2$ and compute the process number of $T = (V_1 \cup V_2, E_1 \cup E_2 \cup (w_1, w_2))$, in at most D steps.*

Proof First we change the roots of T_{r_1} and T_{r_2} respectively to w_1 and w_2 using Lemma 6. Then, w_1 and w_2 decide of a root (see Remark 3) which finally computes the process number of T. □

Lemma 8 (Deletion of an edge) *Given a tree $T = (V, E)$ rooted at r and an edge $(w_1, w_2) \in E$, after the deletion of edge (w_1, w_2), we can compute the process number of the two disconnected trees in at most D steps.*

Proof W.l.o.g. we may assume that w_2 is the father of w_1. Let T_{w_1} be the subtree rooted at w_1 and $T \setminus T_{w_1}$ the tree rooted at r. Remark that it includes w_2. The process number of T_{w_1} is deduced from the previously computed hierarchical decomposition. Now, to compute the process number of $T \setminus T_{w_1}$, we apply the change root algorithm and node w_2 becomes the new root of $T \setminus T_{w_1}$. □

4.2 Incremental algorithm

From Lemma 7, we obtain an incremental algorithm (`IncHD`) that, starting from a forest of n disconnected vertices with hierarchical decomposition $((0,0,)[\,])$, add tree edges one by one in any order and updates the process number of each connected component. At the end, we obtain the process number of T.

This algorithm is difficult to analyze in average, but the best and worst cases are straightforward:

- Worst case: T consists of two subtrees of size $n/3$ and process number $\log_3(n/3)$ linked via a path of length $n/3$. Edges are inserted alternatively in each opposite subtrees. Thus `IncHD` requires $O(n^2)$ steps and messages, and overall $O(n^2 \log n)$ operations

inria-00288304, version 3 - 27 Jun 2008

- Best case: edges are inserted in the order induced by `algoHD` (inverse order of a breadth first search). `IncHD` needs $O(n)$ messages and an overall of $O(n \log n)$ operations.

Actually, the overall number of messages is $O(nD)$ and the number of operations is $O(nD\mathrm{pn(T)})$. They both strongly dependent on the order of insertion of the edges. Thus an interesting question is to determined the average number of messages and operations.

5 Improvements and extensions

Reducing the amount of transmitted information In our algorithms, it is possible to reduce the size of some messages and so the overall amount of information transmitted during the algorithm. For example, instead of transmitting $\log n$ bits for t, we may transmit only $L(t)$ bits plus the value $L(t)$ on $\log\log n$ bits. Overall we will exchange less than $n(\mathrm{pn(T)} + \log_2 \log_3 \mathrm{n} + 2 + \varepsilon)$ bits, where $\varepsilon = 1$ for the dynamic version of the algorithm (`IncHD`). Further improvements are possible with respect to the following lemma.

Lemma 9 *Assuming that when an edge is added at vertex v, v asks its neighbours information once and simultaneously, any dynamic algorithm satisfying this assumption induces a transmission of at least* $\frac{k-1}{k} n(\mathrm{pn(T)} - 2)$ *bits for any* $k \in \mathbb{N}$ *and value of* $\mathrm{pn(T)} \leq \log_3(\mathrm{n/k})$ *in some trees* T.

Proof Suppose that we are given a dynamic algorithm such that when an edge is added at vertex v, v asks its neighbours information once and simultaneously, and let $k > 1$ be an integer. We consider a tree made of a path u-v of length $\frac{k-1}{k} n$ with a tree T' at u. One of the messages received by v gives information about T'. If for all tree T' with process number p, the algorithm uses less than $p-2$ bits to encode this message, and since there is more than 2^{p-2} hierarchical decompositions corresponding to a tree with process number p, there exists two trees T'_1 and T'_2 with different minimal hierarchical decompositions but which are encoded in the same way. We note T_1 when $T' = T'_1$ and T_2 when $T' = T'_2$. Then, it exists a tree T" such that if we join it to (w.l.o.g) T_1 at v, the process number of T_1 increases by one whereas if we join T" to T_2 at v, the process number of T_2 does not increase.

Hence, there is a tree T' for which the algorithm encodes the information transmitted to v on at least $p-2$ bits. For this T' in our construction of T, the information received by v comes from u and hence it has transited through $\frac{k-1}{k} n$ nodes. Therefore, the total of transmitted bits is at least $\frac{k-1}{k} n(p-2)$. □

Corollary 1 *Assuming that when an edge is added at vertex v, v asks its neighbours information once and simultaneously, any dynamic algorithm induces a transmition of at least* $\frac{k-1}{k} n(\log_3 n)$ *bits in some large enough trees, for any* $k \in \mathbb{N}$.

Proof Let $k \in \mathbb{N}$. By the previous Lemma for $k+1$, there is a tree T with process number $\log_3(n/(k+1))$ which induces a transmition of at least $\frac{k}{k+1} n(\log_3(n/(k+1)) - 2)$ bits, and this larger than $\frac{k-1}{k} n(\log_3 n)$ when $\log n > k^2(\log_3(k+1) + 2)$. □

inria-00288304, version 3 - 27 Jun 2008

Reducing the number of operations It makes no doubt that the worst case complexity of `IncHD` and more specifically of Lemma 7 can be seriously improved. In particular, instead of changing the roots of both trees, we may change only r_1 to w_1, then transmit information in the direction of r_2, and eventually stop the transmissions before r_2 if the minimal hierarchical decomposition of some node remains unchanged.

It is also interesting to notice that using arguments similar to [5], we can get a centralized algorithm using a linear number of operations.

Trees and forests of unknown size If the size n of the tree is unknown, a node encodes each bit of the transmitted table t on 2 bits, that is 00 for 0 and 01 for 1. It allows to use 11 to code the end of the table and hence to know its length. Thus the receiver may decode the information without knowing n. In this coding the table requires $2L(t)+2$ bits and the transmission requires $2L(t)+4+\varepsilon$ bits, where $\varepsilon = 1$ for `IncHD` and 0 for `algoHD`. Remember that $L(t) \leq \mathrm{pn(T)}$.

Computing other parameters Our algorithms can be adapted to compute the node search number or the pathwidth of any tree with the same time complexity and transmission of information. For that, it is sufficient to change the values of the initial cases (lines 1 and 1) in Algorithm 1.

For the node search number we would use the initial cases of the left of Figure 5. Notice that in this case we do not use the vector $(1,2)$.

if $\forall v_i \in M_v$, $\mathrm{pn(v_i)} < 2$ **then**

$$(p_v, p_v^+) := \begin{cases} (1,1) & \text{when } \forall v_i \in M_v,\ \mathrm{pn(v_i)} = -1 \\ (2,2) & \text{otherwise} \end{cases}$$

if $\forall v_i \in \Gamma(v)$, $\mathrm{pn^+(v_i)} < 2$ **then**

$$(p_v, p_v^+) := \begin{cases} (0,0) & \text{when } |M_v| = 0 \\ (1,1) & \text{when } |M_v| = 1 \\ (1,2) & \text{when } |M_v| = 2 \\ (2,2) & \text{otherwise} \end{cases}$$

Figure 7: Initial cases for node search number (left) and edge search number (right).

For the edge search number of a tree, we can prove that $\mathrm{ns}(T) - 1 \leq \mathrm{es}(T) \leq \mathrm{ns}(T)$, whereas on a general graph we only have $\mathrm{ns}(T) - 1 \leq \mathrm{es}(T) \leq \mathrm{ns}(T) + 1$. To adapt Algorithm 1 for the edge search number, we would use the initial cases of the right of Figure 5 plus the extra rule that all received vectors $(1,2)$ are interpreted as if they were vectors $(2,2)$. Also, if all received vectors verifies $\mathrm{pn^+(v_i)} < 2$, M_v is the set of all received vectors different from (-1,-1). Notice that it gives the first algorithm to compute the edge search number of trees.

Algorithm `algoHD` has been implemented for the process number, the node search number and the edge search number, as well as corresponding search strategies [1].

About the difference of the parameters Finally, the following lemma characterizes the trees for which the process number (resp. edge search number) equals the pathwidth.

Lemma 10 *Given a tree T, $\mathrm{pn(T)} = \mathrm{pw(T)} + 1 = \mathrm{p} + 1$ (resp. $\mathrm{pn(T)} = \mathrm{es(T)} + 1 = \mathrm{p} + 1$) iff there is a node v such that any components of $T - \{v\}$ has pathwidth at most p and there is at least three components with process number (resp. edge search number) p of which at most two have pathwidth p.*

inria-00288304, version 3 - 27 Jun 2008

This lemma means that the difference between, e.g., the process number and the pathwidth comes from the difference on trees with smaller parameter and ultimately from trees with those parameters equal to 1 or 2.

To give such characterisations for more general classes of graphs remains a challenging problem.

Acknowledgments

We would like to thanks Nicolas Nisse and Hervé Rivano for fruitfull discussions on this problem.

References

[1] `http://www-sop.inria.fr/members/Dorian.Mazauric/Capture/index.php.htm`.

[2] D. Coudert, S. Perennes, Q.-C. Pham, and J.-S. Sereni. Rerouting requests in wdm networks. In *AlgoTel'05*, pages 17–20, Presqu'île de Giens, France, mai 2005.

[3] D. Coudert and J-S. Sereni. Characterization of graphs and digraphs with small process number. Research Report 6285, INRIA, September 2007.

[4] J. Díaz, J. Petit, and M. Serna. A survey on graph layout problems. *ACM Computing Surveys*, 34(3):313–356, 2002.

[5] J.A. Ellis, I.H. Sudborough, and J.S. Turner. The vertex separation and search number of a graph. *Information and Computation*, 113(1):50–79, 1994.

[6] F. V. Fomin and D. Thilikos. An annotated bibliography on guaranteed graph searching. *Theoretical Computer Science, Special Issue on Graph Searching*, 2008, to appear.

[7] N. G. Kinnersley. The vertex separation number of a graph equals its pathwidth. *Inform. Process. Lett.*, 42(6):345–350, 1992.

[8] M. Kirousis and C.H. Papadimitriou. Searching and pebbling. *Theor. Comput. Sci.*, 47(2):205–218, 1986.

[9] N. Megiddo, S. L. Hakimi, M. R. Garey, D. S. Johnson, and C. H. Papadimitriou. The complexity of searching a graph. *J. Assoc. Comput. Mach.*, 35(1):18–44, 1988.

[10] T. D. Parsons. Pursuit-evasion in a graph. In *Theory and applications of graphs*, pages 426–441. Lecture Notes in Math., Vol. 642. Springer, Berlin, 1978.

[11] N. Robertson and P. D. Seymour. Graph minors. I. Excluding a forest. *J. Combin. Theory Ser. B*, 35(1):39–61, 1983.

[12] P. Scheffler. A linear algorithm for the pathwidth of trees. In R. Henn R. Bodendiek, editor, *Topics in Combinatorics and Graph Theory*, pages 613–620. Physica-Verlag Heidelberg, 1990.

inria-00288304, version 3 - 27 Jun 2008

[13] K. Skodinis. Construction of linear tree-layouts which are optimal with respect to vertex separation in linear time. *J. Algorithms*, 47(1):40–59, 2003.

inria-00288304, version 3 - 27 Jun 2008

inria-00288304, version 3 - 27 Jun 2008

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399

Pathwidth of outerplanar graphs *

David Coudert, Florian Huc and Jean-Sébastien Sereni
MASCOTTE, I3S-CNRS–INRIA–UNSA
2004 route des Lucioles – BP93
06902 Sophia Antipolis, FRANCE
E-mail: lastname@sophia.inria.fr

Abstract

We are interested in the relation between the pathwidth of a biconnected outerplanar graph and the pathwidth of its (geometric) dual. Bodlaender and Fomin [3], after having proved that the pathwidth of every biconnected outerplanar graph is always at most twice the pathwidth of its (geometric) dual plus two, conjectured that there exists a constant c such that the pathwidth of every biconnected outerplanar graph is at most c plus the pathwidth of its dual. They also conjectured that this was actually true with c being one for every biconnected planar graph. Fomin [10] proved that the second conjecture is true for all planar triangulations. First, we construct for each $p \geq 1$ a biconnected outerplanar graph of pathwidth $2p+1$ whose (geometric) dual has pathwidth $p+1$, thereby disproving both conjectures. Next, we also disprove two other conjectures (one of Bodlaender and Fomin [3], implied by one of Fomin [10]). Finally we prove, in an algorithmic way, that the pathwidth of every biconnected outerplanar graph is at most twice the pathwidth of its (geometric) dual minus one. A tight interval for the studied relation is therefore obtained, and we show that all cases in the interval happen.

1 Introduction

A *planar graph* is a graph that can be embedded in the plane without crossing edges. It is said to be *outerplanar* if it can be embedded in the plane without crossing edges and such that all its vertices are incident to the unbounded face. For any graph G, we denote by $V(G)$ its vertex set and by $E(G)$ its edge set. The *dual* of the planar graph G, denoted by G^*, is the graph obtained by putting one vertex for each face, and joining two vertices if and only if the corresponding faces are adjacent. The *weak dual* of G, denoted by $\mathcal{T}_G$, is the induced subgraph of G^* obtained by removing the vertex corresponding to the unbounded face. As is well known, the weak dual of an outerplanar graph is a forest, and the weak dual of a biconnected outerplanar graph is a tree. Furthermore, linear-time algorithms to recognise and embed outerplanar graphs are known (see for instance [15, 21]). Note that the dual of a planar graph can also be computed in linear-time.

The notion of pathwidth was introduced by Robertson and Seymour [17]. A *path decomposition* of a graph $G = (V,E)$ is a set system $(X_1, \ldots, X_r)$ of V such that

(i) $\bigcup_{i=1}^{r} X_i = V$;

(ii) $\forall xy \in E, \exists i \in \{1,2,\ldots,r\} : \{x,y\} \subset X_i$;

*This work was partially funded by the European projects IST FET AEOLUS and COST 293 GRAAL, and done within the CRC CORSO with France Telecom R&D.

(iii) $\forall (i_0,i_1,i_2) \in \{1,2,\ldots,r\}^3, i_0 < i_1 < i_2 \Rightarrow X_{i_0} \cap X_{i_2} \subseteq X_{i_1}$.

The *width* of the path decomposition $(X_1,\ldots,X_r)$ is $\max_{1\leq i\leq r} |X_i| - 1$. The *pathwidth* of G, denoted by $\mathrm{pw}(G)$, is the minimum width over its path decompositions.

The pathwidth of a graph was shown to be equal to its vertex separation [12]: a *layout* (or *vertex-ordering*) L of a graph $G = (V,E)$ is a one-to-one correspondence between V and $\{1,\ldots,|V|\}$. The *vertex separation of* (G,L) is $\max_{1\leq i\leq |V|} |M(i)|$ where

$$M(i) := \{v \in V : L(v) > i \text{ and } \exists u \in N(v) : L(u) \leq i\}.$$

The *vertex separation of* G, denoted by $\mathrm{vs}(G)$, is the minimum of the vertex separation of (G,L) taken over all vertex-orderings L.

Computing the pathwidth of graphs is an active research area, in which a lot of work has been done (survey papers are for instance [6, 2, 16]). It was shown [4] that the pathwidth of graphs with bounded treewidth can be computed in polynomial time. As outerplanar graphs have treewidth two, the pathwidth of an outerplanar graph is polynomially computable. However, the exponent in the running time of the algorithm is rather large, so the algorithm is not useful in practice. This is why Govindan et al. [11] gave an $O(n\log(n))$ time algorithm for approximating the pathwidth of outerplanar graphs with a multiplicative factor of three. For biconnected outerplanar graphs, Bodlaender and Fomin [3] improved upon this result by giving a linear-time algorithm which approximates the pathwidth of biconnected outerplanar graphs with a multiplicative factor two (and a corresponding path decomposition is obtained in time $O(n\log(n))$). To do so, they exhibited a relationship between the pathwidth of an outerplanar graph and the pathwidth of its dual. More precisely, the following holds.

Theorem 1 (Bodlaender and Fomin [3]) *Let G be a biconnected outerplanar graph without loops and multiple edges. Then* $\mathrm{pw}(G^*) \leq \mathrm{pw}(G) \leq 2\mathrm{pw}(G^*) + 2$.

Observe that adding a vertex linked to all other vertices of any graph increases its pathwidth by exactly one. Since the weak dual of an outerplanar graph (which can be computed in linear-time) is a tree and there exist linear-time algorithms to compute the pathwidth of a tree [8], this yields the desired approximation (obtaining a corresponding path decomposition needs more work).

Bodlaender and Fomin [3] suggested that a stronger relationship holds between the pathwidth of a planar graph and the pathwidth of its dual.

Conjecture 1 (Bodlaender and Fomin [3]) *There is a constant c such that for every biconnected outerplanar graph G without loops and multiple edges* $\mathrm{pw}(G) \leq \mathrm{pw}(G^*) + c$.

Conjecture 2 (Bodlaender and Fomin [3]) *For every biconnected planar graph G without loops and multiple edges,* $\mathrm{pw}(G) \leq \mathrm{pw}(G^*) + 1$.

Fomin [10] proved that if G is any biconnected planar graph of maximum degree at most three, then $\mathrm{pw}(G) \geq \mathrm{pw}(G^*) - 1$. This implies that Conjecture 2 is true for every planar triangulation, since any planar triangulation is the dual of a biconnected planar graph of maximum degree three.

It is worth noting that these conjectures are motivated by the following result about the treewidth, conjectured by Robertson and Seymour [18] and proved by Lapoire [13] using algebraic methods (notice that Bouchitté, Mazoit and Todinca [5] gave a shorter and combinatorial proof of this result).

Theorem 2 (Lapoire [13]) *For every planar graph G,* $\mathrm{tw}(G) \leq \mathrm{tw}(G^*)$.

In Section 2, we exhibit a family $(G_p)_{p\geq 1}$ of biconnected outerplanar graphs with maximum degree four such that $\mathrm{pw}(G_p) = 2p+1$ and $\mathrm{pw}(G_p^*) = p+1$, thereby disproving both conjectures. To construct these graphs, we introduce a general construction which actually allows us to prove the following result.

Theorem 3 *For every integer $p \geq 1$ and every integer $k \in \{1,2,\ldots,p+1\}$, there exists a biconnected outerplanar graph of pathwidth $p+k$ whose weak dual has pathwidth p.*

Let us mention here that Fomin and Thilikos [9] also disproved, independently, Conjecture 1.

Next, we disprove the following conjecture of Bodlaender and Fomin [3].

Conjecture 3 (Bodlaender and Fomin [3]) *For every simple 2-connected planar graph G,* $\mathrm{pw}(G) \geq \mathrm{pw}(G^*) - 1$.

In Section 3, we prove the following result which improves the upper bound given by Theorem 1.

Theorem 4 *Let G be a biconnected outerplanar graph without loops and multiple edges. Then* $\mathrm{pw}(G) \leq 2\mathrm{pw}(G^*) - 1$.

As a consequence, the previous approximation for the pathwidth of biconnected outerplanar graphs is also improved. We give an algorithmic proof which allows to obtain a layout of the outerplanar graph G considered (and whose vertex separation is hence at most $2\mathrm{pw}(G)-1$).

Furthermore, Theorem 3 shows that this bound is best possible in general.

2 Counter-examples

In this section, we establish Theorem 3 and deduce the following corollary which disproves Conjectures 1 and 2.

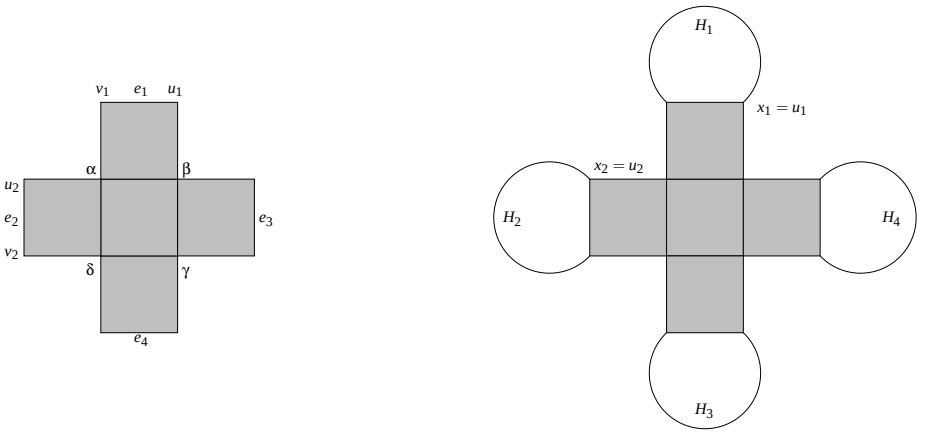
Corollary 1 *For every integer $p \geq 1$, there exists a triangle-free biconnected outerplanar graph G_p of maximum degree four whose pathwidth is $2p+1$ such that the pathwidth of its dual is $p+1$.*

For each $i \in \{1,2,3,4\}$, let H_i be a biconnected outerplanar graph of pathwidth p whose weak dual has pathwidth p'. We shall describe a construction which yields a biconnected outerplanar graph $\mathcal{C}(H_1,H_2,H_3,H_4)$ of pathwidth $p+2$ whose weak dual has pathwidth $p'+1$. This construction will be illustrated by examples yielding the graphs G_p of Corollary 1.

A 4-cycle is called a *square*. Two squares are *adjacent* if they share exactly one edge. The *degree* of the square S is the number of squares adjacent to S. Let the *cross* K be the biconnected outerplanar graph consisting of four squares of degree one and one square of degree four (see Figure 1(a)).

For each $i \in \{1,2,3,4\}$, let x_iy_i be an edge of H_i incident to the unbounded face in an outerplanar embedding of H_i. This edge is chosen such that there exists an optimal layout L of $\mathcal{T}_{H_i}$ where the vertex v corresponding to the the bounded face incident to x_iy_i fulfils $L(v) = |V(\mathcal{T}_{H_i})|$ for $i \in \{2,3\}$. Notice that this is always possible, and directly follows from Theorem 6 cited in Section 3. For $i \in \{1,2\}$, we denote by L_i an optimal layout of H_i, i.e. a layout with vertex separation p, and without loss of generality we assume that $L_i(x_i) < L_i(y_i)$.

Consider the cross K of Figure 1(a). For each $i \in \{1,2,3,4\}$, the edge e_i of K is identified with the edge x_iy_i. We assume moreover that the vertices x_1 and x_2 are identified with the vertices u_1 and u_2 respectively (see Figure 1(b)). Notice that there is generally not a unique way to achieve this



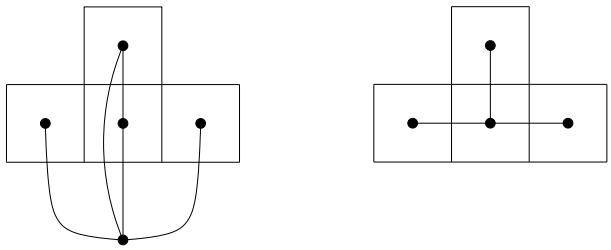
(a) The cross K.

(b) Gluing four graphs H_1, H_2, H_3, H_4 on the cross K

Figure 1: When identifying the edges, we ensure that x_1 is identified with u_1 and x_2 with u_2.

construction, but we shall denote by $\mathcal{C}(H_1, H_2, H_3, H_4)$ any graph obtained from H_1, H_2, H_3, H_4 in this way.

It is clear by the construction that any such graph $\mathcal{C}(H_1, H_2, H_3, H_4)$ is a biconnected outerplanar graph. As an example, let G_1 be the biconnected outerplanar graph consisting of three squares of degree one and one square of degree three (see Figure 2). For any integer $p \geq 2$, let G_p be the graph $\mathcal{C}(G_{p-1}, G_{p-1}, G_{p-1}, G_{p-1})$, obtained as indicated in Figures 3 and 4. Remark that the condition on the vertices x_1 and x_2 is clearly fulfilled in this case thanks to the symmetry of the graphs G_p, and that the maximum degree of G_p is four.



(a) G_1 and G_1^*

(b) G_1 and $\mathcal{T}_{G_1}$

Figure 2: G_1, graph consisting of one square of degree three and three squares of degree one, the dual G_1^* and the weak dual $\mathcal{T}_{G_1}$, a star.

In the following three lemmata, we prove the announced properties of the construction. The central square of the cross is denoted by S, and the corresponding vertex of the dual is s.

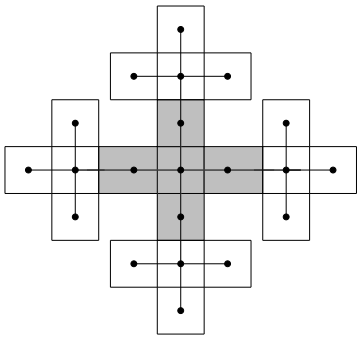


Figure 3: G_2, four disjoint copies of G_1 glued with a grey cross K, and its weak dual $\mathcal{T}_{G_2}$.

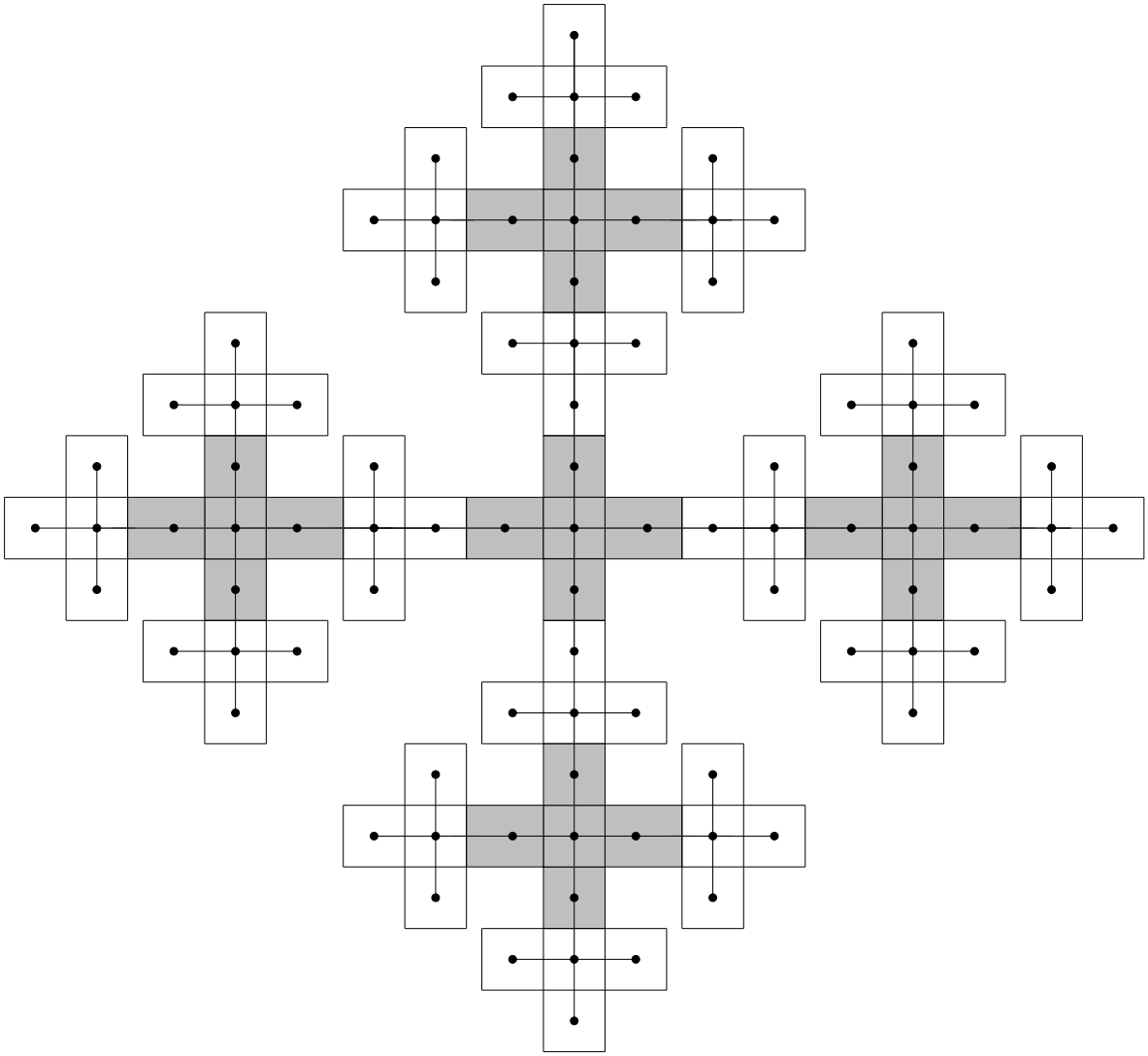


Figure 4: G_3, four disjoint copies of G_2 glued with a grey cross K, and its weak dual $\mathcal{T}_{G_3}$.

Lemma 1 *For each $i \in \{1,2,3,4\}$, let H_i be a biconnected outerplanar graph whose weak dual T_i has pathwidth $p \geq 1$. The pathwidth of the weak dual of graph $\mathcal{C}(H_1,H_2,H_3,H_4)$ is $p+1$.*

We introduce the following definition: for every vertex v of a tree T, a *branch at v* is any maximal subtree which contains a neighbour of v without containing v. The following result will be useful to prove Lemma 1.

Theorem 5 (Scheffler [19]) *For every integer $p \geq 1$ and every tree T, $\mathrm{pw}(T) \geq p+1$ if and only if there exists a vertex t of T with at least three branches of pathwidth at least p.*

Proof of Lemma 1. By induction on $p \geq 1$, the result being true for $p = 1$. If the pathwidth of each tree T_i is p, it is not difficult to construct a layout L of the weak dual of $\mathcal{C}(H_1,H_2,H_3,H_4)$ with vertex separation $p+1$: first, label the vertices of T_i according to an optimal layout of T_1. Then, label the vertex s_1, i.e. define $L(s_1) := |V(T_1)|+1$. Next, label the vertices of T_2 (from $L(s_1)+1$ up to $L(s_1)+|V(T_2)|$) according to an optimal layout of T_2 such that the unique neighbour of s_2 in T_2 is given the biggest integer (such a layout exists by the construction). The next vertex to be labelled is s_2, and an analoguous labeling is done for the vertices of T_3 and s_3. Finally, the vertex s est labelled, then the vertices of T_4 (according to an optimal layout of T_4) and last the vertex s_3.

Also, the pathwidth of the weak dual of $\mathcal{C}(H_1,H_2,H_3,H_4)$ is more than p by Theorem 5 since the vertex s has four branches with pathwidth p. □

Lemma 2 *For each $i \in \{1,2,3,4\}$, let H_i be a biconnected outerplanar graph of pathwidth $p \geq 1$. The vertex separation of the graph $\mathcal{C}(H_1,H_2,H_3,H_4)$ is at least $p+2$.*

Proof. Consider any layout L of $H := \mathcal{C}(H_1,H_2,H_3,H_4)$. We shall prove that the vertex separation of (H,L) is at least $p+2$. The subgraph of H induced by removing the vertices of the square S is the disjoint union of the four graphs H_1,H_2,H_3 and H_4, each of them having pathwidth p. Note that the roles played by those four graphs in this proof are symmetric. Assume that the vertex a such that $L(a) = 1$ and the vertex b such that $L(b) = |V(H)|$ are in $V(H_1) \cup V(S)$ and $V(H_1) \cup V(H_2) \cup V(S)$ respectively. By hypothesis, there exists $i \in L(V(H_4))$ such that there are p vertices x of H_4 with $L(x) > i$, each having a neighbour y in H_4 with $L(y) \leq i$. As a similar integer exists for H_3, we suppose without loss of generality that there exists a vertex $v \in V(H_3)$ with $L(v) > i$. Let $X := \cup_{j=1}^{3} V(H_j) \cup V(S)$. Say that a vertex $x \in X$ is an *m-vertex* if $L(x) < i$ and an *M-vertex* if $L(x) > i$. In particular, a is an m-vertex and b and v are M-vertices. An edge is *bad* if it links an m-vertex to an M-vertex. A *bad pair* is a pair of bad edges that are either disjoint, or incident to the same m-vertex. Denote by Q the subgraph of G induced by X, and note that the existence of a bad pair in Q implies that $\mathrm{vs}(H,L) \geq \mathrm{vs}(H_4)+2 = p+2$.

Remark now that Q is 2-connected, so according to the Fan Lemma, there exists in Q two paths P_1 and P_2 respectively from a to b and from a to v, which are vertex-disjoint except in a. Note that P_1 and P_2 are both disjoint from H_4. As a in an m-vertex and b and v are M-vertices, there exists a bad edge on P_1 and a bad edge on P_2, which necessarily from a bad pair (since the only common vertex of P_1 and P_2 is a, an m-vertex). Therefore, the vertex separation of (H,L) is at least $p+2$. □

Lemma 3 *For each $i \in \{1,2,3,4\}$, let H_i be a biconnected outerplanar graph of pathwidth $p \geq 1$. The pathwidth of $\mathcal{C}(H_1,H_2,H_3,H_4)$ is at most $p+2$.*

Proof. We shall construct a layout of $\mathcal{C}(H_1,H_2,H_3,H_4)$ from optimal layouts L_i of H_i, $i \in \{1,2,3,4\}$. Start by labelling all the vertices of H_4 according to L_4. The vertex separation never exceeds $p+2$,

since the only unlabelled vertices not in H_4 that might have labelled neighbours are β and γ. By the construction, the optimal layout L_1 of H_1 can be chosen such that $L_1(x_1) < L_1(y_1)$. Label the vertices of H_1 until the vertex x_1 is labelled. As previously, the vertex separation does not exceed $p+2$ when doing so. Now, label the vertex β, which does not change the vertex separation, as β has exactly one unlabelled vertex, α. Now go on labelling the vertices of H_1 according to L_1. The vertex-separation still does not exceed $p+2$, the only unlabelled vertices not in H_1 with labelled neighbours being α and γ. By the construction again, the layout L_2 of H_2 can be chosen such that $L_2(x_2) < L_2(y_2)$. Therefore we can apply the same procedure to label the vertices of H_2: first label them until x_2 is labelled, then label the vertex α and finish labelling the vertices of H_2. At last, label the vertices of H_3 (the vertex separation does not exceed $p+2$ when doing so, since the only unlabelled vertices with labelled neighbours not in H_3 are δ and γ), and then label the vertices δ and γ. The obtained layout has vertex separation at most $p+2$. □

Proof of Theorem 3. The proof is by induction on $p \geq 1$. If $p = 1$, two adjacent squares and G_1 give the desired result when $k = 1$ and $k = 2$ respectively.

Suppose that the result is true for $p-1 \geq 1$, and let $k \in \{1,2,\ldots,p+1\}$. First, let $k = 1$: as is well known, there exist biconnected outerplanar graphs of pathwidth $p+1$ whose weak dual has pathwidth p. If $k \in \{2,3,\ldots,p+1\}$, then $k-1 \in \{1,2,\ldots,p\}$ so, by the induction hypothesis, there exists a biconnected outerplanar H of pathwidth $(p-1)+(k-1)$ whose weak dual has pathwidth $p-1$. Then by Lemmata 1, 2 and 3, $\mathcal{C}(H,H,H,H)$ has pathwidth $p+k$ and its weak dual has pathwidth p, as desired. □

Now we show how the family $(G_p)_{p\geq 1}$ can be used to also disprove Conjecture 3.

First, observe that the pathwidth of a multigraph G is equal to the pathwidth of its underlying simple graph, denoted $\mathcal{U}(G)$.

Let uv be an edge of a 2-connected planar graph G. Denote F_1 and F_2 the two faces incident to uv, and f_1 and f_2 the corresponding vertices of G^*. To subdivide i times the edge uv (i.e. to replace it by an induced path of length $i+1$) leads to replace the edge $f_1 f_2$ in G^* by $i+1$ parallel edges.

Now, consider G_p and an embedding of G_p^* such that $G_p^{**} \simeq G_p$. Call o the vertex corresponding to the external face of G_p in G_p^*. Let H_p be the grah obtained by subdividing each edge of G_p^* incident to o (i.e. by replacing it by an induced path of length two). Notice that H_p is a simple 2-connected planar graph. According to the preceding remarks, $\mathcal{U}(H_p^*) \simeq G_p$, and hence $\mathrm{pw}(H_p^*) = \mathrm{pw}(G_p)$.

For every face F of G_p, let $m(F)$ be the number of edges of F incident to the unbounded face. Let $m := \max_F(m(F))$.

Lemma 4 $\mathrm{pw}(H_p) \leq \mathrm{pw}(G_p^*) + m = \mathrm{pw}(G_p^*) + 3$

Proof. It is clear by the definition of G_p that $m = 3$. Let l be an optimal layout of G_p^* (i.e. a layout of minimum vertex-separation), and let construct a layout of H_p of width at most $\mathrm{pw}(G^*) + m$. Let us label every vertex v belonging to both H_p and G_p^* with $(l(v),0)$. Now, if the vertex labelled $(l(v),0)$ has j unlabelled neighbours, label them $(l(v),1),\cdots,(l(v),j)$. The obtained labeling surely has vertex separation at most $\mathrm{pw}(G_p^*) + m$. □

If Conjecture 2 is true, then $\mathrm{pw}(H_p) \geq \mathrm{pw}(H_p^*) - 1 = \mathrm{pw}(G_p) - 1$. However, by lemma 4, $\mathrm{pw}(H_p) \leq \mathrm{pw}(G_p^*) + 3$ and so $\mathrm{pw}(G_p^*) + 3 \geq \mathrm{pw}(G_p) - 1$ which is false for $p > 6$.

3 Upper bound

We shall present in this section an algorithm which, given a biconnected outerplanar graph G, computes a layout of G with vertex separation at most $2\mathrm{pw}(\mathcal{T}_G)+1$. As $\mathrm{pw}(\mathcal{T}_G) = \mathrm{pw}(G^*)-1$ for any biconnected outerplanar G (see [3]), this establishes Theorem 4.

First, recall that a *caterpillar* is a tree in which a single path, the *spine*, is incident to (or contains) every edge. The caterpillars are the only trees of pathwidth one: every caterpillar has surely pathwidth one, and if a tree T is not a caterpillar, then it contains a spider with three legs of length two (see Figure 5). But such a tree has pathwidth at least two by Theorem 5.

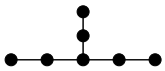


Figure 5: A spider with three legs of length two.

Proposition 1 *Let G be a biconnected outerplanar graph whose weak dual is a caterpillar. Then G has pathwidth at most three.*

Proof. Here is a layout of the vertices of G with vertex separation at most three. Let $P := v_1 v_2 \dots v_k$ be a longest path of $\mathcal{T}_G$. Denote by F_i the face of G corresponding to the vertex v_i, $i \in \{1,2,\dots,k\}$. Label by 1 a vertex v of F_1 of degree two (such a vertex exists as G is outerplanar and v_1 is a leaf of T). Then recursively label every vertex of F_1 of degree two which is adjacent to a labelled vertex.

Now, apply the following procedure in which we suppose that $V(F_{i-1}) \cap V(F_i) = \{x_i, y_i\}$ and $V(F_i) \cap V(F_{i+1}) = \{x_{i+1}, y_{i+1}\}$, see Figure 6.

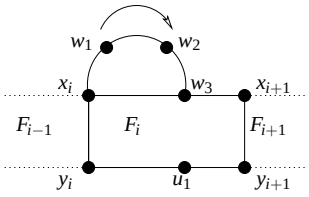


Figure 6: Vertices for step i.

1: **for** $i = 2$ to $k-1$ **do**
2: let $P := x_i w_1 \dots w_j x_{i+1}$ be the path of G from x_i to x_{i+1} consisting of edges incident to the unbounded face. Label the vertices of P from x_i to w_k
3: let $P' := y_i u_1 \dots u_t y_{i+1}$ be the path of G from y_i to y_{i+1} consisting of edges incident to the unbounded face. Label the vertices of P' from y_i to u_t
4: **end for**

Last, label the vertices clockwise from x_k to y_k.

The obtained layout surely has vertex separation at most three. □

The procedure given in the preceding proof actually achieves an optimal layout of the corresponding graph. Indeed, such a graph has pathwidth two if its weak dual is path, and pathwidth three otherwise. Note also that the time complexity of the procedure is linear.

We will use the following result about the pathwidth of trees.

Theorem 6 (Ellis, Sudborough and Turner [8]) *For any tree T, and any integer $p \geq 2$, $\mathrm{pw}(T) \leq p$ if and only if there is a path P such that every connected component of the forest induced by the vertices of $V(T) \setminus V(P)$ has pathwidth at most $p-1$.*

We consider the recursive procedure given by Algorithm 1. It computes a layout of G stored in the list l, which is initialised by $l(v) := \infty$ for every vertex $v \in V(G)$ (this means that all vertices are unlabelled at the beginning).

Notice that what is done in lines 16–17 and 32–33 is equivalent to label all the vertices of H except y (or y' respectively), and to keep s updated.

The following lemma suffices to establish Theorem 4.

Lemma 5 *For any biconnected outerplanar graph G whose weak dual T has pathwidth p, the procedure* `Layout` *of Algorithm 1 returns a layout with vertex separation at most $2p+1$.*

Proof. Algorithm 1 clearly assigns a unique label to every vertex of G.

For the vertex separation of the obtained layout, the proof is by induction on the pathwidth p of T. If p is one, then T is a caterpillar and Proposition 1 gives the conclusion.

Suppose now that for every biconnected outerplanar graph whose weak dual has pathwidth at most $p-1 \geq 1$, the procedure `Layout` of Algorithm 1 returns a layout with vertex separation at most $2p-1$. Let us prove that the obtained layout for G has pathwidth at most $2p+1$.

Stop the labelling of G at any moment and denote by F the set of unlabelled vertices with a labelled neighbour. If no subgraph H has been labelled yet, then the set F consists of x and x', so its size is at most $2p+1$. If a subgraph H has just been labelled, then F consists of two vertices, namely x' and y or x and y'.

Suppose now that a subgraph H is being labelled. Without loss of generality, say that its intersection with the current face F_i is $\{x,y\}$. There is only one vertex of F not in H, namely x'. Therefore, if $|F \cap V(H)| \leq 2p$ we have $|F| \leq 2p+1$ as wanted. As the vertex separation of the layout used to label H is at most $2p-1$, the only problem that might occur is if $|F \cap (V(H) \setminus \{x,y\})| = 2p-1$, and x, x' and y also belong to F. This implies that y was requested to be labelled in the original layout l used for H, but kept unlabelled as indicated in the algorithm. But in this case, in the labelling l of H, the vertex x is unlabelled, and has at least a labelled neighbour, y. So the number of unlabelled vertices of H with a labelled neighbour in H is $|F \cap (V(H) \setminus \{x,y\})| + 1 = 2p$, a contradiction. □

As one can see in the preceding proof, the subgraphs H, labelled in lines 16 and 32, can actually be labelled by any layout with vertex separation at most $2p-1$.

Corollary 2 *For any biconnected outerplanar graph G, $\mathrm{pw}(\mathcal{T}_G)+1 \leq \mathrm{pw}(G) \leq 2\mathrm{pw}(\mathcal{T}_G)+1$. Furthermore the bounds are tight.*

As proved in [19], the pathwidth of a tree with f vertices is less than $\log_3(2f+1)$. Thus we have the following corollary.

Corollary 3 *The pathwidth of any biconnected outerplanar graph G with f inner faces is less than $2\log_3(2f+1)+1$.*

Proposition 2 *The time complexity of Algorithm 1 is $O(n\log(n))$.*

Algorithm 1 Procedure `Layout`

Require: a biconnected outerplanar graph G, a list l and an integer s.

Ensure: returns the integer j, which is one more than the biggest label used. Every vertex v of G is given a unique label, stored in $l(v)$.

if the weak dual T of G is a caterpillar **then**
  label it according to Proposition 1 and starting with the label s.
  return $s+|V(G)|$.
end if
Compute a path $P := v_1v_2\dots v_k$ of T fulfilling the property of Theorem 6, with the additional property that its endvertices are leaves. Denote by F_i the face of G corresponding to the vertex v_i of P, $i \in \{1,2,\dots k\}$.
Let v be a vertex of degree two of the face F_1, and denote by x and x' its clockwise and counter-clockwise neighbours respectively {note that such a vertex always exists}
$l(v) := s$
$s := s+1$
for $i = 1$ to k **do** {throughout the following, y and y' respectively denote the clockwise neighbour of x and the counter-clockwise neighbour of x' on F_i}
  while $x \notin V(F_{i+1})$ **do**
    if x has at most one unlabelled neighbour different from x' **then**
      $l(x) := s$
      $s := s+1$
    else
      let H be the maximal biconnected subgraph of G whose intersection with F_i is $\{x,y\}$.
      $s := \texttt{Layout}(H,l,s)$
      $l(y) := \infty$
    end if
    if $y == x'$ **then**
      $l(x') := s$
      return $s+1$
    else
      $x := y$
    end if
  end while
  while $x' \notin V(F_{i+1})$ **do**
    if x' has at most one unlabelled neighbour (different from x) **then**
      $l(x') := s$
      $s := s+1$
    else
      let H be the maximal biconnected subgraph of G whose intersection with F_i is $\{x,y\}$.
      $s := \texttt{Layout}(H,l,s)$
      $l(y') := \infty$
    end if
    $x' := y'$
  end while
end for

Proof. It is easy to see that the time complexity of Algorithm 1 depends mainly on the recursive calls and on the time complexity of line 5 (since computing the weak dual of a biconnected outerplanar graph and determining whether a tree is a caterpillar is linear in time, as is the procedure of Proposition 1). We first show that the time complexity of Algorithm 1 without line 5 is linear.

For that, remark that a node x of face F_i is labelled directly during the processing of face F_i if it has at most one unlabelled neighbour different from x', otherwise during the recursive call, or it will be considered again during the processing of face F_{i+1}. So a node x is considered once in each inner face to which it belongs, that is its degree minus one. So altogether we have $2(|E|-|V|)$ steps, which is equal to $2(f-1)$ using Euler's formula for planar graphs, where f is the total number of inner faces. Since the number of faces of a biconnected outerplanar graph is smaller that its number of vertices, the time complexity of Algorithm 1 without line 5 is linear.

The computation of a path P fulfilling the property of Theorem 6, with the additional property that its endvertices are leaves, is similar in style to the techniques used in [8, 22, 14] on trees to compute vertex separation, cutwidth and search number. Thus it can be done in linear-time. Furthermore, the pathwidth of a tree with f vertices being less than $\log_3(2f+1)$ [19], the computation of all paths takes time $O(f\log_3(2f+1))$, that is $O(n\log(n))$. □

Theorem 4 clearly provides a linear-time algorithm to approximate the pathwidth of a biconnected outerplanar graph G since computing the dual tree of G and its pathwidth can both be done in linear-time. A corresponding layout is given by Algorithm 1, whose time complexity is $O(n\log(n))$. As noted in [3], there exist trees and outerplanar graphs for which a straight representation of a layout needs $\Omega(n\log(n))$ in time just to be written. Skodinis [20] developed a representation so that path decompositions (and layouts) can be written in linear-time. We did not try to use it for Algorithm 1 but we suspect that it can be used to precompute all paths in linear-time and thus reduce the complexity to $O(n)$. For unicyclic graphs, which in particular are outerplanar, Ellis and Markov [7] gave an algorithm that computes the vertex separation along with a corresponding linear layout in time $O(n\log(n))$.

Corollary 4 *For any biconnected outerplanar graph G, Algorithm 1 provides in time $O(n\log(n))$ a layout of G with vertex separation at most* $\mathrm{pw}(G)-1$.

4 Conclusion

We strengthened the previously known relation between the pathwidth of a biconnected outerplanar graph and the pathwidth of its dual. We did so in an algorithmic way and thus obtained a new approximation algorithm. We established the tightness of our bound, thereby disproving two of conjectures of Bodlaender and Fomin [3, 10], and moreover we showed that all cases in the interval happen.

To conclude, we note here that, according to [10], Conjecture 3 is implied by another conjecture of Fomin [10]. We need two new definitions to state it.

Given an edge-ordering σ of $G=(V,E)$, let $\delta(i)$ be the number of vertices incident to at least two edges e,e' such that $\sigma(e)\leq i$ and $\sigma(e')>i$. The *linear width of* (G,σ) is the maximum of $\delta(i), i\in\{1,2,\ldots,|E|\}$. The *linear width of* G, denoted by $\mathrm{lw}(G)$, is the minimum of the linear width of (G,σ) taken over all the edge-orderings σ. Notice that if G has minimum degree at least two, then $\mathrm{pw}(G)\leq\mathrm{lw}(G)\leq\mathrm{pw}(G)+1$. For a planar graph G, a *split* H of G is a graph obtained by a sequence of the following operations: take a vertex v, partition its neighbourhood in two sets M and N, replace v by two new vertices x,y. Link x to $M\cup\{y\}$ and y to N.

Conjecture 4 (Fomin [10]) *For every planar graph G, there exists a planar split H of maximum degree three such that* $\mathrm{lw}(H) = \mathrm{lw}(G)$.

As Conjecture 3 is disproved in Section 2, Conjecture 4 does not hold.

We end with a question. Fomin and Thilikos [9] proved that, for every 3-connected planar graph G, the pathwidth of G^* is at most 6 times the pathwidth of G. Amini, Huc and Pérennes [1] showed that this bound can be reduced to three (and even to two if G is 4-connected).

Problem 1 *Is there a constant c such that, for every 2-connected planar graph G,*

$$\frac{1}{2}\mathrm{pw}(G^*) - c \leq \mathrm{pw}(G) \leq 2\mathrm{pw}(G^*) + c?$$

If the answer is positive, the multiplicative factor two would be optimal by Corollary 1.

References

[1] O. Amini, F. Huc, and S. Pérennes. On the pathwidth of planar graphs. Research report, INRIA Research Report HAL-00082035, July 2006.

[2] H. L. Bodlaender. A partial *k*-arboretum of graphs with bounded treewidth. *Theor. Comput. Sci.*, 209(1-2):1–45, 1998.

[3] H. L. Bodlaender and F. V. Fomin. Approximation of pathwidth of outerplanar graphs. *J. Algorithms*, 43(2):190–200, 2002.

[4] H. L. Bodlaender and T. Kloks. Efficient and constructive algorithms for the pathwidth and treewidth of graphs. *J. Algorithms*, 21(2):358–402, 1996.

[5] V. Bouchitté, F. Mazoit, and I. Todinca. Chordal embeddings of planar graphs. *Discrete Math.*, 273(1-3):85–102, 2003. EuroComb'01 (Barcelona).

[6] J. Díaz, J. Petit, and M. Serna. A survey of graph layout problems. *ACM Computing Surveys*, 34(3):313–356, 2002.

[7] J. Ellis and M. Markov. Computing the vertex separation of unicyclic graphs. *Inform. and Comput.*, 192(2):123–161, 2004.

[8] J. A. Ellis, I. H. Sudborough, and J. S. Turner. The vertex separation and search number of a graph. *Inform. and Comput.*, 113(1):50–79, 1994.

[9] F. Fomin and D. M. Thilikos. On self duality of pathwidth in polyhedral graph embeddings. Report in Informatics 316, University of Bergen, March 2006.

[10] F. V. Fomin. Pathwidth of planar and line graphs. *Graphs and Combinatorics*, 19(1):91–99, 2003.

[11] R. Govindan, M. A. Langston, and X. Yan. Approximating the pathwidth of outerplanar graphs. *Inform. Process. Lett.*, 68(1):17–23, 1998.

[12] N. G. Kinnersley. The vertex separation number of a graph equals its pathwidth. *Inform. Process. Lett.*, 42(6):345–350, 1992.

[13] D. Lapoire. *Structuration des graphes planaires*. PhD thesis, Université de Bordeaux, France, 1999.

[14] N. Megiddo, S. L. Hakimi, M. R. Garey, D. S. Johnson, and C. H. Papadimitriou. The complexity of searching a graph. *J. Assoc. Comput. Mach.*, 35(1):18–44, 1988.

[15] S. L. Mitchell. Linear algorithms to recognize outerplanar and maximal outerplanar graphs. *Inform. Process. Lett.*, 9(5):229–232, 1979.

[16] B. Reed. Treewidth and tangles: an new connectivity measure and some applications. In R. A. Bayley, editor, *Surveys in Combinatorics*, pages 87–162. Cambridge University Press, 1997.

[17] N. Robertson and P. D. Seymour. Graph minors. I. Excluding a forest. *J. Combin. Theory Ser. B*, 35(1):39–61, 1983.

[18] N. Robertson and P. D. Seymour. Graph minors. III. Planar tree-width. *J. Combin. Theory Ser. B*, 36(1):49–64, 1984.

[19] P. Scheffler. A linear algorithm for the pathwidth of trees. In R. Henn R. Bodendiek, editor, *Topics in Combinatorics and Graph Theory*, pages 613–620. Physica-Verlag Heidelberg, 1990.

[20] K. Skodinis. Construction of linear tree-layouts which are optimal with respect to vertex separation in linear time. *J. Algorithms*, 47(1):40–59, 2003.

[21] M. M. Sysło. Characterisations of outerplanar graphs. *Discrete Math.*, 26(1):47–53, 1979.

[22] M. Yannakakis. A polynomial algorithm for the min-cut linear arrangement of trees. *J. Assoc. Comput. Mach.*, 32(4):950–988, 1985.

On the Path-width of Planar Graphs *

Omid Amini[1,2] Florian Huc [1] Stéphane Pérennes [1]

FirstName.LastName@sophia.inria.fr

Abstract

In this paper, we present a result concerning the relation between the path-with of a planar graph and the path-width of its dual. More precisely, we prove that for a 3-connected planar graph G, $pw(G) \leq 3pw(G^*) + 2$. For 4-connected planar graphs, and more generally for Hamiltonian planar graphs, we prove a stronger bound $\text{pw}(G^*) \leq 2\ \text{pw}(G) + c$. The best previously known bound was obtained by Fomin and Thilikos who proved that $\text{pw}(G^*) \leq 6\ \text{pw}(G) + cte$. The proof is based on an algorithm which, given a fixed spanning tree of G, transforms any given decomposition of G into one of G^*. The ratio of the corresponding parameters is bounded by the maximum degree of the spanning tree.

1 Introduction

A *planar graph* is a graph that can be embedded in the plane without crossing edges. It is said to be *outerplanar* if it can be embedded in the plane without crossing edges and such that all its vertices are incident to the unbounded face. For any graph G, we denote by $V(G)$ its vertex set and by $E(G)$ its edge set. The *dual* of a planar graph G, denoted by G^*, is the graph with one vertex for each face, and joining two vertices by one edge in G^* for each edge that the corresponding faces in G share. The weak dual $\mathcal{T}_G$ is the induced subgraph of G^* obtained by removing the vertex corresponding to the unbounded face. Note that the dual of a planar graph can be computed in linear time.

The notion of path-width was introduced by Robertson and Seymour [11]. A *path decomposition* of a graph G is a set system $(X_1, \ldots, X_r)$ of $V(G)$ (X_is are called *bags*) such that

1. $\bigcup_{i=1}^{r} X_i = V(G)$;
2. $\forall\ xy \in E, \exists\ i \in \{1, \ldots, r\} :\ \ \{x, y\} \subseteq X_i$;
3. for all $1 \leq i_0 < i_1 < i_2 \leq r$, $X_{i_0} \cap X_{i_2} \subseteq X_{i_1}$.

The *width* of the path-decomposition $(X_1, \ldots, X_r)$ is $\max_{1 \leq i \leq r} |X_i| - 1$. The *path-width* of G, denoted by $\text{pw}(G)$, is the minimum width over its path decompositions. For the definition of other width-parameters, *branch-width* and *tree-width*, we refer to the survey of Bodlaender [3] and Reed [10]. We denote the tree-width and branch-width of G by $tw(G)$ and $bw(G)$, respectively.

Comparing the width-parameters of G and G^* seems to be a very natural question. Indeed, a more interesting (algorithmic) problem should ask for a natural way of transforming a given decomposition of G to a decomposition of G^* without changing "too much" the width of the corresponding decompositions.

It is a consequence of Seymour and Thomas work [13] that such a comparison exists for branch-width:

*This work has been partially funded by the European Community project IST FET AEOLUS. The second author was funded by CNRS and Provence Alpes Côte d'Azur.

[1]Mascotte, join project- INRIA/CNRS-I3S/UNSA- 2004, route des Lucioles - Sophia-Antipolis, FRANCE.

[2]École Polytechnique, Palaiseau, FRANCE.

Theorem 1 (Seymour and Thomas [13]) *For every bridgeless planar graph G, $bw(G) = bw(G^*)$.*

Calculating branch-width can be also done in polynomial time for planar graphs.
For tree-width, Lapoire [9] proved the following theorem using algebraic methods:

Theorem 2 (Lapoire [9]) *For every planar graph G, $\mathrm{tw}(G) \leq \mathrm{tw}(G^*) + 1$.*

This was a conjecture of Robertson and Seymour [12] and a combinatorial shorter proof of this theorem can be found in Bouchitté et al [4]. Remark that it is an open question to see if tree-width can be calculated in polynomial time in planar graphs.

But how about the path-width? Is there any relation? Note that computing the path-width of graphs is an NP-complete problem even for planar graphs with maximum degree 3. For biconnected outerplanar graphs, Bodlaender and Fomin [2] provided a linear time algorithm which approximates the path-width of biconnected outerplanar graphs with a multiplicative factor of 2. To do so, they exhibit a relationship between the path-width of an outerplanar graph and the path-width of its dual. More precisely they prove that for any biconnected outerplanar graph G without loops and multiple edges, $\mathrm{pw}(G^*) \leq \mathrm{pw}(G) \leq 2\ \mathrm{pw}(G^*) + 2$.

By the results of Coudert, Huc and Sereni [5] it is impossible to have $pw(G) = pw(G^*)$ (Fomin and Thilikos provided similar constructions in [8]): they constructed an infinite family of outerplanar graphs such that each one has path-width twice the path-width of its dual. Indeed they proved the following theorem:

Theorem 3 (Coudert et al. [5]) *For every biconnected outerplanar graph G, $pw(G^*) \leq pw(G) \leq 2\ pw(G^*) - 1$. Furthermore, for every integer $p \geq 1$ and every integer $k \in \{1, 2, \ldots, p+1\}$, there exists a biconnected outerplanar graph of path-width $p+k$ whose dual has path-width $p+1$.*

Fomin and Thilikos showed in [8] a linear inequality between the two parameters:

Theorem 4 (Fomin and Thilikos [8]) *There is a constant c such that for every 3-connected planar graph G we have $pw(G^*) \leq 6\ pw(G) + c$.*

In this paper, we propose an algorithm which given a spanning tree of G, transforms a given decomposition of G into one of G^*. The ratio of the corresponding parameters is bounded by the maximum degree of the spanning tree. Our transformation then reduces the question of comparing the different width-parameters of G and G^* to the problem of finding spanning trees of low maximum degree in a given planar graph.
Theorems 5 and 6 are the main theorems of this paper.

Theorem 5 *For every 3-connected planar graph G, we have $pw(G^*) \leq 3\ pw(G) + 2$.*

Remark that Theorem 5 improves Theorem 4.

Theorem 6 *If G is a planar graph with a Hamiltonian path, then $pw(G^*) \leq 2\ pw(G) + 1$.*

Theorem 6 in particular proves that for a 4-connected planar graph G we always have $\mathrm{pw}(G^*) \leq 2\ \mathrm{pw}(G) + 1$. Indeed, by a theorem of Tutte [14], every such graph has a Hamiltonian cycle.

2 Main Theorem

In this section we present the proofs of Theorems 5 and 6. We will use the following notations:

Given a planar graph G on vertex set $V(G)$ and edge set $E(G)$ of maximum degree $\Delta(G)$, by $F(G)$ we mean the set of faces of G, which is also the vertex set of its dual. The number of faces, edges and vertices of G are respectively denoted by f_G, e_G and n_G. Given a face $F \in F(G)$, we denote the set of vertices belonging to this face by $V(F)$. $E(F)$ is the set of edges appearing on the boundary of F. Given a set A, by $\mathcal{P}(A)$ we denote the family of all subsets of A.

Definition 1 Let G and H be two graphs and σ a map from $V(G)$ to $\mathcal{P}(V(H))$. We say that σ is a *connected map from G to H* if it satisfies the following two properties:

1. for every $v \in V(G)$, the subgraph of H induced by $\sigma(v)$ is connected.
2. for every edge $vw \in E(G)$, the subgraph of H induced by $\sigma(v) \cup \sigma(w)$ is also connected.

For every vertex $w \in V(H)$, we define $\sigma^{-1}(w) := \{v \in V(G) |\ w \in \sigma(v)\}$. The *degree* of σ is the integer $k = \max |\sigma^{-1}(w)|$.

Lemma 1 *Let G and H be two graphs. If there exists a connected map σ of degree at most k from G to H, we have:*

$$pw(G) \leq k \cdot pw(H) + k - 1$$

Proof

Let $(X_1, \ldots, X_r)$ be a path-decomposition of H. We first show that the sequence $(\sigma^{-1}(X_1), \ldots, \sigma^{-1}(X_r))$ provides a path-decomposition of G. For this, we should prove the three properties of a path-decomposition:

- Every vertex v of G appears in one $\sigma^{-1}(X_i)$. To show this, let $u \in \sigma(v)$. As $(X_1, \ldots, X_r)$ forms a path-decomposition of H, there exists an i such that $u \in X_i$. It is clear that for this bag, v appears in $\sigma^{-1}(X_i)$.
- For every edge $xy \in E(G)$, there is one $\sigma^{-1}(X_i)$ which contains both x and y. To prove this, let $A = \sigma(x)$ and $B = \sigma(y)$. The graph induced by H on $A \cup B$ is connected, so at least one of the two following two cases appears:
  - $A \cap B \neq \emptyset$: let $u \in A \cap B$ and X_i be the bag which contains u. Then $\sigma^{-1}(X_i)$ contains both x and y.
  - There exist $a \in A$ and $b \in B$ such that $ab \in E(H)$. Let X_i be the bag which contains both a and b. It is clear that $\sigma^{-1}(X_i)$ contains both x and y.
- For all $1 \leq i_0 < i_1 < i_2 \leq r$, we should prove $\sigma^{-1}(X_{i_0}) \cap \sigma^{-1}(X_{i_2}) \subseteq \sigma^{-1}(X_{i_1})$. Let $v \in \sigma^{-1}(X_{i_0}) \cap \sigma^{-1}(X_{i_2})$. The graph induced by $\sigma(v)$ in H, i.e. $H[\sigma(v)]$, is connected and intersects both X_{i_0} and X_{i_2}. The graph $H[\sigma(v)] \setminus X_{i_1}$ is not connected. We infer that $\sigma(v) \cap X_{i_1} \neq \emptyset$, which implies $v \in \sigma^{-1}(X_{i_1})$.

As the degree of σ is at most k and $|X_i| \leq pw(H) + 1$, we have $|\sigma^{-1}(X_i)| \leq k(pw(H) + 1)$, which proves that the width of the path-decomposition $(\sigma^{-1}(X_1), \ldots, \sigma^{-1}(X_r))$ is at most $k.pw(H) + k - 1$. This finishes the proof of the lemma. □

Remark that the same proof applies for other types of decompositions.

From now on, our aim will be to find a way to produce low degree connected maps from G^* to G. The key role will be played by spanning trees of G: every spanning tree of maximum degree k produces a connected map from G^* to G of degree at most k. Before we proceed, we need some new definitions:

An *edge-assignment* to faces of G is a one-to-one map from the faces of G to the edges of G which to each face F of G, associates one edge of $E(F)$. More formally:

Definition 2 An edge-assignment is a function: $\tau : F(G) \to E(G)$ such that

1. for every face $F \in F(G)$, $\tau(F)$ is an edge on the boundary of F, and
2. $\tau(F) \neq \tau(F')$ for all distinct faces $F, F' \in F(G)$.

Given an edge-assignment, we define the map $\sigma_\tau : F(G) \to \mathcal{P}(V(G))$ as follows: σ_τ associates to every face F in $F(G)$ the subset $V(F) \setminus V(\tau(F))$.

Proposition 1 *For every edge-assignment τ, the map σ_τ is connected.*

Proof It is clear that the graph induced by $\sigma_\tau(F)$ is connected, since it forms a path. Two faces F_1, F_2 sharing an edge e can not be both associated to e (since they are associated to different edges). Consequently $\sigma_\tau(F_1) \cup \sigma_\tau(F_2)$ induces also a connected subgraph of G. This proves that σ_τ is connected. □

Given an edge-assignment τ, let H be the subgraph of G consisting of non selected edges; i.e. $H = G \setminus \{\tau(F) | F \in V(G^*)\}$. Using Euler's Formula ($f_G + n_G = e_G + 2$), we infer that H contains exactly $n_G - 2$ edges. We have

Proposition 2 *For all $v \in V(G)$, $|\sigma_\tau^{-1}(v)| = deg_H(v)$.*

Proof A selected edge (an edge of $G \setminus H$) is associated to one of the two faces containing it. Given a vertex v of G of degree d, it appears exactly in d faces. Suppose r edges incident to v are selected; they are associated to exactly r faces incident with v. The image of this faces by σ_τ does not contain v, and v appears in $\sigma_\tau(F)$ for all other faces F incident to v. So $|\sigma_\tau^{-1}(v)| = d - r = deg_H(v)$. □

Corollary 1 *σ_τ is of degree $\Delta(H)$.*

Remark that the average degree in H is always < 2.

Definition 3 A subgraph $H \subseteq G$ is *nice* if it has $n_G - 2$ edges and if there exists an edge-assignment τ with $\tau(F) \in E(G) \setminus E(H)$.

Hence to prove Theorem 5, by using Lemma 1, we need to find a nice subgraph with maximum degree at most 3. To proceed we need the following lemma:

Lemma 2 *Let G be a planar graph and T a spanning tree of maximun degree k in G. Let e be an edge of T. The subgraph $H = T \setminus e$ is a nice subgraph of maximum degree at most k.*

Proof To prove that H is nice we will apply Hall's matching theorem to the adjacency graph A between faces of G and edges of $G \setminus H$. More precisely, A is the bipartite graph on vertex set $F(G) \sqcup (E(G) \setminus E(H))$. An edge of A connects $F \in F(G)$ (i.e. F a face of G) to $e \in E(G) \setminus E(H)$, if e belongs to $E(F)$. We want to prove that there exists a matching in A covering all the vertices of $F(G)$. Given a set of faces $\{F_1, \ldots, F_i\}$, we need to prove that the corresponding set has at least i neighbours in A. Let us consider the planar graph S obtained by taking the union of F_i's, we have:

- $f_S \geq i + 1$ (because G^* is connected), and
- $f_S + n_S = e_S + 2$ (Euler's formula).

We conclude that $e_S - (n_S - 1) \geq i$. As H is a forest, the number of edges of H incident to vertices of S is at most $n_S - 1$. So the hypothesis of Hall's theorem are satisfied. This proves that H is a nice subgraph. □

Barnette proved in [1] the following theorem:

Theorem 7 (Barnette [1]) *Every 3-connected planar graph has a spanning tree of maximum degree 3.*

Later, Czumaj and Strothmann [6] proved that such a spanning tree in G can be found in linear time. We can now present the proof of Theorems 5 and 6:

Proof of Theorem 5 Barnette's theorem insures the existence of a spanning tree of maximun degree three, which by Lemma 2 provides a nice subgraph of maximun degree 3. The associated edge-assignment gives a connected map of degree three (Proposition 1 and Corollary 1), hence Theorem 5 follows from Lemma 1. The algorithm of Czumaj and Strothmann [6] combined with the algorithm of finding a maximum matching in A (see the proof of Lemma 2) results in a polynomial time algorithm to find the corresponding decompositions. □

Proof of Theorem 6 Using Lemma 2, deleting an edge of the Hamiltonian path gives a nice subgraph of G of maximum degree 2. □

3 Discussion

A 3-connected planar graph does not contain in general a nice subgraph of maximum degree two. Indeed, it can happen that the graph G does not contain a subgraph of maximum degree 2 with $n-2$ edges. Even more, for every constant k, there exist planar graphs such that for every subgraph H, containing $n-2$ edges and covering all vertices, we have $\sum_{deg(v)\geq 2} deg(v) \geq k$. Examples of such graphs are planar graphs which can not be covered by less than k disjoint paths. In [7], an inductive construction of an infinite family of non Hamiltonian Delaunay triangulations is presented. Two examples of graphs of this family are given in Figure 3. By looking at the central separating triangle and using induction, one can prove that for any k, a large graph of this family does not contain k disjoint paths covering all its vertices. This shows that we can not expect to improve Theorem 5 via nice subgraphs.

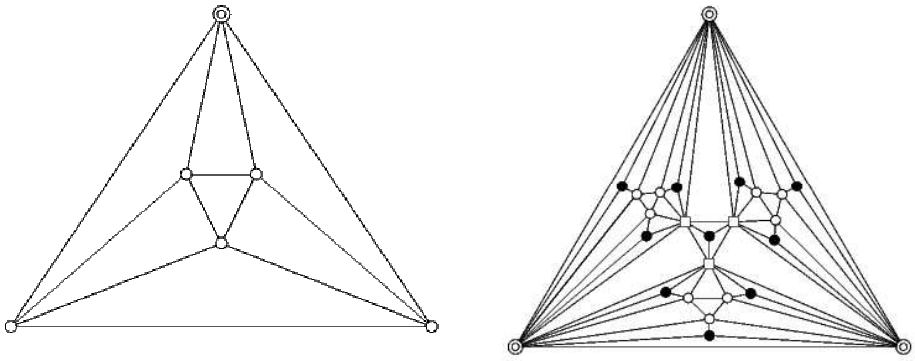


Figure 1: Two graphs of the family constructed in [7]

Acknowledgements

The authors would like to thank David Coudert and Jean-Sébastien Sereni for interesting discussions. Special thanks to Frédéric Havet and Stéphan Thomassé for their comments on the first drafts of this paper. We are also indebted to anonymous referees for valuable comments.

References

[1] D. Barnette. Trees in Polyhedral Graphs. *Canadian Journal of Mathematics*, (18):731–736, 1966.

[2] H. L. Bodlaender and F. V. Fomin. Approximation of Path-width of Outerplanar Graphs. *Journal of Algorithms*, 43(2):190–200, 2002.

[3] H.L. Bodlaender. Tree-width: Algorithmic Techniques and Results. *Proceedings of 22nd International Symposium MFCS, LNCS*, 1295, 1997.

[4] V. Bouchitté, F. Mazoit, and I. Todinca. Chordal Embeddings of Planar Graphs. *Discrete Mathematics*, 273(1-3):85–102, 2003.

[5] D. Coudert, F. Huc, and J. S. Sereni. Path-width of Outerplanar Graphs. *Journal of Graph Theory*, 55(1):27–41, 2007.

[6] A. Czumaj and W.B. Strothmann. Bounded Degree Spanning Trees. *Algorithms, ESA 97 (Graz), Springer Lecture Notes in Computer Science 1284*, pages 104–117, 1997.

[7] M. B. Dillencourt. Finding Hamiltonian Cycles in Delaunay Triangulations Is NP-Complete. *Discrete Applied Mathematics*, 64(3):207–217, 1996.

[8] F. V. Fomin and D. M. Thilikos. On Self-Duality for Path-width in Polyhedral Graph Embeddings. *Journal of Graph Theory*, 55(1):42–45, 2007.

[9] D. Lapoire. *Structuration des graphes planaires*. PhD thesis, Université de Bordeaux, France, 1999.

[10] B. Reed. Tree-width and tangles: an new connectivity measure and some applications. In R. A. Bayley, editor, *Surveys in Combinatorics*, pages 87–162. Cambridge University Press, 1997.

[11] N. Robertson and P. D. Seymour. Graph minors. I. Excluding a Forest. *Journal of Combinatorial Theory Series B*, 35(1):39–61, 1983.

[12] N. Robertson and P.D. Seymour. Graph minors. III. Planar tree-width. *Journal of Combinatorial Theory Series B*, 36(1):49–64, 1984.

[13] P. Seymour and R. Thomas. Call routing and the ratcatcher. *Combinatorica*, 14:217–241, 1994.

[14] T. Tutte. A Theorem on Planar Graphs. *Transactions of American Mathematical Society*, 82:99–116, 1956.

Annexe F

Mixing Digraphs

In this appendix I introduce a new parameter for digraphs called mixing. A digraph D is said f-mixing if for every non necessarily disjoint pairs of subsets $A, B \subset V$ with $e(A,B) \geq f$, we have $e(B,A) > e(A,B)$. I then study the conditions on this parameter which force a digraph D to contain a subdigraph H. In the case $H = \overrightarrow{C_4}$, I fully answer this question : there is some $C_1 < C_2$, such that every $C_1(e^2/n^2)$-mixing digraph D contains a directed 4-cycle and such that, when $e \geq 100n^{3/2}$, there is $C_2(e^2/n^2)$-mixing digraph D without directed 4-cycles. I also answer this question when H is non-bipartite and when D is dense. I finish this appendix on the question of how mixing a digraph can be.

This appendix is based on a joint work with O. Amini and S. Griffiths. Part of these results have been presented at the conference LAGOS 07, which took place from the 24th to the 28th of November 2007 in Chile.

Computing the best constants for the theorems we propose was not our objective, hence the constants given in the proofs are not the best possible.

Graphs and forbidden subgraphs. The idea underlying mixing digraphs is to find a sufficient condition to force an oriented graph to have some specific patterns. Conversely we can see the problem as asking for the strongest conditions we can force while avoiding those patterns. In our case, the pattern is given by a small digraph H, and given a digraph D, we say that either D has H as a subgraph or that D is H-free. More formally we can define subgraphs and H-free digraphs as follow :

Definition F.1 Given a digraph H on vertex set $V(H) = \{u_1, \dots, u_h\}$ and arc set $E(H)$, we say that a digraph D contains H as a (non necessarily induced) subgraph iff there is a set of h vertices of D, $\{v_1, \dots, v_h\}$, such that for every arc u_iu_j of H, v_iv_j is an arc of D.

We say a digraph D is H-free iff it does not contain H as a subgraph. Given a family $\mathcal{H}$, we say D is $\mathcal{H}$-free if it is H-free for any $H \in \mathcal{H}$.

The analog problem has been studied for undirected graphs. Turan's theorem answers the question of the maximum number of edges a K_k-free graph can have, for all k. It says that it is $(1-1/(k-1))n^2/2, \forall k \in \mathbb{N}^*$. Given $k \in \mathbb{N}$, the extremal example is the Turan graph T_{k-1} which has $(1-1/(k-1))n^2/2$ edges and does not contain any graph of chromatic number k or more (and in particular they are K_k-free). A generalization of this theorem due to P. Erdős and M. Simonovits in [ES66] states that these graphs are asymptotically extremal for every family $\mathcal{H}$, each graph of which has chromatic number at least k and one of them has chromatic number exactly k. Denoting $ex(n,\mathcal{H})$ the maximum number of edges a $\mathcal{H}$-free graph on n vertices can have, one can state this result as follows :

Theorem F.2 ([ES66]) *Let $\mathcal{H}$ be a family, in which each graph has chromatic number at least k and one of them has chromatic number exactly k, we have $ex(n,\mathcal{H}) = (1-1/(k-1)+o(1))n^2/2$.*

This theorem gives an asymptotic answer when $k > 2$, however, when $k = 2$, we only get $ex(n,\mathcal{H}) = o(n^2)$. Still, more precise results are known, as for example when $\mathcal{H} = \{C_4\}$, the 4-cycles : it was shown by P. Erdős, A. Rényi and V.T. Sós [ERS64] that $ex(n,C_4) = (\frac{1}{2}+o(1))n^{3/2}$. In fact, for any n, they present a C_4-free graph with edges which turns out to be extremal [Fur83]. Their example (which is produced using a projective plane) is discussed further in Section F.1.2.

In 1973, in [BES73], W.G. Brown, P. Erdős and M. Simonovits studied the question in the directed case. The digraphs they considered are without loops and multiple arcs, but they allow 2-cycles. Exactly, they considered the question of the maximum number of arcs a digraph on n vertices can have if it does not contain any digraph of a family $\mathcal{H} = \{H_1, \ldots, H_q\}$ as subgraph. They proved the existence of a sequence of asymptotically extremal graphs. Later, in [BES85], they proved it in an algorithmic way.

Theorem F.3 ([BES85]) *There exist a finite algorithm which determines sequences of digraphs which are asymptotically extremal for any given finite family $\mathcal{H}$.*

However, there is no simple characterization, similar to the minimal chromatic number in simple graphs, of digraphs which arise in asymptotically extremal sequences.

When 2-cycles are forbidden. When $\mathcal{H}$ contains the directed 2-cycle $\overrightarrow{C_2}$, and digraphs which are not acyclic, the transitive tournament D on vertex set $V = \{1,...,n\}$, and arc set $E = \{ij : i < j\}$ is an extremal graph for $\mathcal{H}$. Indeed D has $\binom{n}{2}$ arcs (the most possible while avoiding 2-cycles) but contains no elements of $\mathcal{H}$ since it is acyclic. So in this situation, no condition on the number of arcs guarantees the presence of H as a subgraph. However, in this example D has an extreme bias in its orientation : we can find subsets $A, B \subset V$ with $e(A,B) = n^2/4$ but $e(B,A) = 0$. To obtain more information about

the $\mathcal{H}$-free digraphs when $\mathcal{H}$ contains the directed 2-cycle and non-acyclic digraphs, we propose to investigate if only "strongly biased digraphs" can avoid containing elements of $\mathcal{H}$ as subgraph.

Remark F.4 Given a set of forbidden digraphs $\mathcal{H}$, if it contains the directed 2-cycle, investigating if only "strongly biased *digraphs*" can avoid containing elements of $\mathcal{H}$ as subgraph is equivalent to the same question over *oriented graphs*. Hence, from now on, instead of considering $\mathcal{H}$ with the directed 2-cycle, we will investigate the question only over oriented graphs.

When $\mathcal{H}$ contains a directed 4-cycle, we indeed prove that only "strongly biased oriented graphs" can avoid containing elements of $\mathcal{H}$. More precisely, we show that, if for all subsets $A,B \subset V$ with $e(A,B) \geq e^2/32n^2$ we have $e(B,A) > e(A,B)/2$, then D must contain a 4-cycle. This is Theorem F.10.

In Section F.2, we extend this result to dense oriented graphs when $\mathcal{H}$ is any set of digraphs, one of which is bipartite. However, when no element of $\mathcal{H}$ is bipartite, we prove that there are dense oriented graphs whose orientations are not biased and which are $\mathcal{H}$-free.

Mixing digraphs. To describe more precisely what we mean by "strongly biased oriented graphs", we introduce a new notion : the notion of mixing digraphs.

Definition F.5 Given a digraph D, $f \in \mathbb{R}$ and $\gamma \in]0,1[$ we say D is (f,γ)-*mixing* if for every non necessarily disjoint pairs of subsets $A,B \subset V$ with $e(A,B) \geq f$, we have $e(B,A) > \gamma e(A,B)$. When $\gamma = \frac{1}{2}$, we simply say f-*mixing*.

In other words, a digraph D is f-mixing if when we have a lot of arcs from a set A of vertices to a set B (i.e. at least f arcs) then we have more than half as many back. More generally for $\gamma \in]0,1[$ we say that D is (f,γ)-mixing if for every pair $A,B \subset V$ with more than f arcs from A to B, we have more than $\gamma e(A,B)$ arcs from B to A.

This definition gives us a large range of mixing conditions getting stronger as f decreases. In particular $e+1$-mixing is a really weak condition as all digraphs are $e+1$-mixing. Hence the weakest mixing condition we will ask for will be when $f = O(e)$. Being 1-mixing means that all arcs are in a 2-cycle. In the following, the conditions we will use are mainly (in strengthening order when e is big enough) e-, $(e^k/n^2)^{1/(k-1)}$-, e^2/n^2- and n-mixing.

Given this definition, one can ask for the weakest mixing condition which forces an oriented graph to have a given digraph H as subgraph ? If we write $mex(n,e,H) = min_f\{\forall D,\ f-mixing,\ with\ e\ edges\ and\ n\ vertices : H \subset D\}$ the question is to determine $mex(n,e,H)$. More generally we can ask, given a set $\mathcal{H}$ of forbidden subgraphs, for $mex(n,e,\mathcal{H}) = min_f\{\forall D\ f-mixing,\ with\ e\ edges\ and\ n\ vertices : \exists H \in \mathcal{H} : H \subset D\}$

We fully answer this question when $\mathcal{H}$ is one of the orientation of the 4-cycle.

Definition F.6 We write $\overrightarrow{C_4}$ for the directed 4-cycle, $a\overrightarrow{C_4}$ for the almost directed 4-cycle, i.e., the oriented 4-cycle with a single reversed arc, $p\overrightarrow{C_4}$ for the 4-cycle made of

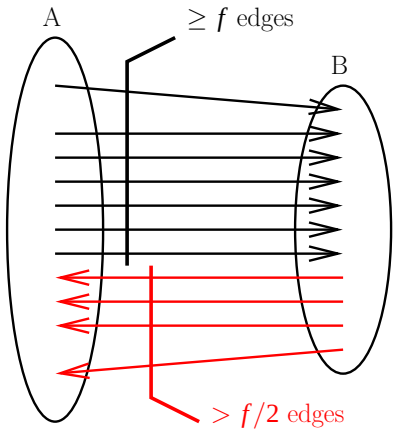


FIG. F.1 – Illustration of the mixing condition

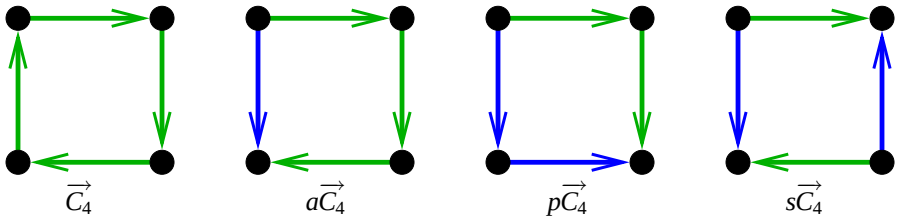


FIG. F.2 – The four different orientations of C_4

two parallel paths of length two and $s\overrightarrow{C_4}$ for an antidirected 4-cycle, i.e., an oriented 4-cycle made of two 2-stars. This is illustrated by Figure F.2.

The main results of this section are Theorem F.10 and Corollary F.23 which give the value of $mex(n,e,\overrightarrow{C_4})$ up to a multiplicative constant. There are some constants $C_1 < C_2$ such that :

$$C_1\frac{e^2}{n^2} \leq mex(n,e,\overrightarrow{C_4}) \leq C_2 max(\frac{e^2}{n^2},n).$$

Before going any further, one has to insure that an f-mixing digraph indeed exists, at least for some values of f. Thankfully this question is answered positively by Lemma F.7. In fact we prove much more : if G is a simple graph and D is obtained from G by orienting the arcs of G at random then D is Ln-mixing with positive and high probability (where L is a sufficiently large constant).

Lemma F.7 *There exists L such that for any graph G on n vertices, the oriented graph D obtained by orienting the edges of G at random, is Ln-mixing with positive probability (always) and with high probability (as n tends to infinity). In particular, $\forall \varepsilon > 0$, if e is much larger than $n^{3/2}$ then, for n large enough, Ln is less than $\varepsilon e^2/n^2$ and so D is*

$(\varepsilon e^2/n^2)$-mixing with positive probability and with high probability. More generally for any $\gamma \in]0,1[$ there exists L_γ such that D is $(L_\gamma n, \gamma)$-mixing with positive probability and with high probability.

Proof.

We prove the second, more general statement. For a pair $(A,B) \subset V^2$ we write $e(A,B)$ for the number of arcs between A and B in the graph G. Consider a pair $(A,B) \subset V^2$ with $e(A,B) \geq Ln$, L a constant to be defined later. From Chernoff's inequality we have,

$$\mathbb{P}(e(B,A) \leq \gamma e(A,B)) \leq \exp(-c(\gamma)e(A,B)) \leq \exp(-c(\gamma)Ln)$$

with $c(\gamma)$ a positive constant dependent on γ. Set $L = 2/c(\gamma)$. Let $W = \{(A,B) : (A,B) \subset V^2 \text{ and } e(A,B) \geq Ln\}$. Noting that there are only 4^n ways to pick a pair of subsets $(A,B) \subset V^2$ we have $|W| \leq 4^n$, and so

$$\mathbb{P}\left(e(B,A) \leq \gamma e(A,B) \text{ for some pair } (A,B) \subset W\right) \leq 4^n \exp(-c(\gamma)Ln) \leq 4^n \exp(-2n).$$

This quantity goes to 0 as n tends to infinity. If an oriented graph G is not (Ln,γ)-mixing then there exist subsets $(A,B) \subset V^2$ with $e(A,B) \geq Ln$ (implying $(A,B) \subset W^2$) and $e(B,A) < \gamma e(A,B)$. So that with high probability D is (Ln,γ)-mixing.

□

Now that we know mixing digraphs exist, let us give some general results on mixing digraphs.

Given a digraph D and a set of vertices A, we note by $d_A^+(u)$ the number of arcs from A to u and $d_A^-(u)$ the number of arcs from u to A.

Lemma F.8 *Let D be an f-mixing digraph and A a set of vertices. Most of the arcs are incidents to vertices u satisfying $1/2d_A^+(u) \leq d_A^-(u) \leq 2d_A^+(u)$, more precisely, at most $3f$ arcs do not have this property.*

Proof. Set $B = \{u, d_A^+(u) > 2d_A^-(u)\}$. We have $e(B,A) = \sum_B d_A^+(u) \geq 2\sum_B d_A^-(u) = e(A,B)$, hence (A,B) does not satisfy the mixing condition and we have $\sum_B d_A^+(u) < f$ and $\sum_B d_A^-(u) < f/2$, hence at most $3/2f$ arcs are incident to vertices with $d_A^+(u) > 2d_A^-(u)$. We proceed similarly for the other inequality. □

Lemma F.9 *Given an f-mixing digraph D and a set of vertices $A \subset V$, the number of paths of length two from A to A is at least $\frac{(e(A,V)-3f)^2}{2n}$.*

Proof. The number of paths of length two from A to A is precisely $\sum_{u \in A} d_A^+(u)d_A^-(u)$.

Set $B = \Gamma^+(A)$. Set $W = \{u \in B : 1/2d_A^+(u) \leq d_A^-(u) \leq 2d_A^+(u)\}$. From Lemma F.8, we have $e(A,W) \geq e(A,V) - 3f$. We can lower bound the number of path of length two from A to A by $\sum_B d_A^-(u)d_A^+(u) \geq \sum_W d_A^-(u)d_A^+(u) \geq 1/2\sum_B d_A^+(u)^2 \geq 1/2n(\sum d_A^+(u))^2 \geq 1/2n(e(A,V)-3f)^2$. □

F.1 On the value of $mex(n,e,\overrightarrow{C})$

In this section, we study $mex(n,e,\overrightarrow{C})$ when $\overrightarrow{C}$ is a directed cycle of even length. First we will give a lower bound for $mex(n,e,\overrightarrow{C_4})$, then we will give upper bounds for $mex(n,e,\overrightarrow{C_4})$ and more generally for $mex(n,e,\overrightarrow{C_{2k}})$ (under some assumptions). Finally we will look at some lower bounds for $mex(n,e,\overrightarrow{C_{2k}})$.

F.1.1 A lower bound for $mex(n,e,\overrightarrow{C_4})$

We now wish to show that if D obeys a certain mixing condition then it contains a directed 4-cycle. (We do not require any separate condition on e, but we keep in mind the case with e much larger than $n^{3/2}$, a particular example being $e = n^{1+\alpha}$ for some $\alpha \in]0.5,1]$.) The main result of this section is, if C_1 is small enough, every C_1e^2/n^2-mixing oriented graph contains a directed 4-cycle. (For example when $e = n^{1+\alpha}$ the condition C_1e^2/n^2-mixing, means $C_1n^{2\alpha}$-mixing. For $\alpha \in]0.5,1[$ this is a strictly stronger condition than C_1e-mixing and strictly weaker than C_1n-mixing.)

Theorem F.10 *For $C_1 = 1/32$, every (C_1e^2/n^2)-mixing oriented graph D contains a directed 4-cycle.*

Remark F.11
i) In Section F.1.2, we give examples of oriented graph which are (C_2e^2/n^2)-mixing (where C_2 is a large constant) and do not contain directed 4-cycles. This means that the mixing condition cannot be weakened, so Theorem F.10 is best possible.
ii) In the case where e is of order n^2, it is not too difficult to deduce Theorem F.10 using Szemerédi's regularity Lemma (c.f. [KSSS02] for a survey and Section F.2).

In our proof we apply the mixing condition repeatedly, receiving on each application a new piece of information. We begin with Corollary F.12 that shows that there must be many paths of length 2. In fact this lemma requires only that D is $e/6$-mixing (this is much weaker than our mixing condition because $e/6$ is much larger than $e^2/32n^2$)

For each $v \in V$, we write $e_v^{(2)}$ for the number of paths of length two starting at v, i.e., $e_v^{(2)} = e(\Gamma^+(v),\Gamma^{++}(v))$.

Corollary F.12 *An $e/6$-mixing digraph D contains at least $e^2/8n$ paths of length two, i.e.,*

$$\sum_{v\in V} e_v^{(2)} \geq e^2/8n$$

Proof of Theorem F.10. This a direct consequence of Lemma F.9. □

We now prove Theorem F.10 :

Proof. We focus our attention on the set $W = \{v \in V : e_v^{(2)} \geq e^2/16n^2\}$.

We let $d^{+-}(u,v) = |\Gamma^+(u) \cap \Gamma^-(v)|$, and similarly $d^{++}(u,v) = |\Gamma^+(u) \cap \Gamma^+(v)|$. These *joint degrees* are useful for expressing quantities such as $e(U,\Gamma^+(v))$ (for $v \in V$ and $U \subset$

V), which can be expressed as $e(U,\Gamma^+(v)) = \sum_{u\in U} d^{++}(u,v)$. In particular we shall use that, $e(\Gamma^{++}(v),\Gamma^+(v)) = \sum_{u\in\Gamma^{++}(v)} d^{++}(u,v)$ and $e(\Gamma^{--}(u),\Gamma^+(u)) = \sum_{v\in\Gamma^{--}(u)} d^{++}(u,v)$.

The idea is to note that for vertices $v \in W$, the quantity $e_v^{(2)} = e(\Gamma^+(v),\Gamma^{++}(v))$ is quite large, so, by the mixing condition, $e(\Gamma^{++}(v),\Gamma^+(v))$ is also quite large. We then change perspective and view things from the point of view of vertices $u \in \Gamma^{++}(v)$. By doing so we can prove that, using again the mixing conditions, there are arcs from $\Gamma^+(u)$ to $\Gamma^{--}(u)$.

Notice that even if we restrict ourselves to vertices in W, the number of paths of length two starting at W is still large

$$\sum_{v\in W} e_v^{(2)} \geq \sum_{v\in V} e_v^{(2)} - \sum_{v\in V\setminus W} e_v^{(2)} \geq e^2/8n - ne^2/16n^2 \geq e^2/16n.$$

Now for each $v \in W$ we have $e_v^{(2)} = e(\Gamma^+(v),\Gamma^{++}(v)) \geq e^2/32n^2$. By the $e^2/32n^2$-mixing condition, we have $e(\Gamma^{++}(v),\Gamma^+(v)) \geq e_v^{(2)}/2$. In other words, for all $v \in W$, $\sum_{u\in\Gamma^{++}(v)} d^{++}(v,u) \geq e_v^{(2)}/2$.

Hence,

$$\sum_u \sum_{v\in\Gamma^{--}(u)} d^{++}(v,u) = \sum_v \sum_{u\in\Gamma^{++}(v)} d^{++}(v,u) \geq \frac{1}{2}\sum_{v\in W} e_v^{(2)} \geq \frac{e^2}{32n}.$$

So there exists u with $\sum_{v\in\Gamma^{--}(u)} d^{++}(v,u) \geq e^2/32n^2$, i.e.,

$$e(\Gamma^{--}(u),\Gamma^+(u)) \geq e^2/32n^2.$$

By the mixing condition, we have $e(\Gamma^+(u),\Gamma^{--}(u)) \geq e^2/64n^2$. Since an arc from $\Gamma^+(u)$ to $\Gamma^{--}(u)$ gives an oriented 4-cycle and that $e^2/64n^2 > 0$ (as the empty graph is not $e^2/32n^2$-mixing), we are done. □

Interestingly, a variant of the above method allows us to find a large number of 4-cycles rather than just one. In fact we shall show that, for some C_1', in $(C_1'e^2/n^2)$-mixing oriented graph D, we can find ce^4/n^4 4-cycles (for some constant $c > 0$) which is Theorem F.13. We shall prove this with $c = 1/32768$ and $C_1' = 1/128$ (which gives a stronger mixing condition than the one which was used in the proof of Theorem F.10).

Theorem F.13 *Every* $(e^2/128n^2)$*-mixing oriented graph* D *contains at least* $1/32768e^4/n^4$ *directed* 4*-cycles.*

There are two main weaknesses to the above count. Firstly we say that the number of 4-cycles containing u is at least $e(\Gamma^+(u),\Gamma^{--}(u))$. Whereas in fact each arc wv from $\Gamma^+(u)$ to $\Gamma^{--}(u)$ gives rise to not just one 4-cycle but $d^{+-}(v,u)$ 4-cycles. To take this into account we must restrict our attention to pairs v,u for which $d^{+-}(v,u)$ is not too small. The second main weakness is of course that we have been finding many 4-cycles containing a single vertex. For a better count we must consider a set of vertices $U \subset V$, and find many 4-cycles for each $u \in U$.

Recall that $e_v^{(2)} = e(\Gamma^+(v), \Gamma^{++}(v))$, and recall the set $W = \{v \in V : e_v^{(2)} \geq e^2/16n^2\}$. Observe that the set $\Gamma^{++}(v)$ is exactly $\{u \in V : d^{+-}(v,u) \geq 1\}$. Our proof of Theorem F.10 began by showing that for vertices $v \in W$ there are many arcs from $\Gamma^+(v)$ to $\Gamma^{++}(v)$ and many arcs back. We now wish to restrict our attention to pairs u, v for which $d^{+-}(v,u)$ is not too small. So we define, for each $v \in W$, the set $T_v = \{u \in \Gamma^{++}(v) : d^{+-}(v,u) \geq e_v^{(2)}/2n\}$. We are now ready to prove Theorem F.13.

Proof of Theorem F.13. We have defined for each $v \in W$, a set $T_v \subset \Gamma^{++}(v)$, let us observe that there are many arcs from $\Gamma^+(v)$ to T_v. By definition of T_v, $e(\Gamma^+(v), \Gamma^{++}(v) \setminus T_v) < e_v^{(2)}/2$, so that $e(\Gamma^+(v), T_v) \geq e_v^{(2)}/2 \geq e^2/32n^2$. Now by the mixing condition, $e(T_v, \Gamma^+(v)) \geq e(\Gamma^+(v), T_v)/2 \geq e_v^{(2)}/4$, for all $v \in W$. Using the joint degrees, we have

$$\sum_{u \in T_v} d^{++}(v,u) \geq \frac{e_v^{(2)}}{4} \quad \text{for all } v \in W.$$

It means that there are many arcs from $\Gamma^{++}(v)$ to $\Gamma^+(v)$. We now look from the point of view of vertices u in $\Gamma^{++}(v)$ which are destination of many paths of length two. Set $S_u = \{v \in W : u \in T_v\} = \{v \in W : d^{+-}(v,u) \geq e_v^{(2)}/2n\}$, we have

$$\sum_u \sum_{v \in S_u} d^{++}(v,u) = \sum_{v \in W} \sum_{u \in T_v} d^{++}(v,u) \geq \frac{1}{4} \sum_{v \in W} e_v^{(2)} \geq \frac{e^2}{64n}.$$

We recall that $e(S_u, \Gamma^+(u)) = \sum_{v \in S_u} d^{++}(v,u)$, so that we have

$$\sum_u e(S_u, \Gamma^+(u)) \geq \frac{e^2}{64n}.$$

Now consider the subset $U \subset V$ defined by $U = \{u \in V : e(S_u, \Gamma^+(u)) \geq e^2/128n^2\}$. It represents the vertices which have many out-neighbours which also are their in-in-neighbours, it gives many directed 4-cycles with an arc in the opposite direction. We note that the sum of $e(S_u, \Gamma^+(u))$ over vertices $u \in V \setminus U$ is at most $e^2/128n$, and so,

$$\sum_{u \in U} e(S_u, \Gamma^+(u)) \geq \frac{e^2}{64n} - \frac{e^2}{128n} = \frac{e^2}{128n}.$$

As each vertex $u \in U$ has $e(S_u, \Gamma^+(u)) \geq \varepsilon e^2/n^2$ we obtain from the mixing condition that $e(\Gamma^+(u), S_u) \geq e(S_u, \Gamma^+(u))/2$ for each $u \in U$, and so

$$\sum_{u \in U} e(\Gamma^+(u), S_u) \geq \frac{1}{2} \sum_{u \in U} e(S_u, \Gamma^+(u)) \geq \frac{e^2}{256n}.$$

This last equation means that there are many arcs which can replace the arc in the opposite direction of the previously found almost directed 4-cycles. We count how many directed 4-cycles it gives. For a vertex $u \in U$ an arc wv from $\Gamma^+(u)$ to S_u gives rise to $d^{+-}(v,u)$ 4-cycles. Since $v \in S_u \subset W$, we have that $d^{+-}(v,u) \geq e_v^{(2)}/2n \geq e^2/32n^3$. For each $u \in U$ let $C_4(u)$ be the number of 4-cycles with u in. We have that $C_4(u) \geq e(\Gamma^+(u), S_u)e^2/32n^3$. Summing over vertices $u \in U$, we obtain

$$\sum_{u\in U} C_4(u) \geq \frac{e^2}{32n^3} \sum_{u\in U} e(\Gamma^+(u), S_u) \geq \frac{e^4}{8192n^4}.$$

Since each 4-cycle in D is counted at most four times by this sum we have found at least $e^4/32768n^4$ 4-cycles. □

Remark F.14 The number of directed 4-cycles in a typical digraph (a graph in which everything is average) is of the order e^4/n^4. Hence the number of directed 4-cycles given to us by the above theorem is (up to multiplication by a constant) the largest we could possibly hope for. **In fact any $(e^2/128n^2)$-mixing oriented graph contains the correct order of any orientation of a 4-cycle.**

Theorem F.10 can be reformulated using the notation $mex(n,e,\overrightarrow{C_4})$:

Corollary F.15 *For $n,e \in \mathbb{N}$, we have*

$$mex(n,e,\overrightarrow{C_4}) \geq \frac{1}{32}\frac{e^2}{n^2}.$$

Notice that during the proof, we obtained the same result for almost directed 4-cycles $a\overrightarrow{C_4}$, i.e., oriented 4-cycles with a reversed arc. Hence we have a similar theorem :

Theorem F.16 *For $n,e \in \mathbb{N}$, we have*

$$mex(n,e,a\overrightarrow{C_4}) \geq \frac{1}{32}\frac{e^2}{n^2}.$$

Concerning 4-cycles consisting of two parallels paths of length two, $p\overrightarrow{C_4}$, it follows from Lemma F.9 that the condition $e/2$ mixing is sufficient as soon as $e \geq 2\sqrt{2}n^3/2$. This mixing condition is a lot weaker than the two previous ones but it needs a condition on the number of arcs. Setting $mex_{D:e\geq m}(n,e,H) = min_f\{\forall D$ f-mixing, with n vertices and e edges, $e \geq m : H \subset D\}$, we have :

Corollary F.17 *For $n,e \in \mathbb{N}$, we have*

$$mex_{D:e\geq 2\sqrt{2}n^2}(n,e,p\overrightarrow{C_4}) \geq e/2.$$

Concerning the last type of 4-cycles $s\overrightarrow{C_4}$, a condition on the number of arcs is sufficient ; in fact almost the same condition as for 4-cycles in undirected graphs will be sufficient.

Lemma F.18 *The number of 2-stars (two arcs outgoing the same vertex) in a digraph is at least e^2/n^2.*

This lemma is a direct application of Cauchy-Schwartz inequality. It then follows a corollary.

Corollary F.19 *A digraph with more than $n^{3/2}$ arcs contains a 4-cycle $s\overrightarrow{C_4}$ made of two 2-stars.*

F.1.2 Upper bounds for some $mex(n,e,\overrightarrow{C_{2k}})$

In the previous section we proved that every C_1e^2/n^2-mixing oriented graph contains a 4-cycle (and in fact many). It is natural to ask whether this result would fail if C_1 was replaced by a large constant C_2. We give a class of examples of oriented graphs which are C_2e^2/n^2-mixing but which do not contain 4-cycles. Our class of examples is comprehensive in the following sense : for a fixed large constant C_2, and for any pair of natural numbers n and e, with $e \geq 100n^{3/2}$, there is an oriented graph D with approximately n vertices and approximately e arcs which is C_2e^2/n^2-mixing but does not contain a directed 4-cycle.

F.1.3 Simple graphs with no 4-cycles

In constructing our examples we begin with a simple (undirected) graph G which has many edges but does not contain a 4-cycle. Let q be a prime power. The Erdős-Rényi graph G [ERS64] has $V(G)$ being the set of points of the finite projective plane $PG(2,q)$ over the field of order q (so that $n = q^2+q+1$), and an edge between (x,y,z) and (x',y',z') if and only if $xx'+yy'+zz' = 0$. In fact this implies $e(G) = \frac{1}{2}q(q+1)^2$ and this graph does not contain a 4-cycle. So for all n of the form q^2+q+1 (where q is a prime power) there is a graph G on n vertices with at least $\frac{1}{2}n^{3/2}$ edges which does not contain a 4-cycle. We would like a result which holds for all n. We recall that Bertrand's postulate states that for all $k \geq 2$ there is a prime between k and $2k$, combining this with the above example one may deduce that,

Lemma F.20 *Given $n \geq 2$, there exists a (simple) graph G on n vertices with at least $\frac{1}{20}n^{3/2}$ edges which does not contain a 4-cycle.*

F.1.3.1 First examples

Let $m \in \mathbb{N}$ and let G be a 4-cycle free graph on m vertices with at least $\frac{1}{20}m^{3/2}$ edges. For L the constant of Lemma F.7, by that lemma there exists an oriented graph D obtained by orienting the edges of G which is Lm-mixing.

Setting $n = m$ and $e = e(D) = e(G)$, we have that D is our first example. For D is Lm-mixing and since $e^2/n^2 \geq m^3/400m^2 = m/400$ it follows that D is $400Le^2/n^2$-mixing, and since G does not contain a 4-cycle, certainly D cannot contain any directed 4-cycle.

This example in itself does show that Theorem F.10 is best possible. It shows that the statement would fail if we set $C_1 = 400L$. However this example itself does not show that e^2/n^2 is the best expression we could have in the statement of the theorem. To show this we must give a class of examples of C_2e^2/n^2-mixing oriented graphs which do not contain 4-cycles. These examples are obtained as blow ups of the first examples.

F.1.3.2 Blow-ups

An oriented graph D obeys a strong mixing condition, i.e., is f-mixing for a small value of f, if it contains no large biased subgraphs. We show in this section that a

mixing condition on an oriented graph D' carries over, in a weakened form, to a blow up D of D'. A *blow up* of an oriented graph D' is defined as follows.

Definition F.21 Let D' be an oriented graph on $\{1,...,m\}$ and let $l \in \mathbb{N}$. The l-blow up of D' is the oriented graph D with vertex set $V = V_1 \cup ... \cup V_m$, where the sets V_i are disjoint and each of cardinality l, and with arc set $E(D) = \cup_{ij \in E(D')} B(V_i, V_j)$, where $B(V_i, V_j)$ represents the complete bipartite oriented graph on $V_i \cup V_j$ with all arcs going from V_i to V_j.

The key result we need about blow ups is the following :

Lemma F.22 *Let D' be a $(f, 0.9)$-mixing oriented graph and let D be an l-blow up of D', then D is $16fl^2$-mixing.*

What must we prove ? We must prove that for any pair $A, B \subset V$ with $e(A,B) \geq 16fl^2$ that $e(B,A) \geq e(A,B)/2$. We suppose for contradiction that there are subsets $A, B \subset V$ with $e(A,B) \geq 16fl^2$ and $e(B,A) < e(A,B)/2$.

We use this irregularity between A and B, this bias in the direction from A to B, to find subsets $I, J \subset \{1,...,m\}$ such that in D' we have $e_{D'}(I,J) \geq f$ and $e_{D'}(J,I) \leq 0.9e(I,J)$, which contradicts the mixing property of D'.

This is easy in the case where A and B are both unions of cells of the blow up, in this case we may write $A = \cup_I V_i$ and $B = \cup_J V_j$. We now have,

$$e(I,J) = \frac{e(A,B)}{l^2} \geq \frac{16fl^2}{l^2} = 16f \geq f,$$

while,

$$e(J,I) = \frac{e(B,A)}{l^2} \leq \frac{e(B,A)}{2l^2} = \frac{e(I,J)}{2},$$

which is a contraction of the fact that D' is $(f, 0.9)$-mixing.

For the general case A and B may not be unions of cells of the blow up, to deal with this case we use the bias in the orientation between A and B to find sets $A'', B'' \subset V$ which are unions of cells of the blow and for which there is still a biased orientation between A'' and B''.

Proofof Lemma F.22.

Suppose D is not $16fl^2$-mixing, then there exist subsets $A, B \subset V$ such that $e(A,B) \geq 16fl^2$ and $e(B,A) \leq e(A,B)/2$. There is a relative deficiency in the number of arcs from B to A. We shall focus on the points $x \in A$ which receive many fewer arcs from B than they send to B. We define $A' = \{x \in A : e(\{x\}, B) \geq \frac{2}{3} e(B, \{x\})\}$. Note that $e(B, A \setminus A') \geq \frac{2}{3} e(A \setminus A', B)$ so that,

$$e(A,B) \geq 2e(B,A) \geq 2e(B, A \setminus A') \geq \frac{4}{3} e(A \setminus A', B)$$

so we deduce that $e(A \setminus A', B) \leq \frac{3}{4} e(A,B)$ and so $e(A', B) \geq e(A,B)/4 \geq 4fl^2$.

For each $x \in A'$ we have $e(\{x\},B) \geq \frac{2}{3}e(B,\{x\})$. Let $I = \{i : V_i \cap A' \neq \phi\}$, now if $i \in I$ then there exists $x \in A' \cap V_i$, now for any $y \in V_i$ we know that y has exactly the same neighbours as x and so $e(\{y\},B) \geq \frac{2}{3}e(B,\{y\})$. Defining $A'' = \cup_I V_i$, we have $e(A'',B) \geq e(A',B) \geq 4fl^2$ and $e(B,A'') \leq \frac{2}{3}e(A'',B)$.

We now begin a similar procedure to find B''. Let $B' = \{y \in B : e(A'',\{y\}) > \frac{10}{9}e(\{y\},A'')$, this implies that $e(B \setminus B',A'') \geq \frac{9}{10}e(A'',B \setminus B')$ so that,

$$e(A'',B) \geq \frac{3}{2}e(B,A'') \geq \frac{3}{2}e(B \setminus B',A'') \geq \frac{27}{20}e(A'',B \setminus B')$$

so we deduce that $e(A'',B \setminus B') \leq \frac{20}{27}e(A'',B)$ and so $e(A'',B') \geq \frac{7}{27}e(A'',B) \geq \frac{1}{4}e(A'',B) \geq fl^2$. Let $J = \{j : V_j \cap B' \neq \phi\}$, and set $B'' = \cup_J V_j$. With a similar argument to that given previously we obtain that $e(A'',B'') \geq e(A'',B') \geq fl^2$ and $e(B'',A'') < \frac{9}{10}e(A'',B'')$

This tells us that in D' we have $e(I,J) \geq fl^2/l^2 = f$, while $e(J,I) = e(B'',A'')/l^2 < 9e(A'',B'')/10l^2 = 9e(I,J)/10$. This contradicts the fact that D' is $(f,0.9)$-mixing. And so we have completed our proof that D is $16fl^2$-mixing. □

F.1.3.3 The Class of examples

Let L equal the constant $L_{0.9}$ of Lemma F.7 and let $C_2 = 6400L$. Fix natural numbers m and l. Let G be a 4-cycle free graph on m vertices with at least $m^{3/2}/20$ edges. By Lemma F.7 there exists an oriented graph D' obtained by orienting the edges of G which is $(Lm,0.9)$-mixing.

Properties of D' :

(i) $|D'| = m$,
(ii) $e(D') \geq m^{3/2}/20$,
(iii) D' is $(Lm,0.9)$-mixing,
(iv) D' does not contain a 4-cycle.

We now define D to be an l-blow up of D', we note that D has the following properties :

Properties of D :

(i) $|D| = ml$,
(ii) $e(D) \geq m^{3/2}l^2/20$,
(iii) D is $16Ll^2m$-mixing (for a proof, apply Lemma F.22),
(iv) By inspection D does not contain a directed 4-cycle.

We also have

$$\frac{C_2 e(D)^2}{|D|^2} \geq \frac{C_2(m^{3/2}l^2/20)^2}{(ml)^2} = \frac{C_2 l^2 m}{400} = 16Ll^2m,$$

so that D is $C_2e(D)^2/|D|^2$-mixing.

Hence for each m,l we obtain an oriented graph D which is C_2e^2/n^2-mixing but does not contain a 4-cycle. Furthermore this class of examples is general in the sense that if we fix e_0 and n_0, with $e_0 \geq 100n_0^{3/2}$, then we may find an example D with approximately e_0 arcs and approximately n_0 vertices with D being free of 4-cycles while

being C_2e^2/n^2-mixing, we may do this simply by choosing m to be an integer close to $n_0^4/400e_0^2$, choosing a 4-cycle free graph G with close to $m^{3/2}/20$ edges and choosing l to be close to $400e_0^2/n_0^3$.

Corollary F.23 *There exists some C_2 such that, for $n,e \in \mathbb{N}$, we have*

$$mex(n,e,\overrightarrow{C_4}) \leq C_2 \max\left(\frac{e^2}{n^2}, n\right).$$

Proof. When $e > 100n^{3/2}/20$, the example described above are $C_2\frac{e^2}{n^2}$-mixing. When $e < 100n^{3/2}$, they are C_2n-mixing. □

It gives the same result for almost directed 4-cycles :

Corollary F.24 *There exists some C_2 such that, for $n,e \in \mathbb{N}$, we have*

$$mex(n,e,a\overrightarrow{C_4}) \leq C_2 \max\left(\frac{e^2}{n^2}, n\right).$$

Considering now $p\overrightarrow{C_4}$, the previous example is of no interest. Indeed, as soon as we do the blow up, there are plenty of $p\overrightarrow{C_4}$. However, the previous example is interesting before the blow up since it shows that the condition on the number of arcs can not be dropped. Indeed, the oriented graph D', before the blow-up, is Ln-mixing (which is strictly stronger than $e/2$-mixing), it has n vertices, $\frac{1}{20}n^{3/2}$ arcs and does not contain any $p\overrightarrow{C_4}$.

The following lemma gives upper bounds :

Lemma F.25 *There exists a constant C_2' such that, for $n,e \in \mathbb{N}$, we have*
when $e \leq n^2/2$, $mex(n,e,a\overrightarrow{C_4}) \leq e+1$,
when $e \leq n^{3/2}/20$, $mex(n,e,a\overrightarrow{C_4}) \leq C_2'n$.

Proof. The only point which remains to prove is the first one.

Consider the complete bipartite graph on vertex set $V = V_1 \cup V_2$ with V_1 and V_2 of size $n/2$ and with all arcs going from V_1 to V_2. This oriented graph has no $a\overrightarrow{C_4}$. Finally notice that all oriented graph are $(e+1)$-mixing. □

At this point we have determined, up to a multiplicative constant, the value of $mex(n,e,H)$ when H is any orientation of C_4. The next step is now to determine it when H is a longer cycle. Notice that all cycles of odd length and more general orientation of non bipartite graphs will be treated in Section F.2.

F.1.4 Bounds for $mex(n,e,\overrightarrow{C_{2k}})$

To find a lower bound for $mex(n,e,\overrightarrow{C_4})$, we strongly used the densest C_4-free graph. Hence to generalise this result we need dense C_{2k}-free graphs.

Hypothesis F.26 *For $k \in \mathcal{N}$, there exist C_{2k}-free graphs on n vertices with $cn^{1+1/k}$ edges for some constant c.*

Due to a result of A. Bondy and M. Simonovits [BS74], this hypothesis is best possible. They prove the following theorem, which is usualy called the even circuit theorem :

Theorem F.27 $ex(n, C_{2k}) = O(n^{1+1/k})$

But when do we know that this hypothesis really hold ?

It was conjectured by P. Erdös and M. Simonovits [ES82] that $ex(v, C_{2k}) = 1/2v^{1+1/k} + o(v^{1+1/k})$.

As mentioned before this conjecture is true for $k = 2$, it has also been proved in [NV06] for $k = 3$ and disproved for $k = 5$ by a construction presented in [LUW95] even if it is not mentioned in this paper. Still, for $k = 5$ our hypothsesis is valid for $c \approx 0.58$. For bigger values of k, less is known. The best lower bound is due to F. Lazebnik, V.A. Ustimenko and A.J. Woldar [LUW95] and is $cn^{1+2/(3k)}$, for some c.

We can notice that when all the graphs of even length less than $2k$ are forbidden, the previous lower bound is still valid and a better upper bound has been proved in [LV05], namely $1/2n^{1+1/k} + 2^{k^2}n$. This result is asymptotically tight for $k \in \{2, 3, 5\}$.

Under Hypothesis F.26, Lemma F.22 implies :

Corollary F.28 *Under Hypothesis F.26, let D' be an $(Lm, 0.9)$-mixing $\overrightarrow{C_{2k}}$-free oriented graphs on m vertices with $cn^{1+1/k}$ edges and let l in $\mathbb{N}$ and let D be a blow up of D' where each vertex is replaced by l vertices. There is a constant C independent of l such that D is $C(e^k/n^2)^{1/(k-1)}$-mixing.*

From this lemma we can deduce an upper bound on $mex(n, e, \overrightarrow{C_{2k}})$.

Corollary F.29 *Under Hypothesis F.26, there is some constant C_2 such that $mex(n, e, \overrightarrow{C_{2k}}) \leq C_2 \max((e^k/n^2)^{1/(k-1)}, n)$.*

It remains now to adapt Theorem F.13 to prove a lower bound. We do it for the directed 6-cycle.

Theorem F.30 *i) There exists $C_1 > 0$ such that every $(C_1 e^2/n^2)$-mixing oriented graph D contains a directed 6-cycle.*

ii) There exists $C_1' > 0$ such that every $(C_1' e^2/\log(n)n^2)$-mixing oriented graph D contains at least $ce^6/n^6 \log(n)$ of them (for some constant $c > 0$).

Proof. We do not track the constant during the proof, instead we use a constant C which evolves during the proof. We focus our attention on the set of vertices $W' = \{v \in V : d(v) \geq \frac{e}{16n}\}$ ($d(v)$ is the total degree, i.e., $d^+(v) + d^-(v)$) and on the set $W = \{v \in W' : d(v) \geq \frac{e}{16n}$ *and* $e_v^{(2)} \geq \frac{e^2}{16n^2}\}$.

The number of arcs incident to W' is still large

$$\sum_{v\in W'} d(v) \geq \frac{15e}{16}.$$

Hence, by Lemma F.9, there are many paths of length two from W' to W', namely

$$\sum_{v\in W'} e^{(2)}(W',W') \geq \frac{15}{16e} - \frac{3C_1e^2}{n^2} \geq \frac{15^2e^2}{2\cdot 17^2\cdot n}.$$

It implies that the number of paths of length two starting at W and ending in W' is large

$$\sum_{v\in W} e_v^{(2)}(W,W') \geq \sum_{v\in W'} e^{(2)}(W',W') - \frac{ne^2}{16n^2} \geq \frac{e^2}{3n}.$$

We recall that $d^{+-}(v,u) = |\Gamma^+(v)\cap\Gamma^-(u)|$, and $d^{++}(v,u) = |\Gamma^+(v)\cap\Gamma^+(u)|$. Using the joint degrees, $e(U,\Gamma^+(v))$ (for $v\in V$ and $U\subset V$), which becomes $e(U,\Gamma^+(v)) = \sum_{u\in U} d^{++}(v,u)$. We shall also use that $e(\Gamma^{++}(v),\Gamma^+(v)) = \sum_{u\in\Gamma^{++}(v)} d^{++}(v,u)$.

For vertices $v\in W$, the quantity $e^{(2)}(v) = e(\Gamma^+(v),\Gamma^{++}(v))$ is, by definition, larger than $\frac{e^2}{16n^2}$. By the mixing condition, for each $v\in W$, we have $e(\Gamma^{++}(v),\Gamma^+(v)) \geq \frac{e^{(2)}(v)}{2}$.

If we consider the set $A=\Gamma^+(v)$ and apply Lemma F.8, we get that most of the arcs are incident to a vertex u with $1/2d^{++}(v,u) \leq d^{+-}(v,u) \leq 2d^{++}(v,u)$. We call $T_v = \{u\in\Gamma^{++}(v) : 1/2d^{+-}(v,u) \leq d^{++}(v,u) \leq 2d^{+-}(v,u)\}$.

We have, for all $v\in W$,

$$\begin{aligned}\sum_{u\in T_v} d^{++}(v,u) &\geq \sum_{u\in\Gamma^{++}(v)} d^{++}(v,u) - \frac{3C_1e^2}{n^2}\\ &\geq e(\Gamma^{++}(v),\Gamma^+(v)) - \frac{3C_1e^2}{n^2}\\ &\geq \frac{e^{(2)}(v)}{2} - \frac{3C_1e^2}{n^2} \geq \frac{e^{(2)}(v)}{3}.\end{aligned}$$

Until now, the proof concerned both point *i*) and point *ii*). From now on, we concentrate ourself on point *i*).

To find 6-cycles, we want to find paths of length two from T_v to T_v.

To apply Lemma F.9, we want to estimate $e(T_v,V)$. We have $e(T_v,V) \geq \sum_{u\in T_v} d^+(u) \geq \sum_{u\in T_v} d^+ + (u,v) \geq \frac{e^{(2)}(v)}{3}$. Hence by Lemma F.9 :

$$\begin{aligned}e^{(2)}(T_v,T_v) &\geq \frac{(\frac{e^{(2)}(v)}{3} - \frac{3C_1e^2}{n^2})^2}{2n}\\ &\geq \frac{(e^{(2)}(v))^2}{32n}\\ &\geq \frac{e^4}{32\cdot 16^2\cdot n^5}.\end{aligned}$$

We are only interested in the vertices of T_v which are the end of at least one path of length two from T_v, we note them T'_v. As the vertices of $T_v \setminus T'_v$ receive no path of length two from T_v, in particular they receive none from T'_v. Hence by lemma F.9, $e(T'_v, V) \leq \frac{3C_1e^2}{n^2}$.

Now we change of point of view and we look at the following sum :

$$\sum_u \sum_{v:u\in T'_v} d^{++}(v,u) = \sum_{v\in W} \sum_{u\in T'_v} d^{++}(v,u) \geq \sum_{v\in W} \left(\sum_{u\in T_v} d^{++}(v,u) - \frac{3C_1e^2}{n^2} \right)$$
$$\geq \sum_{v\in W} \left(\frac{e^{(2)}(v)}{3} - \frac{3C_1e^2}{n^2} \right) \geq \frac{e^2}{12n}.$$

There is some u for which $\sum_{v:u\in T'_v} d^{++}(v,u) \geq \frac{e^2}{18n^2}$. For this u, it means there are many arcs from $\{v : u \in T'_v\}$ to $\Gamma^+(u)$, hence by the mixing condition, there is at least an arc from $\Gamma^+(u)$ to a vertex v for which $u \in T'_v$. As $u \in T'_v$, there is a vertex $w \in T_v$ with a path of length two from w to u and, as $w \in T_v$, there is a path of length two from v to w. This a 6-cycle going through u, v and w.

To find this 6-cycle, we only used the condition $\frac{C_1e^2}{n^2}$-mixing. To prove the existence of many 6-cycles, we use a stronger condition, namely $\frac{C'_1e^2}{\log(n)n^2}$-mixing.

When, during the proof of point *i*), we were considering the path of length two from T_v to itself, we discarded vertices of T_v which received none. In fact, the number of 6-cycles each path from T_v to itself gives will depend on the indegree and the outdegree of the start and end vertices. Hence to settle this problem, we divide T_v in subsets $T_v^i = \{u \in T_v : n/2^i < d^{+-}(v,u) \leq n/2^{i-1}\}$ for $1 \leq i \leq \log(n)$.

We write $e_v^{(2)}(i)$ for $e(\Gamma^+(v), T_v^i)$ and I_v for $\{i : e_v^{(2)}(i) \geq e_v^{(2)}/20\log(n)\}$. For $i \in I_v$ we have :

$$e_v^{(2)}(i) = e(\Gamma^+(v), T_v^i) \geq \frac{e_v^{(2)}}{20\log(n)} \geq \frac{C'_1e^2}{n^2}\log(n).$$

Hence, by the strong mixing condition of point *ii*), there are many arcs back which, by Lemma F.9 gives

$$e^{(2)}(T_v^i, T_v^i) \geq \frac{(e(T_v^i, V) - 3\frac{C'_1e^2}{\log(n)n^2})^2}{2n} \geq \frac{(e|T_v^i|/16n - 3\frac{C'_1e^2}{\log(n)n^2})^2}{2n} \geq \frac{|T_v^i|^2e^2}{2\cdot 17^2\cdot n^3}.$$

We set U_v^i for the vertices u which are destination of many paths of length two from $|T_v^i|$, i.e., $U_v^i = \{u \in T_v^i : e^{(2)}(T_v^i, u) \geq \frac{C|T_v^i|\cdot e^2}{100n^3}\}$. There are at most $\frac{(C|T_v^i|\cdot e)^2}{100n^3}$ paths of length two that are not considered, hence $|U_v^i| \geq 9/10|T_v^i|$ (otherwise it would contradict Lemma F.9).

As all vertices of T_v^i have almost the same degree, there are at least $2e_v^{(2)}(i)/5$ arcs from $\Gamma^+(v)$ to U_v^i. Hence, by the mixing condition, there are at least $e_v^{(2)}(i)/5$ arcs from U_v^i to $\Gamma^+(v)$.

Given $i=1,...,l$ we write $g(i)$ for the sum $\sum_{v\in W}\sum_{u\in U_v^i} d^{++}(v,u)$. We have

$$\sum_i g(i) \geq \sum_i \frac{1}{5}\sum_{v\in W} e_v^{(2)}(i) \geq \frac{Ce^2}{n}$$

and

$$\sum_{i\in I} g(i) \geq \frac{Ce^2}{2n}.$$

Given i in I, $\sum_u \sum_{v:u\in U_v^i} d^{++}(v,u) = \sum_{v\in W}\sum_{u\in U_v^i} d^{++}(v,u) = g(i) \geq \frac{10C_1'e^2}{n\log(n)}$. For each u with $\sum_{v:u\in U_v^i} d^{++}(v,u)$ at least $\frac{C_1'e^2}{n^2\log(n)}$, we have many arcs from $\{v : u\in U_v^i\}$ to $\Gamma^+(u)$ hence many arcs from $\Gamma^+(u)$ to $\{v : u\in U_v^i\}$.

Each arc gives many 6-cycles. Indeed, recall that u is the end of at least $\frac{C|T_v^i|\cdot e^2}{100n^3}$ path of lengths two and for each of these paths, the starting vertex w has joint degree $d^{+-}(v,w) \geq \frac{e(\Gamma^+(v),T_v^i)}{2|T_v^i|} \geq \frac{e^2}{(20\cdot 16\cdot n^2\log(n))}$.

Hence, each arc from $\Gamma^+(u)$ to $\{v : u\in U_v^i\}$ gives at least $\frac{Ce^4}{n^5\log(n)}$ 6-cycles. The sum (over u with $\sum_{v:u\in U_v^i} d^{++}(v,u)$ at least $\frac{C_1'e^2}{n^2\log(n)}$) of the number of arcs $\Gamma^+(u)$ to $\{v : u\in U_v^i\}$ is at least $Cg(i)$ which gives $\frac{Ce^4 g(i)}{n^5\log(n)}$ 6-cycles. We can do the same thing for each $i\in I$ which in total gives $\frac{Ce^6}{n^6\log(n)}$ 6-cycles. Notice that we counted a 6-cycle at most six times. □

Corollary F.31 *There exist a constant C_1 such that $mex(n,e,\overrightarrow{C_6}) \geq \frac{C_1e^2}{n^2}$.*

F.2 On the value of $mex(n,e,H)$ in dense digraphs

In the previous section we gave results on the value of mex for cycles of even length. We did it in a general setting and no assumptions on the underlying graph were taken. However, to consider more complicated types of patterns, i.e., more complicated digraph $H\in\mathcal{H}$, a possibility is to take more hypothesis on the underlying graph. In this section we consider the case when the underlying graph issufficiently dense, i.e., it has more than βn^2 edges, for some β. Theorem F.33 says that the weakest mixing condition (linear in e) is sufficient to force such an oriented graph to have H as subgraph. However, when H is non bipartite, it is essential to have D sufficiently dense. Indeed in Proposition F.32, we show that there exist digraphs with $n^2/4$ edges satisfying the strongest mixing condition we can hope for regular graph (linear in n) which contains no non bipartite oriented graphs.

Proposition F.32 *Let H be a non bipartite oriented graph on h vertices with chromatic index k, for all n, for the constant L of Lemma F.7, there exist oriented graphs D with $(1-1/(k-1))n^2/2$ arcs which are Ln-mixing and do not contain any copy of H.*

Proof. Just consider a the Turan's graph T_{k-1} oriented at random. Lemma F.7 says it is Ln-mixing for some constant L. □

Theorem F.33 *For every h, every oriented graph H on h vertices, for every $\gamma \in]0,1[$, there exist $\varepsilon > 0$, $0 < \beta < \frac{1}{2}$, and an integer N such that every digraph D on $n \geq N$ vertices, with βn^2 arcs which is $(\varepsilon e, \gamma)$-mixing contains H as a subgraph. When H is bipartite, β can be chosen if n is large enough.*

Corollary F.34 *Given an oriented graph H on h vertices, there exist $\varepsilon > 0$, $0 < \beta < \frac{1}{2}$, and an integer N such that*

$$mex_{D:e \geq \beta n^2}(n,e,H) \geq \varepsilon e.$$

Lemma F.35 *Given an oriented graph H on h vertices which contains a directed path of length two,*

$$mex_{D:e \leq n^2/4}(n,e,H) \leq e+1.$$

Proof. This lemma is proved by the same oriented graph as the one which proved Lemma F.25.

Consider the complete bipartite graph on vertex set $V = V_1 \cup V_2$ with V_1 and V_2 of size $n/2$ and with all arcs going from V_1 to V_2. This oriented graph is H-free and is $e+1$-mixing. □

The proof of Theorem F.33 is based on the use of Szemerédi Lemma so we first state it with the definitions we need.

Definition F.36 *Let U and W be disjoint subsets of the vertices of some graph. The number of edges between U and W (without taking into account orientation) is denoted by $\overline{e}(U,W)$. The density of the pair U,W is $d(U,W) = \frac{\overline{e}(U,W)}{|U||W|}$.*

Definition F.37 *Let $0 < \sigma < 1$. The pair (U,W) is σ-uniform if, for all $U' \subset U$ and $W' \subset W$ with $|U'| > \sigma|U|$ and $|W'| > \sigma|W|$ we have $|d(U',W') - d(U,W)| < \sigma$.*

A simple consequence of σ-uniformity is that the bipartite subgraph must be roughly regular.

Lemma F.38 *Let (U,W) be σ-uniform and let $d(U,W) = d$. Then $|\{u \in U : |\Gamma(u) \cap W| > (d-\sigma)|W|\}| \geq (1-\sigma)|U|$ and $|\{u \in U : |\Gamma(u) \cap W| < (d+\sigma)|W|\}| \geq (1-\sigma)|U|$*

Definition F.39 *An equipartition of $V(G)$ into m parts is a partition $V_1, \ldots, V_m$ such that $\lfloor \frac{n}{m} \rfloor \leq |V_i| \leq \lceil \frac{n}{m} \rceil$, for $1 \leq i \leq m$, where $n = |V(G)|$. The partition is σ-uniform if (V_i, V_j) is σ-uniform for all but $\sigma\binom{m}{2}$ pairs $1 \leq i < j \leq m$.*

Lemma F.40 ([Sze75]) *Let $0 < \sigma < 1$ and let l be a natural number. Then there exists $L = L(l,\sigma)$ such that every graph has a σ-uniform equipartition into m parts for some $l \leq m \leq L$.*

Lemma F.41 *Let H be an oriented graph on h vertices and $\gamma \in]0,1[$. Let D a digraph, with e arcs, containing disjoint vertex sets $V_1,\dots,V_h$ of size u or $u+1$ such that for each $(v_i,v_j) \in E(H)$, (V_i,V_j) is σ-uniform, $d(V_i,V_j) \geq \eta$ and suppose that $D(V_i \cup V_j)$ is $(\varepsilon e,\gamma)$-mixing where $\varepsilon = \frac{\sigma(\eta/(1+1/\gamma)-\sigma)}{h^2}$. Furthermore we suppose $(\eta/(1+1/\gamma))^h \geq (h+1)\sigma$, $\sigma/\eta \leq 1$ and $\sigma u \geq 1$. Then D contains a subgraph isomorphic to H.*

Proof. We call the vertices of H $\{v_1,\dots,v_h\}$.

We prove this statement by induction, we claim that the following property holds for $0 \leq l \leq h$:

$\mathcal{P}(l)$: $x_1,\dots,x_l$ can be chosen so that $x_i \in V_i$ and, for $l < j \leq h$ there is a set $X_j^l \subset V_j$ of candidates for x_j at stage l, meaning that $x_i y_j \in E(D)$ for every $y_j \in X_j^l$ and every $x_i \in N(j,l) = \{xi : 1 \leq i \leq l \text{ and } v_i v_j \in E(H)\}$. Moreover $|X_j^l| \geq (\eta/(1+1/\gamma)-\sigma)^{N(j,l)}u$.

$\mathcal{P}(0)$ clearly holds, just take $X_j^0 = V_j$.

Suppose $\mathcal{P}(l)$ true, we want to prove $\mathcal{P}(l+1)$.

T^+ represents the set of indices j, for which we have an outgoing arcs $v_{l+1}v_j$ and so we will have to be careful to have arcs from the vertex x_{l+1} we will select to the vertices of the set X_j^{l+1}. Y_j is the set of vertices of X_{l+1}^l which does not have enough arcs to X_j^l and hence that are not good candidates for x_{l+1}.

For each $t \in T^+ = \{j > l+1 : v_{l+1}v_j \in E(H)\}$, let $Y_t = \{y \in X_{l+1}^l : |\Gamma^+(y) \cap X_t^l| \leq (\eta/(1+1/\gamma)-\sigma)|X_t^l|\}$.

T^- is the pendent of T^+ for ingoing arcs.

For each $t \in T^- = \{j > l+1 : v_j v_{l+1} \in E(H)\}$, let $Y_t = \{y \in X_{l+1}^l : |\Gamma^-(y) \cap X_t^l| \leq (\eta/(1+1/\gamma)-\sigma)|X_t^l|\}$.

By $\mathcal{P}(l)$ and the hypothesis, we have : $|X_t^l| \geq (\eta/(1+1/\gamma)-\sigma)^{N(j,l)}u \geq (\eta/(1+1/\gamma)-\sigma)^h u \geq \sigma u$.

It follows from Definition F.37 and the $(\varepsilon e,\gamma)$-mixing condition that for all $t \in T^+ \cup T^-$, $|Y^t| \leq \sigma u$.

Indeed, for $t \in T^+$:

- if $e(Y_t,X_t^l) < \varepsilon e$, $(\eta/(1+1/\gamma)-\sigma)|X_t^l||Y_t| < \varepsilon e$ so $|Y_t| < \frac{\varepsilon e}{(\eta/(1+1/\gamma)-\sigma)u} \leq \sigma u$.
- if $e(Y_t,X_t^l) \geq \varepsilon e$ then by the mixing condition, $e(X_t^l,Y_t) \geq \gamma e(Y_t,X_t^l)$. If $e(Y_t,X_t^l) \geq \varepsilon e$, then $e(Y_t,X_t^l) \geq \gamma e(X_t^l,Y_t)$, and if $e(Y_t,X_t^l) \leq \varepsilon e$, then $e(Y_t,X_t^l) \geq e(X_t^l,Y_t)$ which also gives $e(Y_t,X_t^l) \geq \gamma e(X_t^l,Y_t)$. So $\overline{e}(X_t^l,Y_t) = e(X_t^l,Y_t) + e(X_t^l,Y_t) \leq (1+1/\gamma)(\eta/(1+1/\gamma)-\sigma)|X_t^l||Y_t|$, hence, $d(X_t^l,Y_t) \leq (1+1/\gamma)(\eta/(1+1/\gamma)-\sigma) \leq (\eta-\sigma)$ and so by Definition F.37, $|Y_t| \leq \sigma u$

We have the same result for $t \in T^-$.

Therefore $|X_{l+1}^l \setminus (\cup_{t\in T^- \cup T^+} Y^t)| \geq (\eta/(1+1/\gamma)-\sigma)^{N(l+1,l)}u - |T^- \cup T^+|\sigma u \geq ((\eta/(1+1/\gamma))^{N(l+1,l)} - h\sigma)u \geq \sigma u \geq 1$. So we can select $x_{l+1} \in X_{l+1}^l \setminus (\cup_{t\in T^- \cup T^+} Y^t)$.

It is now sufficient to take $X_t^{l+1} = X_t^l \cap \Gamma(x_{l+1})$ for $t \in T^- \cup T^+$ and $X_t^{l+1} = X_t^l$ for $t \notin T^- \cup T^+$ to obtain that $\mathcal{P}(l+1)$ is true.

The above shows that the claim holds for all $l, 0 \leq l \leq h$ so $x_1,\dots,x_h$ can be found as desired. □

Proof of Theorem F.33. We consider an oriented graph H on h vertices, $\gamma \in]0,1[$ and a digraph D on n vertices with $e = \beta n^2$ (β to be fixed later) arcs which is $(\varepsilon e, \gamma)$-mixing. We set l sufficiently big such that a graph G on l vertices with $ex(\overline{H}, l)$ edges contains H.

We now apply Szemerédi regularity Lemma to $\overline{D}$ for l and some σ. We obtain a σ-uniform equipartition of $V(G)$ into m parts with $l \leq m \leq L : V_1, \ldots, V_m$. We require β to satisfy :

$$\beta n^2 \geq \eta n^2/2 + ex(m, L)(n/m)^2$$

. When H is non bipartite, it gives a lower bound on β, when H is bipartite, we can choose β as small as we want as long as η is also choosen small (it just implies n to be big enough).

From this partition we construct a graph G, each part V_i of the partition of $\overline{D}$ is represented by a vertex v_i in G and two vertices of G are linked together if the corresponding parts have a density of at least η in between them (i.e. $(v_i, v_j) \in E(G)$ iff $d(V_i, V_j) \geq \eta$) for some $0 < \eta < 1$ which satisfies the hypothesis of Lemma F.41. As long as η is not too big, there is some β that ensures us that $e(G) \geq ex(m, \overline{H})$. Consequently we get a copy of $\overline{H}$ in G, wlog it uses the vertices $v_1, \ldots, v_h$.

Lemma F.41 with appropriate values of ε, η and σ gives the result. □

A result from N. Alon and A. Shapira [AS04] says that if we have an oriented graph such that we have to remove εn^2 arcs before it becomes H-free, then it has many copies of H, the exact statement of the theorem is the following :

Theorem F.42 ([AS04]) *For every fixed ε and h, there is a constant $c(\varepsilon, h)$ with the following property : for every fixed digraph H of size h, and for every digraph D of a large enough size n, from which we need to delete εn^2 arcs to make it H-free, D contains at least $c(\varepsilon, h)n^h$ copies of H.*

From this theorem and Theorem F.33, we may deduce the following Corollary :

Corollary F.43 *For every h, every oriented graph H on h vertices, there exist $\varepsilon > 0$, $0 < \beta < \frac{1}{2}$, $c(\varepsilon, h, \beta)$ and an integer N such that every digraph D on $n \geq N$ vertices, with more than βn^2 arcs which is εe-mixing contains $c(\varepsilon, h)n^h$ copies of H.*

Proof. Let h be an integer and H an oriented graph on h vertices, Theorem F.33 gives ε, β and N such that every digraph D on $n \geq N$ vertices, with βn^2 arcs which is $(\varepsilon e, 0.4)$-mixing contains H as a subgraph.

Setting $\varepsilon' = \varepsilon\beta/10$, every digraph D on $n \geq N$ vertices, with $(\beta + \varepsilon')n^2$ arcs which is $(\varepsilon e, 0.5)$-mixing contains H as a subgraph. On top of this, if we delete $\varepsilon' n^2$ arcs, it still contains H as a subgraph. Indeed it has more than βn^2 arcs, and we can check it is $(\varepsilon e, 0.4)$-mixing. Hence we can apply Theorem F.42 which says that it contains $c(\varepsilon', h)n^h$ copies of H.

Notice that if $\beta + \varepsilon' \geq 1/2$, we can take ε' smaller to go bellow $1/2$. □

This corollary says that given h, there exist $\varepsilon > 0$, $0 < \beta < \frac{1}{2}$, $c(\varepsilon, h, \beta)$ and an integer N such that every digraph D on $n \geq N$ vertices, with βn^2 arcs which is εe-mixing contains the **correct order of any orientation of any graph on h vertices**.

F.3 $\max_D^{AB}$

Lemma F.7 says that, if C is a large constant, randomly oriented graphs are Cn-mixing with high probability. It is now natural to wonder what is the strongest mixing condition we can ask for. In general it seems really hard to determine how much an oriented graph is mixing, however, the presence of sets of vertices A and B with arcs from A to B and none from B to A ($e(A,B) > 0$ and $e(B,A) = 0$) is a natural obstruction for an oriented graph to be highly mixing and is simpler to analyse. We define more precisely this problem in next section together with a complexity result. In Section F.3.2 we give lower bounds for this number of arcs.

F.3.1 Definition and complexity result

To better understand how mixing we can ask for a digraph to be, we ask the following general question :

Given an oriented graph D, what is the maximum of arcs we can have from a set to an other with no arcs backward, i.e., what is :

$$\max_D^{AB} = \max_{\{A,B:e(B,A)=0\}} (e(A,B))?$$

In fact this question can be asked for digraphs in general and not only for oriented graphs, i.e., we can ask this question when multiple arcs and 2-cycles are allowed. In this case, we have the following complexity result :

Theorem F.44 *Given a digraph, potentially with 2-cycles, computing $\max_D^{AB}$ is NP-hard and hard to approximate within a factor $n^{1/2-\varepsilon}$ for any $\varepsilon > 0$ unless $P = NP$.*

Proof. We prove this theorem by reducing the stable set problem to our problem. Recall that the stable set problem consists in finding a set as large as possible of independent vertices in a graph.

Given a graph $G = (V,E)$, we construct a bipartite digraph as follow :

- It has vertex set $V_1 \times V_2 = V \times V$.
- Given an arc vw of G, we put a 2-circuit between the two vertices $v \in V_1$ and $w \in V_2$.
- Given $v \in V$, we put an arc from $v \in V_1$ to $v \in V_2$.

It is now easy to see that a solution to $\max_D^{AB}$, i.e., two sets A and B with $e(A,B) = \max_D^{AB}$ and $e(B,A) = 0$, gives a solution to the stable set problem. Indeed each arc from A to B corresponds to a vertex and all the corresponding vertices form a stable set since otherwise there would be a 2-circuit with one end vertex in A and the other one in B. This stable set has to be maximal otherwise we could increase $\max_D^{AB}$.

It follows from a result of J. Håstad in [Hås99] that $\max_D^{AB}$ is NP-hard and hard to approximate within a factor $n^{1/2-\varepsilon}$ for any $\varepsilon > 0$ unless $P = NP$. □

F.3.2 Lower bounds on $\max_D^{AB}$

In a digraph, $\max_D^{AB}$ can be equal to zero if all arcs are included in a 2-cycle, but this is strongly due to the presence of 2-cycles. What happens when the digraph has no 2-cycles ? We prove that for **d-regular oriented graphs** (oriented graph such that for all vertices $v \in V$ we have $d^-(v) = d^+(v) = d$), $\max_D^{AB}$ is at least linear in n. In particular, this means that d-regular oriented graphs cannot be εn-mixing for small ε. To prove it, we construct two sets A and B with many arcs from A to B and none from B to A. We do this using a greedy algorithm.

In fact we just need to construct A since B can be chosen to maximize $e(A,B)$ while keeping $e(B,A) = 0$, i.e., given A, we take $B(A) = \{v : e(\{v\},A) = 0\}$.

1: Let $v \in V$, set $A = \{v\}$ and $B = \Gamma^+(v)$.
2: For all $u \in V \setminus A$ evaluate the function :
$f(u) = \sum_{v \in A} \Big(1_{uv \in E} + 1_{vu \in E} + |\Gamma^+(v) \cap \Gamma^-(u)| + |\Gamma^+(u) \cap \Gamma^-(v)| \Big)$.
3: **while** there is a vertex u with $f(u) < d^+(u)$ **do**
4: Add to A the vertex v which maximizes $d^+(v) - f(v)$.
5: Update $B : B = B(A)$.
6: Update $f(u)$ for all $u \in V \setminus A$.
7: **end while**
8: **return** A and B.

Using this algorithm on d-regular oriented graphs, one can prove :

Lemma F.45 *Let D be a d-regular oriented graph, we can construct subsets $A, B \subset V$ with $e(A,B) \geq \frac{n(d+1)}{4d+6} - d - 1$ and $e(B,A) = 0$ in polynomial time. In particular D is emphatically not $\frac{n(d+1)}{4d+6} - d - 1$-mixing.*

Proof. Using the previous algorithm, we define a sequence of pairs of subsets $A_t, B_t \subset V$, with the property that $e(B_t, A_t) = 0$ for all t. We attempt to do this while increasing the value of $e(A_t, B_t)$ so that for some value of t we have $e(A_t, B_t) \geq \frac{n(d+1)}{4d+6} - d - 1$.

A_t is defined inductively adding one vertex at each step so that $|A_t| = t$.

First choose any $x_1 \in V$, set $A_1 = \{x_1\}$ and $B_1 = B(A_1)$.

We now look for a suitable candidate y to join $A_t = \{x_1, ..., x_t\}$, it is unhelpful if y is joined to A (in either direction) or if there are many paths of length two between them (in either direction). It is why we define the function $f_t : V \setminus A_t \to \mathbb{N}$ by

$$f_t(y) = \sum_{i=1}^{t} \Big(1_{yx_i \in E} + 1_{x_i y \in E} + |\Gamma^+(x_i) \cap \Gamma^-(y)| + |\Gamma^+(y) \cap \Gamma^-(x_i)| \Big).$$

Counting arcs involving A_t and using regularity we have that, $\sum_{y\notin A_t} f_t(y) \leq t(2d^2+2d)$, so that there exists y with $f_t(y) \leq \left\lfloor \frac{2td(d+1)}{n-t} \right\rfloor$. Hence the x_{t+1} chosen by the algorithm is such that $f_t(x_{t+1}) \leq \left\lfloor \frac{2td(d+1)}{n-t} \right\rfloor$. We now let $A_{t+1} = \{x_1,...,x_{t+1}\}$ and $B_{t+1} = B(A_{t+1})$

We wish to calculate $e(A_{t+1},B_{t+1})$.

It is clear that by adding x_{t+1} to A_t, we add d outgoing arcs but some of the added arcs do not count, how many ? Precisely $\sum_{i=1}^{t}\left(1_{x_{t+1}x_i\in E} + |\Gamma^+(x_{t+1})\cap\Gamma^-(x_i)\right)$.

Furthermore, we have $B_{t+1}\subset B_t$, hence some arcs that we were counting before do not count anymore, how many arcs do we have to remove ? There are exactly $\sum_{i=1}^{t}\left(1_{x_ix_{t+1}\in E} + |\Gamma^+(x_i)\cap\Gamma^-(x_{t+1})|)\right)$ arcs we were counting before that we can not count anymore.

In total it gives $e(A_{t+1},B_{t+1}) = e(A_t,B_t)+d-f_t(x_{t+1}) \geq e(A_t,B_t)+d-\left\lfloor \frac{2td(d+1)}{n-t} \right\rfloor$ and of course $e(B_{t+1},A_{t+1})=0$. It is now trivial by induction that $e(A_t,B_t) \geq td - \sum_{i=0}^{t}\left\lfloor \frac{2id(d+1)}{n-i} \right\rfloor$ for all $t \leq \frac{n}{2d+3}$. The algorithm goes until at least $t = \left\lfloor \frac{n}{2d+3} \right\rfloor$ and we have :

$$
\begin{aligned}
e(A_t,B_t) &\geq td - \sum_{i=0}^{\left\lfloor \frac{n}{2d+3} \right\rfloor} \left\lfloor \frac{2id(d+1)}{n-i} \right\rfloor \qquad \text{(F.1)}\\
&\geq td - \sum_{i=0}^{d-1} i*\left(\left\lceil \frac{(i+1)n}{2d(d+1)+i+1} \right\rceil - \left\lceil \frac{in}{2d(d+1)+i} \right\rceil\right)\\
&= td - \left\lceil \frac{d(d-1)n}{2d(d+1)+d} \right\rceil + \sum_{1}^{d-1}\left\lceil \frac{in}{2d(d+1)+i} \right\rceil\\
&\geq \frac{nd}{2d+3} - d - \left\lceil \frac{d(d-1)n}{2d(d+1)+d} \right\rceil + \frac{nd(d-1)}{2(2d(d+1)+d-1)}\\
&\geq \frac{nd}{2d+3} - d - \frac{(d-1)n}{2d+3} - 1 + \frac{n(d-1)}{2(2d+3)}\\
&\geq \frac{nd}{2d+3} - \frac{n(d-1)}{4d+6} - d - 1\\
&\geq \frac{n(d+1)}{4d+6} - d - 1
\end{aligned}
$$

And as we have already mentioned, by the choice of B_t, $e(B_t,A_t)=0$. □

This result shows that there exist two sets A and B with many arcs from A to B and none in the other direction. Furthermore it gives a way to construct this two sets. If we are not interested in finding the sets A and B, we can use a probabilistic argument which gives a slightly better result :

Lemma F.46 *Let D be a d-regular oriented graph, there exist subsets $A,B\subset V$ with $e(A,B)\geq\frac{n}{4}$ and $e(B,A)=0$. So that D is emphatically not $\frac{n}{4}$-mixing.*

Having chosen a set $A \subset V$ there is an obvious choice for B. We should choose $B = \{y \in V \ : \ e(\{y\},A) = 0\}$, as this choice of B maximises $e(A,B)$ under the condition that $e(B,A) = 0$. For this choice of B, we have $e(A,B) \geq \sum_{x\in A} d^+(x) - e^{(2)}(A)$, where $e^{(2)}(A)$ is the number of paths of length two from A to A. Indeed, since $\sum_{x\in A} d^+(x) = e(A,B) + e(A,V \setminus B)$, we only need to show that $e(A,V \setminus B) \leq e^{(2)}(A)$, which is obvious since :

$$e^{(2)}(A) = \sum_{x\in A} \sum_{y\in\Gamma^+(x)} |\Gamma^+(y) \cap A| \geq \sum_{x\in A} |\Gamma^+(x) \cap (V \setminus B)| = e(A,V \setminus B)$$

So to prove the proposition we just need to find a set $A \subset V$ for which $\sum_{x\in A} d^+(x) - e^{(2)}(A) \geq n/4$.

Proof. We must find $A \subset V$ for which $\sum_{x\in A} d^+(x) - e^{(2)}(A) \geq n/4$. We pick A at random, each vertex v being in A independently with probability p. What is $\mathbb{E}(\sum_{x\in A} d^+(x) - e^{(2)}(A))$? From regularity it follows that $\mathbb{E}(\sum_{x\in A} d^+(x)) = d\mathbb{E}(|A|) = dpn$. While $\mathbb{E}(e^{(2)}(A))$ is given by the number of paths of length two in the graph (which is d^2n by regularity) multiplied by the probability that both end points lie in A (which is p^2), so that $\mathbb{E}(e^{(2)}(A)) = d^2p^2n$. By linearity of expectation,

$$\mathbb{E}(\sum_{x\in A} d^+(x) - e^{(2)}(A)) = dpn - d^2p^2n.$$

Setting $p = 2/d$ gives us $\mathbb{E}(\sum_{x\in A} d^+(x) - e^{(2)}(A)) = n/4$, and so there exists a set $A \subset V$ for which $\sum_{x\in A} d^+(x) - e^{(2)}(A) \geq n/4$ and we are done. □

The above methods can be easily modified to adapt to the **k-almost d-regular oriented graphs** :

Definition F.47 *an oriented graph is k-almost d-regular if for all vertices $v \in V$ we have $d \leq d^-(v)$ and $d^+(v) \leq kd$,*

Lemma F.48 *Let $d \in \mathbb{N}$, and let D be an k-almost d-regular oriented graph, we can construct subsets $A,B \subset V$ with $e(A,B) \geq \frac{n(d+1)}{4k^2(d+1)+2} - 1 - d$ and $e(B,A) = 0$ in polynomial time. In particular D is emphatically not $\frac{n(d+1)}{4k^2(d+1)+2} - 1 - d$-mixing.*

Proof. We proceed exactly as for the proof of Lemma F.45, but now degrees may vary by a factor of k and squares of degrees (or for our purposes the number of paths of length 2 coming from a point) may vary by a factor of k^2, so that we can only guarantee the existence of y with $f_t(y) \leq \left\lfloor \frac{2k^2td(d+1)}{n-t} \right\rfloor$. So by induction $e(A_t,B_t) \geq td - \sum_{i=0}^{t} \left\lfloor \frac{2k^2id(d+1)}{n-i} \right\rfloor$, setting $t = \left\lfloor \frac{n}{k^2(2d+2)+1} \right\rfloor$ we obtain $e(A_t,B_t) \geq \frac{n(d+1)}{4k^2(d+1)+2} - 1 - d$. □

Using a probabilistic argument, we get :

Lemma F.49 *Let D be an k-almost d-regular oriented graph, there exist subsets $A,B \subset V$ with $e(A,B) \geq \frac{n}{4k^2}$ and $e(B,A) = 0$. So that D is emphatically not $\frac{n}{4k^2}$-mixing.*

Now we note that if we randomly orient a random graph then with high probability it will be 2-almost d-regular for some d. Hence we may deduce :

Corollary F.50 *Let G be a random graph on n vertices with each edge included independently with probability p (this is usual denoted $G(n,p)$) and let D be obtained from G by orienting its edges at random. If $p = \omega(\log n/n)$ then w.h.p. there exist subsets $(A,B) \subset V(G)^2$ with $e(A,B) \geq \frac{n}{16}$ and $e(B,A) = 0$. So that w.h.p. D is emphatically not $\frac{n}{16}$-mixing when n is large.*

Proof. Two applications of the Chernoff inequality show that with high probability all the degrees (out-degrees and in-degrees) are between $0.7pn$ and $1.4pn$ and so we are done by applying Lemma 4.2 with $d = 0.7pn$. □

This results have been completed by S. Griffiths in [Gri07], in which he proves that if an oriented graph D has no isolated vertices then $\max_D^{AB} \geq n/8\log(n)$ and that this result is best possible up to a constant factor by producing an example of a random oriented graph which with high probability has no isolated vertices and for which $\max^{AB} \leq Cn/\log(n)$ for C a large constant.

F.4 Conclusion and Remaining questions

There are many questions one could ask concerning mixing properties. The most immediate is to ask what are the exact results for cycles other than 4-cycles. Perhaps the most interesting case to consider next is the oriented 6-cycle. Theorem F.30 in which we show that there is a constant $\varepsilon > 0$ for which every $\varepsilon e^2/n^2$-mixing oriented graph contains a 6-cycle may not be the best possible. The upper bound we gave for $mex(n, e\overrightarrow{C_6})$ is $Ce^{3/2}/n$ for a large constant C. We believe that this is close to being best possible. Specifically we conjecture,

Conjecture F.51 *For, n and e, there exists $\varepsilon > 0$, such that every $\varepsilon e^{3/2}/n$-mixing oriented graph on n vertices and e edges, contains a directed 6-cycle. When $e >> n^{4/3}$, it would imply $mex(n, e\overrightarrow{C_6}) = \Omega(e^{3/2}/n)$.*

Similar conjectures can be made for even cycles of length $2k$ greater than 6, they would simply say that the example obtained by taking a blow up of a random orientation of the largest graph of girth greater than $2k$ is (up to multiplication by constant) the most mixing oriented graph which does not contain a $2k$-cycle.

More generally for a fixed oriented graph F, what mixing condition is required to ensure D contains a copy of F ? This question has been answered when the underlying graph is dense, but what happen if it is not dense ? What can we say about any orientation of a tree, is there some class of oriented graph for which it is easy to answer this question ?

Bibliographie

Ch. 2 : Notions de base

[CLR05] M.C. Costa, L. Létocart, and F. Roupin. Minimal multicut and maximal integer multiflow : A survey. *European Journal of Operational Research*, 162(1) :55–69, 2005.

[CR02] D. Coudert and H. Rivano. Lightpath assignment for multifibers wdm networks with wavelength translators. In *Global Telecommunications Conference, 2002. GLOBECOM '02. IEEE*, volume 3, pages 2686–2690vol.3, 17-21 Nov. 2002. OPNT-01-5.

[CRR03] D. Coudert, H. Rivano, and X. Roche. A combinatorial approximation algorithm for the multicommodity flow problem. In K. Jansen and R. Solis-Oba, editors, *WAOA 03*, number 2909 in Lecture Notes in Computer Science, pages 256–259, Budapest, Hungary, September 2003. Springer-Verlag.

[DDS05] G. Desaulniers, J. Desrosiers, and M.M. Solomon. *Column Generation*. Kluwer Academic Publishers, Boston, MA, 2005.

[Fle00] L.K. Fleischer. Approximating Fractional Multicommodity Flow Independent of the Number of Commodities. *SIAM Journal on Discrete Mathematics*, 13(4) :505–520, 2000.

[GVY93] N. Garg, V.V. Vazirani, and M. Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. In *ACM Symposium on Theory of Computing*, pages 698–707, 1993.

[JMT05] B. Jaumard, C. Meyer, and B. Thiongane. On Column Generation Formulations for the RWA Problem. *International Network Optimization Conference (INOC'O5), Lisbon, Portugal*, pages 20–23, March 2005.

[Rag88] P. Raghavan. Probabilistic construction of deterministic algorithms : approximating packing integer programs. *Journal of Computer and System Sciences*, 37(2) :130–143, 1988.

Ch. 3 : Conception de réseaux tolérants aux pannes

[ABG+06] O. Amini, J.-C. Bermond, F. Giroire, F. Huc, and S. Pérennes. Design of minimal fault tolerant networks : Asymptotic bounds. In *Huitièmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel'06)*, pages 37–40, Trégastel, France, May 2006.

[AC03] N. Alon and M. Capalbo. Smaller explicit superconcentrators. *Proceedings of the Fourteenth Annual Symposium on Discret Algorithms ACM-SIAM (SODA)*, pages 340–346, 2003.

[AGHP] O. Amini, F. Giroire, F. Huc, and S. Pérennes. Minimal selectors and fault tolerant networks. *Networks*. To appear.

[AGM87] N. Alon, Z. Galil, and V.D. Milman. Better expanders and superconcentrators. *Journal of Algorithms*, 8 :337–347, 1987.

[AHK99] N. Alon, P. Hamburger, and A.V. Kostochka. Regular honest graphs, isoperimetric numbers, and bisection of weighted graphs. *European Journal of Combinatorics*, 1999.

[AHU74] A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1974.

[Alo86] N. Alon. Eigenvalues and expanders. *Combinatorica*, 6(2) :83–96, 1986.

[Alo93] N. Alon. On the edge-expansion of graphs. *Combinatorics, Probability and Computing*, 11 :1–10, 1993.

[AM84] N. Alon and V.D. Milman. Eigenvalues, expanders and superconcentrators. *Proceedings of 25th Annual Symposium on Fondations of Computer Science (FOCS)*, pages 320–322, October 1984.

[BDD02] J-C. Bermond, E. Darrot, and O. Delmas. Design of fault tolerant on-board networks in satellites. *Networks*, 40 :202–207, 2002.

[BGP07] J-C. Bermond, F. Giroire, and S. Pérennes. Design of Minimal Fault Tolerant On-Board Networks : Practical constructions. In *14th International Colloquium on Structural Information and Communication Complexity (SIROCCO 07)*, volume 4474 of *Lecture Notes in Computer Sciences*, pages 261–273, Castiglioncello, Italy, June 2007.

[BHT06] J-C. Bermond, F. Havet, and D. Tóth. Fault tolerant on board networks with priorities. *Networks*, 47(1) :9–25, 2006.

[BL06] Y. Bilu and N. Linial. Lifts, discrepancy and nearly optimal spectral gaps. *Combinatorica*, 26(5) :495–519, 2006.

[Bol88] B. Bollobás. The isoperimetric number of random regular graphs. *European Journal of Combinatorics*, 9(3) :241–244, 1988.

[Chu97] F.R.K. Chung. *Spectral graph theory Regional conference series in mathematics*, volume 92. American Mathematical Society, 1997.

[CRVW02] M. Capalbo, O. Reingold, S. Vadhan, and A. Wigderson. Randomness conductors and constant-degree lossless expanders. In *34th ACM Symposium on Theory of Computing (STOCS)*, pages 659–668, 2002.

[Dar97] E. Darrot. *Diffusion de messages longs et propriétés structurelles dans les réseaux d'interconnexion*. PhD thesis, École doctorale STIC, Université de Nice-Sophia Antipolis, June 1997.

[DHMP06] O. Delmas, F. Havet, M. Montassier, and S. Pérennes. Design of fault tolerant on-board networks. Research report, INRIA Research Report 5866, March 2006.

[DN01] D-Z. Du and H.Q. Ngo. *Notes on the complexity of switching networks*, pages 307–367. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2001.

[DSV03] G. Davidovf, P. Sarnak, and A. Valette. *Elementary Number Theory, Group Theory, and Ramanujan Graphs*. Cambridge University Press, 2003.

[EH78] K.Y. Eng and M. Hecht. Switch matrix for TWTA redundancy on communication satellites. *National Telecommunications Conference (NTC'78)*, 3, December 1978.

[Gir06] F. Giroire. *Réseaux, algorithmique et analyse combinatoire de grands ensembles*. PhD thesis, Université Paris 6, France, November 2006.

[Hav06] F. Havet. Repartitors, selectors and superselectors. *Journal of Interconnexion Networks*, 7(3) :391–415, 2006.

[LV83] G. Lev and L.G. Valiant. Size bounds for superconcentrators. *Journal of Theoretical Computeur Science*, 22 :233–251, 1983.

[Mor94] M. Morgenstern. Existence and Explicit Constructions of $q+1$ regular Ramanujan Graphs for Every Prime Power q. *Journal of Combinatorial Theory Series B*, 62 :44–62, 1994.

[MP06] B. Monien and R. Preis. Upper bounds on the bisection width of 3-and 4-regular graphs. *Journal of Discrete Algorithms*, 4(3) :475–498, 2006.

[Pin73] M. Pinsker. On the complexity of a concentrator. *7th International Teletraffic Conference*, 318, 1973.

[Pip77] N. Pippenger. Superconcentrators. *SIAM Journal on Computing*, 6 :298–304, 1977.

[RVW02] O. Reingold, S. Vadhan, and A. Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. *Annals of Mathematics*, 155(1) :157–187, 2002.

[Sch06] U. Schöning. Smaller superconcentrators of density 28. *Information Processing Letters 98*, 4 :127–129, 2006.

[Val75] L.G. Valiant. On non-linear lower bounds in computational complexity. In *Proceedings of seventh annual ACM symposium on Theory of computing (STOC'75)*, pages 45–53, New York, NY, USA, 1975. ACM.

Ch. 4 : Algorithmes de routage

[AAG+05] S. Alouf, E. Altman, J. Galtier, J.F. Lalande, and C. Touati. An algorithm for satellite bandwidth allocation. In *Proceedings of IEEE infocom 2005*, volume 1, pages 560–571, 2005.

[AAG+06] S. Alouf, E. Altman, J. Galtier, J.F. Lalande, and C. Touati. Quasi-Optimal Resource Allocation in Multi-Spot MFTDMA Satellite Networks. *Combinatorial Optimization in Communication Networks*, 2006.

[AHSZ08] O. Amini, F. Huc, I. Sau Valls, and J. Zerovnik. (ℓ,k)-Routing on Plane Grids. *ArXiv e-prints*, 803, March 2008.

[AWW05] I.F. Akyildiz, X. Wang, and W. Wang. Wireless mesh networks : a survey. *Computer Networks*, 47(4) :445–487, 2005.

[BHHL07] J-C Bermond, F. Havet, F. Huc, and C. Linhares. Allocation de fréquence et coloration impropre des graphes hexagonaux pondérés. In *Neuvièmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel'07)*, Ile d'Oléron„ May 2007.

[CBF+03] J.B. Cain, T. Billhartz, L. Foore, E. Althouse, and J. Schlorff. A link scheduling and ad hoc networking approach using directional antennas. *IEEE Military Communications Conference, 2003 (MILCOM'03)*, 1, 2003.

[HKS05] F. Havet, R.J. Kang, and J.S. Sereni. Improper Colourings of Unit Disk Graphs. *Electronic Notes in Discrete Mathematics*, 22 :123–128, 2005.

[HLJ97] F.K. Hwang, T.S. Lin, and R.H. Jan. A Permutation Routing Algorithm for Double Loop Network. *Parallel Processing Letters*, 7(3) :259–265, 1997.

[HLR08] F. Huc, C. Linhares, and H. Rivano. The proportional colouring problem : Optimizing buffers in radio mesh networks. In *IV Latin-American Algorithms, Graphs and Optimization Symposium (LAGOS'07), 25-29 November 2007*, volume 30, pages 141–146, February 2008.

[KMP] R. Klasing, N. Morales, and S. Pérennes. On the complexity of bandwidth allocation in radio networks with steady traffic demands. *Theoretical Computer Science*. To appear.

[Lal04] J.-F. Lalande. *Conception de réseaux de télécommunications : optimisation et expérimentations*. PhD thesis, École doctorale STIC, Université de Nice-Sophia Antipolis, December 2004.

[LMR88] T. Leighton, B. Maggs, and S. Rao. Universal packet routing algorithms. *29th Annual Symposium on Foundations of Computer Science*, pages 256–269, 1988.

[LMR94] F. T. Leighton, B. M. Maggs, and S. B. Rao. Packet Routing and Job-Shop Scheduling in O(congestion + dilation) Steps. *Combinatorica*, 14(2) :167–186, 1994.

[LZ05] H. Liu and B. Zhao. Optimal scheduling for link assignment in traffic-sensitive STDMA wireless ad-hoc networks. *International Conference on Computer Networks and Mobile Computing, Zhangjiajie, China*, pages 218–228, 2005.

[MR00] C. McDiarmid and B. Reed. Channel assignment and weighted coloring. *Networks*, 36(2) :114–117, 2000.

[NS01] L. Narayanan and S.M. Shende. Static Frequency Assignment in Cellular Networks. *Algorithmica*, 29(3) :396–409, 2001.

[PP01] A. Pietracaprina and G. Pucci. Optimal Many-to-One Routing on the Mesh with Constant Queues. *Lecture Notes in Computer Science*, 2150 :645–649, 2001.

[SCK97] J.F. Sibeyn, B.S. Chlebus, and M. Kaufmann. Deterministic Permutation Routing on Meshes. *Journal of Algorithms*, 22(1) :111–141, 1997.

[Ser06] J.S. Sereni. *Coloration de graphes et applications*. PhD thesis, École doctorale STIC, Université de Nice-Sophia-Antipolis, July 2006.

[SK94] J.F. Sibeyn and M. Kaufman. Deterministic 1-k routing on meshes (with applications to worm-hole routing). In LNCS, editor, *11th Symposium on Theoretical Aspects of Computer Science*, volume 775, pages 237–248, 1994.

[ST97] A. Srinivasan and C-P. Teo. A constant-factor approximation algorithm for packet routing, and balancing local vs. global criteria. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing (STOC'97)*, pages 636–643, New York, NY, USA, 1997. ACM.

Ch. 5 : Réseaux d'accès et de cœur

[AA89] I. Algor and N. Alon. The star arboricity of graphs. *Discrete Mathematics*, 75(1-3) :11–22, 1989.

[AHHT07] O. Amini, F. Havet, F. Huc, and S. Thomassé. Wdm and directed star arboricity. Research Report 6179, INRIA, January 2007.

[Alw04] V. Alwayn. Fiber-optic technologies. http ://www.ciscopress.com/articles, April 2004.

[BFI+07] R. Boivie, N. Feldman, IBM, Y. Imai, WIDE / Fujitsu, W. Livens, ESCAUX, D. Ooms, and OneSparrow. Explicit Multicast (Xcast) Concepts and Options. *Network Working Group*, November 2007.

[BG05] R. Brandt and T. F. Gonzalez. Wavelength assignment in multfiber optical star networks under the multicasting communication mode. *Journal of Interconnection Networks*, 6 :383–405, 2005.

[BGC04] A. Boudani, A. Guitton, and B. Cousin. GXcast : generalized explicit multicast routing protocol. *IEEE Ninth Symposium on Computers and Communications (ISCC'04)*, 2, 2004.

[BJN+98] C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, and P.H. Vance. Branch-and-price : Column generation for solving huge integer programs. *Operations Research*, 46(3) :316–329, May-June 1998.

[Bra03] R. Brandt. Multicasting using wdm in multfiber optical star networks. Master's thesis, UCSB, September 2003.

[Cor03] C.C. Cordat. La fibre optique. http ://mptranss.free.fr/cc/fibre.html, 2003.

[Cou01] D. Coudert. *Algorithmique et optimisation de réseaux de communications optiques*. PhD thesis, Université de Nice Sophia-Antipolis (UNSA), December 2001.

[Dee91] S. Deering. *Multicast Routing in a Datagram Network*. PhD thesis, PhD thesis, Department of Computer Science, Stanford University, USA, 1991.

[Gui97] B. Guiduli. On incidence coloring and star arboricity of graphs. *Discrete Mathematics*, 163 :275–278, 1997.

[Gui05] A. Guitton. *Communications multicast, Contributions aux réseaux optiques et au passage à l'échelle*. PhD thesis, PhD thesis, Université de Rennes I, France, 2005.

[HCT01] J. He, S.H.G. Chan, and D.H.K. Tsang. Routing and wavelength assignment for WDM multicast networks. *IEEE Global Telecommunications Conference (GLOBECOM'01)*, 3, 2001.

[HCT01] J. He, S.H.G. Chan, and D.H.K. Tsang. Routing and wavelength assignment for WDM multicast networks. *IEEE Global Telecommunications Conference (GLOBECOM'01)*, 3, 2001.

[HCT02] J. He, S.H.G. Chan, and D.H.K. Tsang. Multicasting in WDM networks. *IEEE Communications Surveys and Tutorials*, 2002.

[JMT06] B. Jaumard, C. Meyer, and B. Thiongane. ILP formulations for the RWA problem for symmetrical systems. In P. Pardalos and M. Resende, editors, *Handbook for Optimization in Telecommunications*, chapter 23, pages 637–678. Kluwer, 2006.

[Muk06] B. Mukherjee. *Optical WDM Networks*. Springer, 2006.

[MZQ98] R. Malli, X. Zhang, and C. Qiao. Benefits of Multicasting in All-Optical Networks. *SPIE All-optical Networking*, 3531 :209–220, November 1998.

[PS06] A. Pinlou and É. Sopena. The acircuitic directed star arboricity of subcubic graphs is at most four. *Discrete Mathematics*, 306(24) :3281–3289, 2006.

[Rou03] G.N. Rouskas. Optical layer multicast : rationale, building blocks, and challenges. *Network, IEEE*, 17(1) :60–65, 2003.

[RS02] R. Ramaswami and K.N. Sivarajan. *Optical Networks : A Practical Perspective*. Morgan Kaufmann, 2002.

[Siu03] J. Siuzdak. ISI and polarization switching in high bit rate optical fiber transmission systems. *Optical and Quantum Electronics*, 35(12) :1113–1121, 2003.

[SKPK01] M.K. Shin, Y.J. Kim, K.S. Park, and S.H. Kim. Explicit multicast extension(Xcast+) for efficient multicast packet delivery. *Electronics and Telecommunication Research Institute (ETRI) Journal*, 23(4) :202–204, 2001.

[SM99] L.H. Sahasrabuddhe and B. Mukherjee. Light trees : optical multicasting for improved performance inwavelength routed networks. *Communications Magazine, IEEE*, 37(2) :67–73, 1999.

Ch. 6 : Fiabilité et tolérance aux pannes

[Bou05] A. Bouffard. Dimensionnement GRWA et protection par segment dans les réseaux optiques WDM. Master's thesis, Université de Montréal, Canada, 2005.

[CDP+07] D. Coudert, P. Datta, S. Perennes, H. Rivano, and M-E. Voge. Shared risk resource group : Complexity and approximability issues. *Parallel Processing Letters*, 17(2) :169–184, June 2007.

[CHPV08] D. Coudert, F. Huc, F. Peix, and M-E. Voge. Reliability of Connections in Multilayer Networks under Shared Risk Groups and Costs Constraints. In *IEEE International Conference on Communications (ICC'08)*, pages 5170–5174, May 2008.

[CPRV06] D. Coudert, S. Pérennes, H. Rivano, and M-E. Voge. Shared risk resource groups and survivability in multilayer networks. In *IEEE-LEOS ICTON / COST 293 GRAAL*, volume 3, pages 235–238, June 2006. Invited Paper.

[DR02] R. Dutta and G.N. Rouskas. Traffic grooming in WDM networks : Past and future. *IEEE Network*, 16(6) :46–56, November/December 2002.

[HDR06] S. Huang, R. Dutta, and G. N. Rouskas. Traffic grooming in path, star, and tree networks : Complexity, bounds, and algorithms. *IEEE Journal on Selected Areas in Communications*, 2006.

[HL04] J.Q. Hu and B. Leida. Traffic grooming, routing, and wavelength assignment in optical wdm mesh networks. In *Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'04)*, volume 1, pages 495–501, March 2004.

[HM02] P-H. Ho and H.T. Mouftah. A framework for service-guaranteed shared protection in wdm mesh networks. *IEEE Communications Magazine*, pages 97–103, February 2002.

[ML01] E. Modiano and P. Lin. Traffic grooming in WDM networks. *IEEE Communications Magazine*, 39(7) :124–129, July 2001.

[NR06] T. Noronha and C. Ribeiro. Routing and wavelength assignment by partition coloring. *European Journal of Operational Research*, 1713(3) :797–810, June 2006.

[PM04] M. Pioro and D. Medhi. *Routing, flow, and capacity design in communication and computer networks*. Elsevier/Morgan Kaufmann Amsterdam, 2004.

[Som06] A.K. Somani. *Survivability and Traffic Grooming in WDM Optical Networks*. Cambridge University Press, 2006.

[Vog06a] M-E. Voge. How to transform a multilayer network into a colored graph. In *IEEE-LEOS ICTON/COST 293 GRAAL*, volume 3, pages 116–119, June 2006.

[Vog06b] M-E. Voge. *Optimisation des réseaux de télécommunications : réseaux multi-couches, tolérance aux pannes et surveillance de trafic*. PhD thesis, École doctorale STIC, Université de Nice-Sophia Antipolis, November 17 2006.

[YVJ05] S. Yuan, S. Varma, and J.P. Jue. Minimum-color path problems for reliability in mesh networks. In *IEEE INFOCOM*, volume 4, pages 2658–2669, 2005.

[ZM03] K. Zhu and B. Mukherjee. A review of traffic grooming in WDM optical networks : Architectures and challenges. *Optical Networks Magazine*, 4(2) :55–64, March/April 2003.

Ch. 7 : Reroutage de connexions et Pathwidth

[AHP] O. Amini, F. Huc, and S. Pérennes. On the pathwidth of planar graphs. *SIAM Journal of Discrete Mathematics*. To appear.

[BF02] H. L. Bodlaender and F. V. Fomin. Approximation of pathwidth of outerplanar graphs. *Journal of Algorithms*, 43(2) :190–200, 2002.

[BFFS02] L. Barrière, P. Flocchini, P. Fraigniaud, and N. Santoro. Capture of an intruder by mobile agents. *Proceedings of the fourteenth annual ACM symposium on Parallel algorithms and architectures*, pages 200–209, 2002.

[BK96] H.L. Bodlaender and T. Kloks. Efficient and Constructive Algorithms for the Pathwidth and Treewidth of Graphs. *Journal of Algorithms*, 21(2) :358–402, 1996.

[CHM08a] D. Coudert, F. Huc, and D. Mazauric. Algorithme générique pour les jeux de capture dans les arbres. In *Dixième Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel'08)*, May 2008.

[CHM08b] D. Coudert, F. Huc, and D. Mazauric. A distributed algorithm for computing and updating the process number of a forest. Research Report 6560, INRIA, 06 2008.

[CHM08c] D. Coudert, F. Huc, and D. Mazauric. A distributed algorithm for computing and updating the process number of a forest (brief announcement). In *22nd International Symposium on Distributed Computing (DISC)*, Lecture Notes in Computer Science, Arcachon, France, September 2008. Springer. To appear.

[CHS07] D. Coudert, F. Huc, and J-S. Sereni. Pathwidth of outerplanar graphs. *Journal of Graph Theory*, 55(1) :27–41, May 2007.

[CPPS05] D. Coudert, S. Pérennes, Q.C. Pham, and J-S. Sereni. Rerouting requests in WDM networks. *Septièmes rencontres francopohones sur les Aspects Algorithmiques des Télécommunications (AlgoTel'05)*, May 2005.

[CS07] D. Coudert and J-S. Sereni. Characterization of graphs and digraphs with small process number. Research Report 6285, INRIA, September 2007.

[EST94] J.A. Ellis, I.H. Sudborough, and J. Turner. The vertex separation and search number of a graph. *Information and Computation*, 113(1) :50–79, 1994.

[Fom03] F. V. Fomin. Pathwidth of planar and line graphs. *Graphs and Combinatorics*, 19(1) :91–99, 2003.

[FT07] F.V. Fomin and D.M. Thilikos. On self-duality for pathwidth in polyhedral graph embeddings. *Journal of Graph Theory*, 55(1), January 2007.

[FT08] F.V. Fomin and D.M. Thilikos. An annotated bibliography on guaranteed graph searching. *Theor. Comput. Sci.*, 399(3) :236–245, 2008.

[KP86] L.M. Kirousis and C.H. Papadimitriou. Searching and pebbling. *Theoretical Computer Science*, 47(2) :205–218, 1986.

[LaP93] A.S. LaPaugh. Recontamination does not help to search a graph. *Journal of the Association for Computing Machinery (ACM)*, 40(2) :224–245, 1993.

[Lap99] D. Lapoire. *Structuration des graphes planaires.* PhD thesis, Université de Bordeaux, France, 1999.

[Nis07] N. Nisse. *Les jeux des gendarmes et du voleur dans les graphes. Théorie des mineurs, stratégies connexes et approche distribuée.* PhD thesis, Université Paris XI, Orsay, France, 2007.

[Pha04] Q.C. Pham. Étude d'un problème algorithmique intervenant dans le routage optique semi-dynamique. Master's thesis, Université de Nice Sophia Antipolis (Ltd. Essex, UKUNSA), France, 2004.

[Ree97] B. Reed. Treewidth and tangles : a new connectivity measure and some applications. In R. A. Bayley, editor, *Surveys in Combinatorics*, pages 87–162. Cambridge University Press, UK, 1997.

[RS83] N. Robertson and P. D. Seymour. Graph minors. I. Excluding a forest. *Journal of Combinatorial Theory Series B*, 35(1) :39–61, 1983.

[Sch90] P. Scheffler. A linear algorithm for the pathwidth of trees. In R. Henn R. Bodendiek, editor, *Topics in Combinatorics and Graph Theory*, pages 613–620. Physica-Verlag Heidelberg, 1990.

[Sko00] K. Skodinis. Computing Optimal Linear Layouts of Trees in Linear Time. *Algorithms-ESA 2000 : 8th Annual European Symposium*, September 2000.

An. F : Mixing Digraphs

[AGH08] O. Amini, S. Griffiths, and F. Huc. 4-cycles in mixing digraphs. In *Electronic Notes in Discrete MathematicsVolume 30, The IV Latin-American Algorithms, Graphs, and Optimization Symposium (LAGOS'07)*, volume 30, pages 63–68, Puerto Varas, Chile, February 2008.

[AS04] N. Alon and A. Shapira. Testing subgraphs in directed graphs. *Journal of Computer and System Sciences*, 69(3) :354–382, 2004.

[BES73] W.G. Brown, P. Erdos, and M. Simonovits. Extremal problems for directed graphs. *Journal Combinatorial Theory Series B*, 15 :77–93, 1973.

[BES85] W.G. Brown, P. Erdos, and M. Simonovits. Algorithmic solution of extremal digraph problems. *Transactions of the American Mathematical Society*, 292(2) :421–449, 1985.

[BS74] J.A. Bondy and M. Simonovits. Cycles of even length in graphs. *Journal of Combinatorial Theory, Series B*, 16 :97–105, 1974.

[ERS64] P. Erdős, A. Rényi, and V.T. Sós. A problem in Graph Theory. *American Mathematical Monthly*, 71 :1107–1110, 1964.

[ES66] P. Erdős and M. Simonovits. A limit theorem in graph theory. *Studia Scientiarum Mathematicarum Hungarica*, 1(51–57) :51, 1966.

[ES82] P. Erdős and M. Simonovits. Compactness results in extremal graph theory. *Combinatorica*, 2(3) :275–288, 1982.

[Fur83] Z. Furedi. Graphs without quadrilaterals. *Journal of Combinatorial Theory, Series B*, 34(2) :187–190, 1983.

[Gri07] S. Griffiths. One way cuts in oriented graphs. *Submited*, 2007.

[Hås99] J. Håstad. Clique is hard to approximate withinn 1- ε. *Acta Mathematica*, 182(1) :105–142, 1999.

[KSSS02] J. Komlós, A. Shokoufandeh, M. Simonovits, and E. Szemerédi. The regularity lemma and its applications in graph theory. *Theoretical aspects of computer science : advanced lectures*, pages 84–112, 2002.

[LUW95] F. Lazebnik, V.A. Ustimenko, and A.J. Woldar. A new series of dense graphs of high girth. *Bulletin of the American Mathematical Society*, 32 :73–79, 1995.

[LV05] T. Lam and J. Verstraete. A note on graphs without short even cycles. *The Electronic Journal of Combinatorics*, 12, April 2005.

[NV06] A. Naor and J. Verstraete. On the Turan number for the hexagon. *Advances in Mathematics*, 203(2) :476–496, July 2006.

[Sze75] E. Szemerédi. On sets of integers containing no k elements in arithmetic progression. *Acta Arithmetica*, 27 :199–245, 1975.

Index

Résumé : Cette thèse aborde différents aspects de la conception d'un réseau de télécommunications. Un tel réseau utilise des technologies hétérogènes : liens antennes-satellites, radio, fibres optiques ou bien encore réseaux embarqués dans un satellite. Les problématiques varient en fonction de la partie du réseau considérée, du type de requêtes et de l'objectif. Le cas des requêtes de type paquets est abordé dans le cadre des réseaux en forme de grille, mais le thème principal est le routage de requêtes de type connections (unicast et multicast). Les objectifs considérés sont : la conception d'un réseau embarqué dans un satellite de télécommunication, de taille minimum et tolérant des pannes de composants ; le dimensionnement des liens d'un réseau afin qu'il supporte des pannes corrélées ou qu'il offre une bonne qualité de service, ou s'il autorise des connections *multicast* ; le dimensionnement de la taille des buffers d'un réseau d'accés radio ; et l'optimisation de l'utilisation des ressources d'un réseau dynamique orienté connections. Dans tous ces cas la problématique du routage de connections est centrale.

Mon approche consiste à utiliser la complémentarité de techniques algorithmique et d'optimisation combinatoire ainsi que d'outils issus de la théorie des graphes tels la pathwidth et des notions reliées -process number, jeux de captures et treewidth-, différents types de coloration -impropre et pondérée, proportionnelle, directed star colouring-, les graphes d'expansion et des techniques de partitions telle la quasi partition.

Mots clefs : Algorithmique, Combinatoire, Télécommunication, Optimisation Combinatoire, Programmation linéaire, Approximation, Modélisation, Coloration, Patwidth, Process number, Jeux de capture, Routage, réseaux embarqués, Tolérance aux pannes, Qualité de service, Dynamique, Unicast, Multicast, Reroutage, Mixing Digraphs.

Abstract : This thesis deals with different aspects of the conception of telecommunication networks. Such networks are a patchwork of various technologies : antenas-satelite links, radio links and optical links, but also onboard networks of satellites. The problematics differ according to the part of the networks on which we focus, to the type of request and to the objective. Paquet requests are studied in the case of grid networks, but this thesis mainly deal with connection requests (either unicast or multicast). The objectives considered are : the conception of onboard, minimal and fault tolerant networks for telecommunication satellites ; the estimation of needed links capacities so that the network is resilient to corelated failures or offers a good quality of service or when it accepts multicast connections ; the estimation of the size of buffers in radio access networks and the optimisation of the usage of the ressources of a connection oriented dynamic network.

My approach consists in using the complementarity of algorithmics, combinatorial optimisation and tools from graph theory such as pathwidth and other related notions as process number, search number and treewidth, but also expandeurs, quasi partitioning, and colouring problems as proportional, weighted improper and directed star colouring.

Key words : Algorithmic, Combinatorial optimisation, Telecommunication, Optimisation Combinatoire, Linner programming, Approximation, Modeling, Coloring, Patwidth, Process number, Graph searching, Routing, Onboard networks, Fault tolerance, Quality of service, Dynamic, Unicast, Multicast, Rerouting, Mixing Digraphs.