The reconfiguration problem in multifiber WDM networks

Gurvan Huiban

The reconfiguration problem in multifiber WDM networks

A dissertation in
Computer Science

Presented to the
University of Nice Sophia Antipolis
and to the
Federal University of Minas Gerais

For the degree of
Doctor of Science
specialized in
Computer Science

by

Gurvan HUIBAN

Thesis directed by
Afonso FERREIRA/Geraldo Robson MATEUS

prepared at the
INRIA of Sophia Antipolis, MASCOTTE lab.
and at the
DCC, UFMG

Defended on July, 28th 2006

Examined by :	Ricardo H. C. TAKAHASHI	Professor
	Antonio A. F. LOUREIRO	Professor
Advisors	Afonso FERREIRA	Researcher
	Geraldo R. MATEUS	Professor
Reviewers	Philippe MAHEY	Professor
	Philippe MICHELON	Professor

UNIVERSITÉ de NICE-SOPHIA ANTIPOLIS – UFR SCIENCES

École Doctorale STIC

THÈSE

pour obtenir le titre de

Docteur en SCIENCES

de l'Université de Nice-Sophia Antipolis

Discipline : **INFORMATIQUE**

présentée et soutenue par

Gurvan HUIBAN

Le problème de la reconfiguration dans les réseaux optiques multifibres

Thèse dirigée par
Afonso FERREIRA/Geraldo Robson MATEUS
et préparée à l'INRIA Sophia Antipolis, projet Mascotte
et au DCC de l'UFMG

Soutenue le 28 juillet 2006

Jury :

Examinateurs	M.	**Ricardo H. C. TAKAHASHI**	Professeur
	M.	**Antonio A. F. LOUREIRO**	Professeur
Directeurs	M.	**Afonso FERREIRA**	Directeur de recherche
	M.	**Geraldo R. MATEUS**	Professeur
Rapporteurs	M.	**Philippe MAHEY**	Professeur
	M.	**Philippe MICHELON**	Professeur

Résumé

Un réseau de télécommunication optique est configuré de manière à répondre à une demande donnée, avec un certain objectif. Avec le temps, la demande d'exploitation du réseau change. C'est dans ce contexte que se pose le problème de la reconfiguration : être capable de changer la configuration du réseau de manière à l'adapter à des nouvelles demandes. Pour ce faire il faut généralement interrompre totalement ou partiellement le trafic. Étant données les quantités de données y circulant, il n'est guère envisageable d'arrêter le réseau. De nombreux paramètres sont à prendre en compte afin de déterminer ce que sera une bonne solution, et plusieurs métriques peuvent être utilisées pour mesurer la qualité d'une solution.

Dans une première partie, nous nous intéressons au problème de la reconfiguration comme problème d'optimisation mono-objectif. Nous proposons un modèle mathématique permettant de représenter le problème. Cependant, le résoudre exactement peut être très coûteux en temps de calcul. Nous proposons également une heuristique gloutonne et une heuristique basée sur le recuit simulé. Les solutions obtenues présentent différentes caractéristiques selon la métrique optimisée. L'algorithme glouton est rapide et trouve des solutions décentes. L'algorithme du recuit simulé obtient des solutions qui sont comparables aux solutions optimales.

Dans une seconde partie, nous nous intéressons à l'aspect multiobjectif du problème. Il consiste à considérer simultanément les différentes métriques et rechercher un ensemble de solutions représentant différents compromis intéressants. Puis nous proposons un algorithme basé sur notre modélisation mathématique. Nous adaptons également un algorithme évolutif. Donner une certaine flexibilité par rapport à une métrique permet généralement d'améliorer de manière significative les solutions obtenues par rapport aux autres métriques.

Mots-clefs : Reconfiguration, optimisation combinatoire, programmation linéaire, optimisation multiobjectif, réseaux optiques

Abstract

An optical telecommunication network is configured to transmit a given traffic in order to meet a given objective. However the demand changes with time and infrastructure development. The reconfiguration problem stands in this context. It consists in being able to alter the configuration of the network to adjust it to the new traffic. It is generally necessary to interrupt partially or totally the traffic to reconfigure a network. Considering the amount of data flowing on it, it may not be possible to regularly stop the network, even for a short amount of time. Many parameters have to be taken into account to find out a good solution, and many metrics can be used in order to measure the quality of a solution.

In a first part, we focus on the reconfiguration problem as a mono-objective optimization problem. We propose a mathematical model representing the reconfiguration problem. However solving exactly the proposed model may require a high computational time. We also propose a greedy and a simulated annealing heuristics. Depending on the metric optimized, the solutions have different characteristics. The greedy algorithm is fast and provides decent solutions whereas the simulated annealing algorithm provides solutions competing with the optimal ones.

In a second part, we focus on the multiobjective aspect of the reconfiguration problem. We consider at the same time different metrics and search for a set of solutions representing different interesting trade-offs instead of a unique solution. We propose an algorithm based on our mathematical formulation. We also adapt an evolutionary algorithm. The proposed methods succeed in finding different interesting trade-offs. Giving a little flexibility with respect to a metric generally allows to significantly improve the solutions with respect to the other metrics.

Keywords: Reconfiguration, combinatorial optimization, linear programming, multiobjective optimization, optical networks

Résumé long

Introduction

Depuis la fin du dix-huitième siècle, les innovations et avancées technologiques se succèdent dans le domaine des télécommunications. Cela permet la mise en place de moyens de communications toujours plus performants et fiables. Ce développement technologique est accompagné d'une augmentation de la demande et le marché des télécommunications croît de manière constante.

Parmi les différentes technologies existantes, la technologie optique, basée sur l'utilisation du LASER (*Light Amplification by Stimulated Emission of Radiation*), est la plus utilisée pour les réseaux à grande échelle et à très haut débit : les *réseaux dorsaux*. Avec cette technologie, les données à transmettre sont modulées sur des rayons lumineux monochromatiques cohérents émis par des LASERs et transportés par des fibres optiques.

Une fibre optique est un fin câble de plastique ou de verre capable de diriger un rayon lumineux. C'est un moyen de transmission présentant de nombreuses qualités, telles qu'un débit théorique extrêmement élevé, un faible taux d'erreur et une faible dissipation d'énergie.

L'installation d'un réseau de télécommunication à grande échelle est une opération coûteuse, notamment de part l'ampleur des opérations d'infrastructure. La finesse d'une fibre optique fait qu'il est facile d'en regrouper des dizaines dans un seul gros câble. De fait, les réseaux à grande échelle installés sont généralement multifibres.

La technologie optique offre des débit très élevés. Cependant, une des limitations de ce type de réseau vient de la partie électronique de la chaîne : les équipements capables de traiter des dizaines de gigabits de données par seconde sont très coûteux. Par ailleurs, les réseaux à grande échelle transportent généralement une agrégation de flux de données. Cela donne une certaine stabilité au trafic et les évolutions de trafic sont continues. Utiliser un système de communication par connexion et des routes préétablies permet de réduire la quantité de traitement électronique nécessaire à la transmission d'information dans ce contexte de lente variation de trafic.

Le trafic transporté par un réseau à grande échelle évolue au cours du temps. Le trafic est généralement plus élevé pendant les heures et les jours

utiles que de nuit ou les fins de semaine. Sur une courte période, le trafic évolue de manière croissante et décroissante. Par contre, la tendance sur le long terme est l'augmentation du trafic, de fait de l'apparition de nouveaux types d'applications.

La technologie WDM (*Wavelength Division Multiplexing* - Multiplexation par longueur d'onde) a radicalement changé l'utilisation de la technologie optique. Avec cette technologie, plusieurs canaux peuvent être transportés simultanément par une seule et même fibre, à partir du moment que ces derniers sont modulés sur des longueurs d'onde différentes. Cela permet d'augmenter la capacité d'un réseaux sans avoir à installer de nouvelles fibres optiques.

Avec l'apparition de la technologie WDM, le concept de couche optique, dont la fonction est de transmettre le signal optique, prend de l'importance. Cette couche permet la création d'une topologie virtuelle, ou topologie logique, remplaçant les connexions point-à-point. À l'intérieur de cette couche, aucun traitement électronique n'est réalisé ; le signal reste sous forme optique. La topologie virtuelle est composé de chemins optiques (*lightpaths*), correspondant à une connexion entre une paire de nœuds. La topologie logique correspond à la topologie qui sera effectivement utilisée pour transmettre les données.

Le problème de la reconfiguration

La définition de la topologie logique est un problème d'optimisation complexe. Il consiste à définir un ensemble de chemins optiques, définir la route suivie par ces derniers, ainsi que la longueur d'onde qu'ils utilisent, de manière à optimiser la qualité de la solution obtenue par rapport à une métrique. Ce problème est NP-difficile dans la plupart des cas.

Chaque nœud du réseau envoie des données aux autres nœuds. Le problème du routage consiste en définir quel chemin(s) doi(ven)t emprunter ces données sur la topologie virtuelle, de leur origine à leur destination. Ce problème peut être réduit à un problème de flot à commodités multiples.

Nous appelons *Problème de définition de la topologie logique et routage* la réunion de ces deux problèmes. À partir d'un ensemble de demandes, nous devons définir une topologie adaptée au trafic, et router ce dernier. Ce problème est un des problèmes clefs lié aux réseaux WDM, et a été grandement étudié.

Le trafic circulant sur les réseaux dorsaux correspond à l'agrégation de nombreuses requêtes. Ceci a deux conséquences. Premièrement, il est fort probable que chaque nœud envoie et reçoive des données de chaque autre nœud. Ceci nous amène à considérer un trafic du type tous pour tous (*all-to-all*). Autre conséquence de cette agrégation de trafics, l'évolution du trafic dans le temps est lente et continue. Nous discrétisons cette évolution et dé-

finissons la notion de *période de temps*. Pendant une période de temps, le trafic reste constant. Elle est considérée comme suffisamment longue pour avoir le temps d'implémenter une nouvelle topologie virtuelle. Nous considérons deux types de trafic. Le premier est défini de la façon suivante : un trafic de base est défini, puis des variations, faibles par rapport au trafic de base, sont calculées. Le second trafic est défini de manière identique à l'exception des variations qui sont nécessairement positives. Dans ce cas le trafic est croissant.

Conséquence de cette évolution du trafic, la topologie virtuelle et le routage définis initialement peuvent ne pas rester optimaux. Il peut être nécessaire de les modifier. Le problème de la reconfiguration consiste à déterminer de quelle façon modifier la topologie virtuelle et le routage lors des évolutions de trafic de manière à maintenir une configuration adaptée au trafic. Cependant, changer la configuration du réseau peut nécessiter une interruption de service, ce qui n'est pas souhaité compte tenu de la quantité de données circulant sur le réseau. Le problème de la reconfiguration est lié à un compromis entre les performances du réseau et le nombre de reconfigurations à appliquer à la topologie virtuelle. Notons que nous ne nous intéressons pas à la façon dont est effectivement effectuée la transition d'une topologie virtuelle à une autre.

Le problème de la reconfiguration a déjà été étudié. On trouve dans la littérature plusieurs types de travaux. Certains ne considèrent qu'une seule évolution de trafic. D'autres ne s'intéressent qu'à l'évolution de la topologie logique et le routage des données n'est pas effectué. La reconfiguration en temps réel est également abordée : le problème est résolu, mais sans qu'il n'y ait vraiment d'optimisation des ressources ou du nombre de reconfigurations. Finalement, d'autres travaux abordent le problème de la reconfiguration défini de la même façon que nous, mais en se restreignant à certains cas particuliers. Notons qu'un problème similaire est parfois trouvé sous le nom de *Groupage de trafic dynamique* (*Dynamic traffic grooming*).

De manière plus spécifique, nous nous intéressons au problème suivant : nous considérons un réseau multifibre WDM de topologie quelconque. Chaque fibre peut transporter un certain nombre de liens optiques, c'est-à-dire de canaux capables de transporter de l'information, transportés par une longueur d'onde. Nous disposons d'une succession de matrices de trafic de type *all-to-all*, une pour chaque période de temps considérée. Nous supposons que les évolutions futures du trafic sont connues. Pour chaque période de temps, nous calculons une topologie virtuelle et un routage des données à sauts multiples (*multihop*). Parmi les solutions possibles, nous choisissons la meilleure par rapport à une métrique. Deux approches différentes sont considérées. La première traite le problème comme un problème d'optimisation mono-objectif : une métrique est choisie initialement et le problème est résolu. La seconde approche revient à considérer simultanément l'ensemble des métriques possibles et à rechercher un ensemble de solutions présentant

les différents compromis intéressants.

Plusieurs métriques peuvent être considérées pour le problème de la reconfiguration. On peut chercher à minimiser le nombre de liens optiques utilisés, ce qui correspond à minimiser la charge du réseau. On peut chercher à minimiser le nombre de chemins optiques, ce qui a une influence directe sur le coût des routeurs installés. Diminuer la charge maximum des liens permet de répartir le trafic sur l'ensemble du réseau, donnant plus de flexibilité à l'administration du réseau. Diminuer le nombre moyen de sauts permet de réduire le délai de transmission des données sur le réseau. Finalement, on peut avoir pour but de minimiser le nombre de reconfigurations à appliquer.

Le problème mono-objectif de la reconfiguration

Dans un premier temps, nous considérons le problème comme un problème d'optimisation mono-objectif. Nous proposons un modèle mathématique en programmation linéaire mixte et entière. La plupart des formulations utilisées pour des problèmes similaires au problème de la reconfiguration sont des formulations flots *origine-destination*. Avec une telle formulation, une commodité est définie pour le trafic associé à chaque paire "origine-destination". En conséquence, le nombre de variables et de contraintes définies est très élevé. Afin d'obtenir un modèle plus concis, nous proposons une formulation *source*. Dans ce cas, une commodité est définie pour le trafic associé à chaque origine. Les deux formulations sont équivalentes lorsque le coût de chaque arête ne dépend pas de la commodité, ce qui est notre cas.

Notre formulation mathématique inclut des contraintes permettant la définition de la topologie virtuelle, un ensemble de contraintes servant à effectuer le routage des données et un ensemble de contraintes définissant la reconfiguration de la topologie virtuelle. Nous exprimons également les cinq métriques mentionnées précédemment. Certaines variables de notre modèle sont entières, ce qui rend généralement la recherche de solution plus difficile. Parmi ces variables entières, certaines peuvent être relâchée : elles prendront des valeurs entières dans la solution optimale.

Nous proposons également plusieurs extensions à notre modèle. Un type d'extensions est dit *technologique*. Il consiste à rendre le modèle mathématique plus réaliste. Nous proposons trois extensions à ce modèle. On peut souhaiter limiter le nombre d'émetteurs et de récepteurs LASERS à installer en chaque nœud, étant donné que ce sont des équipements coûteux. Nous proposons la formulation mathématique de la contrainte à ajouter à notre modèle remplissant ce rôle. Nous proposons également une extension afin de ne pas compter la libération d'un lien optique comme reconfiguration. Finalement, nous proposons une extension imposant un taux de remplissage minimum dans les chemins optiques.

Nous proposons également des coupes, afin de réduire l'espace à explorer

lors de la recherche de la solution optimale. Une première coupe consiste à mettre en relation des variables de flots et le nombre de chemins optiques définis. Une seconde coupe revient à obtenir une limite inférieure au nombre de chemins optiques sortant d'un nœud en considérant la quantité totale de trafic dont le nœud en question est à l'origine. Une coupe similaire est définie, donnant une limite inférieure au nombre de chemins optiques arrivant en un nœud. Finalement, nous proposons une coupe permettant de supprimer des solutions équivalentes à une solution trouvée, obtenues simplement par permutation des longueurs d'onde utilisées.

Nous définissons une borne inférieure pour notre problème. Cette borne est obtenue en résolvant le problème de la reconfiguration en modifiant le réseau de départ. La modification considérée revient à remplacer un lien contenant plusieurs fibres, chacune capable de transporter plusieurs longueurs d'onde, par un lien contenant de nombreuses fibres, chacune capable de ne transporter qu'une seule longueur d'onde. Cela revient à supprimer la contrainte empêchant deux chemins optiques de même longueur d'onde d'emprunter la même fibre.

Malgré la concision de notre modèle, le problème de la reconfiguration est un problème complexe, et il est difficile, sinon impossible, de le résoudre avec de grandes instances. Pour palier à ce problème, nous proposons deux heuristiques.

La première est une heuristique gloutonne, c'est-à-dire un algorithme constructif. À une étape donnée nous choisissons la solution qui nous semble la plus intéressante, sans considérer les possibles conséquences de notre choix. Ce type d'algorithme est très simple et généralement très rapide. L'algorithme que nous décrivons a été pensé de façon à favoriser les routages de données monosauts.

La seconde heuristique proposée est basée sur le *Recuit simulé* (*Simulated annealing*). Nous associons au problème une température qui décroît au fur et à mesure de l'exécution de l'algorithme. Nous effectuons une recherche locale. Lorsqu'une solution meilleure que les solutions obtenues jusqu'à présent est trouvée, elle est conservée. Lorsqu'une solution est moins bonne que la meilleure solution obtenue jusqu'à présent, elle est conservée avec une certaine probabilité. Elle dépend de la valeur de la température, de façon à ce qu'au début de l'algorithme cette probabilité soit relativement élevée, et qu'en fin d'algorithme elle soit faible. Ainsi, en début d'algorithme des solutions très variées sont explorées, alors qu'en fin d'algorithme l'exploration des solutions est beaucoup plus sélectif.

Problème mono-objectif : résultats expérimentaux

Nous effectuons de nombreux tests afin d'étudier les performances de la résolution du modèle mathématique, des coupes et de la borne inférieure

proposés. Nous cherchons à déterminer quelle est l'influence des extensions proposées et si les heuristiques décrites sont efficaces. Pour ce faire, nous utilisons huit topologies différentes, la plus petite ayant 7 nœuds et 20 liens, la plus grande ayant 50 nœuds et 250 liens. Les trafics considérés sont générés aléatoirement. Nous résolvons nos problèmes sur un PC de bureau équipé d'un gigaoctet de RAM. Pour résoudre les instances décrites par le modèle mathématique, nous utilisons le solveur Cplex[1] version 9, avec une limite de temps.

Nous comparons d'abord la formulation source que nous proposons avec une formulation plus classique "origine-destination". Les problèmes générés avec la formulation source sont plus compacts qu'avec la formulation classique : le nombre de variables et de contraintes est nettement plus faible. Le temps de calcul nécessaire à la résolution du problème est plus élevé avec la formulation classique. En temps limité, la solution obtenues avec la formulation source dans la grande majorité des cas meilleure. Avec la formulation source, nous sommes capables de résoudre des problèmes de taille supérieure qu'avec la formulation classique.

Nous nous intéressons ensuite à l'influence des métriques et à l'efficacité des coupes présentées. Le temps de calcul nécessaire à la résolution du problème dépend grandement de la métrique choisie. Si l'on cherche à minimiser le nombre de liens optiques ou le nombre de chemins optiques, le temps de calcul est très élevé. Par contre, avec les métriques sur la charge des liens ou le nombre de sauts, le temps de calcul est nettement plus faible. Minimiser le nombre de reconfiguration est très rapidement résolu avec des petites instances, mais dès que la taille des instances augmente, le temps de calcul devient très élevé.

La coupe mettant en relation le nombre de chemins optiques et les variables de flux est plutôt peu intéressante : en temps limité, les résultats obtenus avec la coupe sont pire que sans. Les coupes donnant une limite inférieure au nombre de chemins optiques entrant et sortant d'un nœud sont intéressantes : elle permette de diminuer le temps de calcul et d'améliorer la qualité de la solution en temps limité. Par ailleurs, le *gap*, c'est-à-dire l'incertitude sur la solution obtenue, est généralement plus faible que sans la coupe. Finalement, la coupe qui permet d'éviter les solutions identiques à une permutation près est d'un intérêt limité. Une des formulations de cette coupe obtient de mauvais résultats, et l'autre obtient des résultats de qualité similaire à ceux obtenus sans coupe.

La borne inférieure proposée obtient de bons résultats lorsque le processus de résolution du problème modifié arrive à son terme. Dans ces cas-là, la solution obtenue est de qualité supérieure à celle obtenue par relaxation linéaire. Dans certains cas cependant, nous n'arrivons pas à obtenir la solution optimale de la borne inférieure, faisant que la valeur obtenue n'est

[1]Copyright ©Ilog 1997-2005. Cplex is a registered trademark of Ilog.

d'aucun secours. Il s'avère que la borne inférieure proposée converge difficilement quand la métrique utilisée est le nombre de liens optiques ou le nombre de chemins optiques.

Selon la métrique choisie comme fonction objectif, les caractéristiques de la solution obtenue sont très différentes. Lorsque l'on minimise le nombre de liens optiques, on obtient des solutions dont la charge n'est pas répartie. Les chemins optiques sont courts et l'agrégation de trafic élevée. Le nombre de reconfigurations à effectuer est relativement faible. Lorsque la métrique choisie est le nombre de chemins optiques, la solution contient un ensemble de longs chemins optiques point à point complètement remplis, et un ensemble de très courts chemins optiques contenant ce qu'il reste du trafic. Si l'on minimise la charge maximum, on trouve les solutions ayant la charge la mieux répartie sur le réseau. Cependant, de telles solutions utilisent beaucoup de ressources et le nombre de reconfigurations est élevé. Minimiser le nombre moyen de sauts tend à créer de longs et directs chemins optiques, ce qui se répercute sur le nombre de reconfigurations qui est élevé. Finalement, si l'on chercher à minimiser le nombre de reconfigurations à effectuer, on obtient des solutions utilisant un très grand nombre de ressources, de manière à toujours avoir des chemins optiques disponibles. Du coup, le nombre de reconfigurations à effectuer est nul.

Nous nous intéressons également à l'influence de la topologie et du trafic sur le solveur chargé de chercher la solution du problème. Le type de trafic ou la topologie du réseau ne semble pas changer la difficulté de résolution du problème. Seule la densité du trafic semble avoir une influence lorsque nous minimisons le nombre de reconfigurations.

Nous nous intéressons ensuite aux différentes extensions technologiques décrites précédemment. Limiter le nombre de récepteurs et de transmetteurs rend le problème plus difficile à résoudre, particulièrement lorsque ce nombre est serré, c'est-à-dire lorsqu'il est proche du minimum tel que le problème admette toujours une solution. Cependant, l'influence d'une telle restriction est faible.

Ne pas considérer pas comme une reconfiguration le fait de libérer des ressources n'a d'influence que lorsque l'on minimise le nombre de reconfigurations. Dans ce cas-là, on observe qu'un nombre élevé de ressources est alloué au début lors de la première période de temps, et les ressources sont progressivement libérées lorsqu'elle ne sont plus utiles.

Lorsque l'on chercher à imposer un taux de remplissage minimum pour les chemins optiques, le solveur augmente "artificiellement" la longueur du routage, de façon à augmenter le taux d'occupation des chemins optiques.

Nous comparons les solutions obtenues par les heuristiques à celles obtenues par le modèle. Le recuit simulé est capable de fournir des solutions de qualité très comparable à celles obtenues en résolvant le modèle mathématique, pour un temps de calcul nettement plus faible. Par ailleurs, le recuit simulé est capable de résoudre des instances de nettement plus grande taille

que le modèle mathématique. L'algorithme glouton est très rapide, mais trouve des solutions de qualité inférieure au recuit simulé ou au modèle mathématique. En fait, les solutions trouvées par l'algorithme glouton correspondent à un compromis entre les ressources utilisées et le nombre de reconfigurations à effectuer. Lors de la définition de la topologie logique, l'algorithme glouton trouve une bonne solution. Cependant, lors du passage à la seconde période de temps, la qualité de la topologie virtuelle trouvée par l'algorithme glouton est en retrait par rapport à celle trouvée par le solveur.

Le problème multiobjectif de la reconfiguration

Le problème de la reconfiguration admet plusieurs métriques différentes. Selon la métrique choisie comme fonction objectif, le résultat obtenu présente des caractéristiques différentes. Cela peut être problématique, et on peut être tenté d'imposer des contraintes supplémentaires au modèle afin d'éviter des solutions finalement peu intéressantes. Cependant, ajouter de telles restrictions supplémentaires nécessite une grande connaissance du modèle mathématique, et peut malgré tout mener à des résultats indésirables.

Par ailleurs, le choix d'une métrique étant généralement fait *a priori* avant le processus de résolution, le décideur ne dispose d'aucune idée de la performance de la solution obtenue par rapport aux autres métriques possibles. Cette approche manque de flexibilité. L'*optimisation multiobjectif*, également appelé *optimisation vectorielle*, permet d'éviter ce problème. On ne calcule pas une unique solution, mais un ensemble de "bonnes" solutions. Une telle approche permet d'obtenir des informations importantes, comme la relation qu'ont les différentes métriques entre elles.

Un problème d'optimisation multiobjectif est un problème d'optimisation dont la fonction objectif est vectorielle, par opposition aux problèmes d'optimisation mono-objectifs dont la fonction objectif est scalaire. Par conséquent de ce fait, il n'existe *a priori* pas de "meilleure solution", mais un ensemble de meilleures solutions. Cet ensemble porte de nom d'*ensemble de Pareto* On dit qu'un point est *dominé* s'il existe un autre point obtenant de meilleures performances avec toutes les métriques considérées simultanément. L'ensemble de Pareto est constitué des points qui ne sont pas dominés. Cela signifie que si l'on choisit deux point de l'ensemble de Pareto, le premier obtiendra de meilleures performances que l'autre par rapport à une métrique, et de moins bonnes performances par rapport à une autre métrique. Autrement dit, on ne peut améliorer les performances d'une solution appartenant à l'ensemble de Pareto par rapport à une métrique qu'en diminuant ses performances par rapport à une autre métrique. On appelle *point idéal* le point dont les coordonnées correspondent à la valeur optimal de chaque métrique. Un tel point n'admet généralement pas de solutions : cela signifierait qu'il existe une solution obtenant les meilleures performances possible simultanément

avec chaque métrique.

Le problème de la reconfiguration est un problème combinatoire. Certaines variables prennent des valeurs discrètes. Par conséquent, l'ensemble de Pareto n'est pas forcément continu, ce qui peut être problématique pour certains algorithmes multiobjectifs.

Résoudre un problème multiobjectif revient à recherche l'ensemble de Pareto. Une fois cet ensemble identifié, le décideur peut choisir la solution qui lui semble la meilleure. Généralement, il n'est pas possible d'identifier analytiquement l'ensemble de Pareto. On cherche alors à en obtenir une approximation, en calculant des points qui lui appartiennent.

Pour ce faire, on peut résoudre des problèmes d'optimisation monoobjectifs, modifiés de façon à ce que leur solution optimale appartienne à l'ensemble de Pareto. Dans cette optique, il existe plusieurs méthodes. On peut donner des poids aux différentes métriques et résoudre le problème d'optimisation dont la fonction objectif est la somme pondérée des métriques. En modifiant la valeur des poids, on obtient des solutions différentes appartenant à l'ensemble de Pareto. Cette méthode est très simple. Dans le cas d'un ensemble de Pareto non-convexe, certains points ne seront jamais obtenus.

Une autre possibilité consiste à définir à partir du point idéal une direction, et à minimiser la distance entre l'espace solution et ce point idéal. Selon la direction choisie, un point ou un autre de l'ensemble de Pareto sera obtenu. Cette méthode est appelée *méthode des relaxations*. Cependant, elle ne nous est pas apparue très adaptée aux problèmes combinatoires. En effet, elle peut retourner des points qui sont dominés. De plus, afin de réduire le nombre de variables entières de notre modèle, nous relâchons la contrainte d'intégralité de certaines variables. Cela ne pose pas problème lorsque l'on a comme fonction objectif l'optimisation d'un critère de performance. Mais la méthode des relaxations utilise une fonction objectif radicalement différente (la distance d'un point à l'ensemble des solutions) et l'intégralité des variables relâchées n'est plus garantie.

Une troisième possible méthode revient à ajouter des contraintes supplémentaires afin "d'empêcher" le solveur de trouver la solution optimale par rapport à une métrique. Cette méthode porte le nom de *méthode $\epsilon-$restreinte*. Le problème de cette méthode est qu'elle peut générer des problèmes sans solution lorsque l'on considère trois ou plus de trois objectifs simultanément.

Afin de résoudre le problème multiobjectif de la reconfiguration, nous adaptons la méthode ϵ-restreinte. Nous définissons la notion de *successeur* et de *prédécesseur*, ce qui nous permet d'ordonner un ensemble de $\epsilon-$vecteurs. Nous décrivons un algorithme permettant de générer un ensemble de $\epsilon-$vecteurs de façon à garantir que cela ne rende pas le problème sans solution. Nous ordonnons cet ensemble de $\epsilon-$vecteurs avec le concept de successeur. Puis nous résolvons un par un les ϵ-problèmes, dans l'ordre. Cela nous permet de fournir une solution initiale au problème. Notre algorithme améliore de deux façons les méthodes $\epsilon-$restreintes défauts, puisqu'il garantit la fai-

sabilité des problèmes générés et il fournit une solution initiale à chaque problème. Cependant, il reste le problème du choix des $\epsilon-$vecteurs. Un autre problème vient de l'incertitude quant au nombre de $\epsilon-$vecteurs nécessaires à l'obtention d'une bonne approximation de l'ensemble de Pareto. Dernier point d'incertitude : il n'est pas prouvé que notre méthode soit capable de décrire dans son intégralité l'ensemble de Pareto.

Les méthodes décrites ci-dessus ont toutes le même fonctionnement : on modifie le problème à résoudre d'une certaine façon, puis on résout le problème modifié en utilisant des méthodes d'optimisation mono-objectif. Une telle approche peut être coûteuse en terme de temps de calcul. On peut vouloir définir un processus qui en une seule exécution renvoie de nombreux points appartenant à l'ensemble de Pareto. C'est possible en utilisant des algorithmes évolutifs, tels que les algorithmes génétiques.

Pour cette approche du problème, nous résolvons un sous-problème au problème de la reconfiguration tel que nous l'avons décrit jusqu'à présent : nous résolvons un problème de définition multipériode de topologie virtuelle.

Les algorithmes évolutifs fonctionnent de la façon suivante : on définit d'abord une *population*. Chaque *individu* de la population représente une possible solution au problème. Cette population évolue grâce à des opérations génétiques. Une *mutation* est une altération aléatoire de l'individu. À partir de deux individus, un *croisement* génère deux nouveaux individus. La qualité d'un individu est évaluée avec une fonction *fitness*. À chaque itération de l'algorithme, de nouveaux individus sont générés. Certains de ces individus sont sélectionnés et servent de population de base pour l'itération suivante. Avec une fonction fitness adéquat, il est possible de faire évoluer l'ensemble de la population vers l'ensemble de Pareto. Le principal problème issu de l'utilisation d'algorithmes génétiques est l'absence de garantie quant à la qualité des solutions obtenues.

Il existe plusieurs variantes d'algorithme évolutifs résolvant des problèmes multiobjectifs. Nous avons choisi d'utiliser l'algorithme SPEA (*Strength Pareto Evolutionary Algorithm*), qui conserve une population externe de solutions non-dominées. Les opérations génétiques sont appliquées tant sur les individus de la population principale que sur les individus de la population externe.

Nous décrivons la façon dont nous encodons une solution en un chromosome, ainsi que la façon dont nous réalisons les différentes opérations génétiques. L'allocation des longueurs d'onde n'est pas intégrée au chromosome, mais est calculée de manière séparée en utilisant un algorithme glouton. Par ailleurs, afin d'éviter que la population externe ne contienne trop d'éléments, nous utilisons une procédure qui permet de supprimer des solutions lorsque celles-ci sont très similaires. La population initiale est générée de manière aléatoire.

Problème multiobjectif : résultats expérimentaux

Dans un premier temps, nous effectuons une étude multiobjectif superficielle du problème de la reconfiguration à l'aide de la méthode des objectifs pondérés. Nous utilisons le solveur et notre implémentation de l'algorithme du recuit simulé afin de résoudre les différents problèmes générés. Utiliser comme fonction objectif une somme pondérée de métriques semblent rendre le problème plus difficile à résoudre. Cependant, elle nous permet d'obtenir des compromis intéressants.

Nous effectuons ensuite une étude multiobjectif plus poussée utilisant notre algorithme basé sur la méthode ϵ-restreinte. Pour deux instances, nous générons de nombreux problèmes que nous résolvons. Pour chaque instance, nous obtenons de nombreux points de l'ensemble de Pareto, mais également des solutions dominées et des solutions en double. Ces solutions identiques apparaissent lorsque les $\epsilon-$vecteurs sont trop proches les unes des autres, ou lorsque le solveur n'arrive pas à trouver une nouvelle solution dans le temps imparti. Nous obtenons cependant de nombreuses solutions différentes présentant des compromis différents.

Nous effectuons également de nombreux tests avec l'algorithme évolutif SPEA. Le nombre de générations nécessaires à la convergence de l'algorithme dépend de la taille de l'instance. Alors que 1000 générations sont suffisantes pour les plus petites instances, 1500 itérations sont nécessaires pour les plus grandes. Au cours de l'exécution de l'algorithme on constate que la population avance progressivement et dans son ensemble vers l'ensemble de Pareto. Il apparaît que le compromis entre le nombre de ressources utilisées et le nombre de reconfigurations n'est pas très marqué, au contraire de celui entre la charge des liens et le nombre de reconfigurations. La relation entre le nombre de ressources utilisées et la charge des liens est très forte.

Conclusion

Dans cette thèse, nous nous intéressons au problème de la reconfiguration dans les réseaux optiques WDM. C'est un problème complexe qui inclut de nombreux paramètres. Nous proposons un modèle mathématique utilisant une formulation source pour le problème, des extensions technologiques, des coupes et une borne inférieure. Nous proposons également une heuristique gloutonne et une heuristique basée sur le recuit simulé. Nous effectuons de nombreux tests afin d'identifier l'influence et l'efficacité des différentes propositions. Cependant, une approche mono-objectif est insuffisante pour prendre en compte les différents compromis impliqués par les différentes métriques. Nous décrivons alors une approche multiobjectif. Nous proposons un algorithme basé sur la méthode ϵ-vecteurs, et nous adaptons l'algorithme SPEA. Nous faisons de nombreux tests illustrant l'efficacité de nos méthodes

et donnant de nombreux compromis possibles.

Dans cette thèse Les travaux partent du principe que l'on connaît à l'avance les futurs trafics. C'est une hypothèse forte qu'il serait intéressant de relâcher au moins partiellement, c'est-à-dire de concilier prévision et adaptation. Nous ne nous sommes également pas intéressé à la façon dont les transitions d'une topologie virtuelle à une autre sont effectuées. Enfin, il serait particulièrement intéressant de fournir un algorithme d'aide à la décision afin d'aider le décideur à choisir parmi les différentes solutions de l'ensemble de Pareto celle qui l'intéresse le plus.

Au légendaire Kaldi.

Remerciements

Je tiens en tout premier lieu à remercier sincèrement mes deux directeurs de thèse qui m'ont permis de mener à bien ce projet. Un grand merci donc à Afonso Ferreira, grâce à qui j'ai pu travailler avec les gens de l'UFMG et entreprendre cette aventure. Un grand merci aussi à Geraldo Robson Mateus, qui a su m'orienter au cours de cette thèse malgré mon portugais parfois approximatif, qui a toujours trouvé le temps de me prodiguer ses conseils ou de relire mes manuscrits.

Je remercie également Philippe Mahey et Philippe Michelon, qui ont rapporté mon manuscrit et ce malgré le court délais dont nous disposions et participé à mon jury. Cette gratitude va aussi aux autres membres du jury, à savoir Ricardo H.C. Takahashi et Antônio A.F. Loureiro qui ont contribué par leurs critiques et commentaires me permettent d'ouvrir des perspectives intéressantes sur mon travail.

Au cours de cette thèse j'ai eu l'occasion de travailler et d'évoluer avec les gens du LaPO, de l'UFMG, dirigé par Geraldo R. Mateus qui ne ménage pas ses efforts pour insuffler une dynamique au groupe et qui nous donne les moyens de mener à bien nos recherches. Je lui témoigne ma gratitude, et à travers lui, à tous les gens du LaPO. J'ai également travaillé avec les gens du projet Mascotte, de l'INRIA Sophia Antipolis, projet commun avec l'I3S et dirigé par Jean-Claude Bermond. Ce dernier a su rendre cette équipe de recherche à la fois riche et agréable. Je le remercie et, par son intermédiaire, tous les membres de l'équipe.

Au cours des quatre ans qu'ont duré mon doctorat j'ai eu le privilège de suivre de nombreux cours particulièrement intéressants. J'adresse mes salutations à N. Ziviani, Antônio A.F. Loureiro, Newton J. Vieira, Claudionor J.N. Coelho Jr, Ricardo H.C Takahashi, João A. de Vasconcelos, Rodney R. Saldanha, ainsi qu'à StamFord. Suivant ces cours, j'ai énormément appris et avec plaisir, parfois au prix de grands moments de désespoir ou de stress.

Ma situation administrative n'a pas toujours été des plus communes, ce qui n'a pas simplifié la tâches des assistantes tant de l'INRIA que de l'UNSA ou de l'UFMG. Je leur témoigne ma gratitude, et tout particulièrement à Éphie Deriche et Renata V.M. Rocha. Je remercie également mes amis, Jean-François Lalande et Marie-Émilie Voge, qui ont accepté de servir d'intermédiaires lorsque je me trouvais loin de Nice.

Naturellement, je salue tous ceux avec qui, autour d'un café ou d'un email, j'ai eu l'occasion d'avoir des conversations enrichissantes. Parmi ceux-ci, j'aimerais citer André, Bertrand, David, Fabíola, Fabrice, Filipe, Gustavo, Jean-François, Jean-Marc, João, Loic, Marie-Émilie, Martin, Michel, Nicolas, Paulo, Pável, Pedro(s), Renan, Yann, etc. Je salue également les inimitables Gros-Nazes.

Pour conclure, je remercie sincèrement ma famille, sans qui je ne serais rien ; famille qui s'est d'ailleurs agrandie pendant ce doctorat.

Gurvan Huiban

O problema da reconfiguração nas redes WDM multifibras

Orientadores: Geraldo Robson Mateus - UFMG
Afonso Ferreira - UNSA

Tese em cotutela apresentada ao programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais (UFMG) como requisito parcial para obtenção do grau de Doutor em Ciência da Computação pela UFMG e pela Universidade de Nice Sophia Antipolis (UNSA).

Belo Horizonte - MG
Julho de 2006

Resumo

Uma rede de telecomunicação ótica está configurada de forma a responder a uma dada demanda com um certo objetivo. No entanto, com o passar do tempo e com o desenvolvimento da infraestrutura, a demanda da rede muda. É neste contexto que se encontra o problema da reconfiguração, que tenta propor mudanças na atual configuração de rede de forma a adaptá-la à novas demandas. Considerando a quantidade de dados transportada, não é viável interromper a utilização da rede para reconfigurá-la, mesmo que por alguns instantes. Vários parâmetros devem ser considerados para determinar o que seria uma boa solução, e varias métricas podem ser usadas para medir a qualidade de uma solução.

Em um primeiro momento, estudamos o problema da reconfiguração como sendo um problema de otimização mono-objetivo. Nós propomos um modelo matemático que atende as condições do problema. No entanto, obter a solução ótima para o problema via modelo pode ser muito custoso em termos de tempo de computação. Propomos então uma heurística gulosa e uma heurística usando o *simulated annealing*. As soluções obtidas apresentam características diferentes dependente da métrica otimizada. O algoritmo guloso é rápido e acha soluções aceitáveis. O algoritmo do *simulated annealing* obtém soluções comparaveis às soluções ótimas.

Em um segundo momento estudamos os aspectos multiobjetivo do problema da reconfiguração. Eles consistem em considerar simultaneamente as diferentes métricas e procurar, não uma única solução, mas sim, um conjunto de soluções representando diferentes compromissos interessantes e em geral em conflito. Depois propomos um algoritmo que usa nosso modelo matemático e uma adaptação do algoritmo evolutivo. Os métodos propostos acham diferentes compromissos interessantes. Ao flexibilizar uma métrica permite geralmente melhorar de forma significativa as soluções obtidas com outras métricas.

Palavras chaves: Reconfiguração, otimização combinatória, programação linear, otimização multiobjetivo, redes óticas.

Resumo estendido

Introdução

Desde o fim do século dezoito, inovações e avanços tecnológicos acontecem no domínio das telecomunicações. Isso permite a instalação de meios de comunicação sempre mais poderosos e confiáveis. Este desenvolvimento tecnológico é acompanhado por uma demanda sempre crescente e o mercado das telecomunicações cresce constantemente.

Das diferentes tecnologias atualmente usadas, a tecnologia ótica, baseada no uso do LASER (*Light Amplification by Stimulated Emission of Radiation*), é a mais usada nas redes *backbones*, que são redes de larga escala e de muita alta velocidade. Com esta tecnologia, os dados a serem transmitidos são modulados em raios de luz coerentes monocromáticos emitidos por LASERs e transportados por fibras óticas. Estes sinais usam um comprimento de onda especifico.

A fibra ótica é um meio de transmissão que apresenta muitas qualidades, como uma taxa de transmissão teórica extremamente alta, uma taxa de erros baixa e uma baixa dissipação de energia.Uma fibra ótica é um cabo fino de plástico ou de vidro capaz de dirigir um raio de luz.

A instalação de uma rede de telecomunicação de larga escala é uma operação cara, particularmente pela complexidade das operações de infraestrutura. A espessura de uma fibra ótica facilita o agrupamento de dezenas delas num único cabo de grandes dimensões. De fato, as redes de larga escala instaladas geralmente são multifibras.

A tecnologia ótica oferece taxas de transmissão muito elevadas. No entanto, uma das limitações deste tipo de rede vem da parte eletrônica da cadeia. Os equipamentos capazes de tratar com dezenas de gigabits de dados por segundo são muito caros. Além disso, as redes de larga escala carregam geralmente uma agregação de fluxos de dados. Isso dá uma certa estabilidade no tráfego fazendo com que as evoluções de tráfego sejam contínuas. Usar um sistema de comunicação por conexões e rotas pré-definidas permite reduzir a quantidade de tratamento eletrônico que a transmissão de informação precisa neste contexto de variação lenta de tráfego.

O tráfego carregado pelas redes de larga escala evolui ao longo do tempo. O tráfego é geralmente mais alto durante os horários e os dias úteis que de

noite ou fins de semana. Num período de tempo curto, o tráfego evolui de forma crescente e decrescente. No entanto, a tendência à longo prazo é o aumento do tráfego, com a chegada de novos tipos de aplicações.

A tecnologia WDM (*Wavelength Division Multiplexing* - Multiplexação por Comprimento de Onda) mudou radicalmente o uso da tecnologia ótica. Com esta tecnologia, vários canais podem estar carregados simultaneamente por uma única fibra, sendo que estes canais são modulados em comprimentos de onda diferentes. Isso permite aumentar a capacidade de uma rede sem precisar instalar novas fibras óticas.

Com a aparição da tecnologia WDM, o conceito de camada ótica, cuja função é a transmissão do sinal ótico, ganha importância. Esta camada permite a criação de uma topologia virtual, ou topologia lógica, substituindo as conexões ponto-a-ponto. Nesta camada, nenhum tratamento eletrônico é realizado; o sinal fica na forma ótica. A topologia virtual é composta de caminhos óticos (*caminhos óticos*), que correspondem a conexões entre pares de nós. A topologia lógica corresponde à topologia que será efetivamente usada para transmitir dados.

O problema da reconfiguração

A definição da topologia lógica é um problema de otimização complexo. Ele consiste em definir um conjunto de caminhos óticos, definir a rota e o comprimento de onda que eles usam, de forma a otimizar a qualidade da solução obtida em relação com uma métrica. Este problema é NP-difícil na maioria dos casos.

Cada nó da rede manda dados aos outros nós. O problema do roteamento consiste em definir qual(is) caminho(s) deve(m) seguir estes dados na topologia lógica, da origem até o destino. Este problema pode ser reduzido a um problema de fluxo com múltiplos produtos.

Nós chamamos *Problema de definição da topologia lógica e roteamento* o conjunto destes dois problemas. À partir de um conjunto de demandas, nós devemos definir uma topologia adequada ao tráfego e rotear este último. Este problema é um dos problemas chaves relacionados com as redes WDM e esta sendo muito estudado.

O tráfego circulando nos backbones corresponde à agregação de várias requisições. Isso tem duas consequências: a primeira é que é muito provável que cada nó envie e receba dados de cada outro nó. Isso nos leva a considerar um tráfego do tipo todos para todos (*all-to-all*); a outra consequência desta agregação de tráfego é que a evolução do tráfego no tempo é devagar e continua. Nós discretizamos esta evolução e definimos a noção de período de tempo. Durante um período de tempo, o tráfego fica constante. Este período de tempo é considerado como grande o suficiente para implementar uma nova topologia lógica. Nós consideramos dois tipos de tráfego. O primeiro

é definido da seguinte forma: um tráfego de referência é definido e depois variações em relação ao tráfego inicial são efetuadas. O segundo tráfego é definido de forma idêntica sendo que as variações são sempre positivas. Neste ultimo caso, o tráfego é crescente.

Devido à evolução do tráfego, a topologia virtual e o roteamento definido no início podem não ser ótimos. Pode ser preciso modificá-los. O problema da reconfiguração consiste em achar de qual forma modificar a topologia virtual e o roteamento durante as evoluções de tráfego, de forma a manter uma configuração adequada ao tráfego. No entanto, mudar a configuração da rede pode precisar de uma interrupção de serviço, que não é desejada considerando a quantidade de dados circulando na rede. O problema da reconfiguração é ligado a um compromisso entre o desempenho da rede e o número de reconfigurações a serem aplicadas à topologia lógica. Notem que a nós não interessa à forma como é realizada a transição de uma topologia lógica a outra.

O problema da reconfiguração já foi estudado em vários tipos de trabalhos. Alguns trabalhos só consideram uma evolução do tráfego. Outros trabalhos consideram apenas a evolução da topologia lógica. Neste caso, o roteamento dos dados não é feito. A reconfiguração em tempo real é também estudada. O problema é resolvido sem que tenha realmente otimização dos recursos ou do número de reconfigurações. Finalmente, outros trabalhos abordam o problema da reconfiguração definido da mesma forma que nós, mas se restringindo a alguns casos específicos. Notamos que o problema chamado "Agrupamento de tráfego dinâmico" é um problema parecido com o que aqui tratamos.

De forma mais específica, nós nos interessamos pelo seguinte problema: consideramos uma rede WDM multifibra com qualquer topologia, sendo que cada fibra ótica pode carregar um certo número de enlaces óticos, ou seja de canais capazes de transportar informações mandadas em comprimento de onda. Nós consideramos uma sucessão de matrizes de tráfego do tipo *all-to-all*, uma para cada período de tempo considerado. Nós supomos que as evoluções futuras do tráfego são conhecidas. Para cada período de tempo, nós calculamos uma topologia virtual e um roteamento dos dados com múltiplos saltos (*multihop*). Dentre as diferentes soluções possíveis, nós escolhemos a melhor em relação com uma métrica. Duas abordagens diferentes são consideradas. A primeira considera o problema como um problema de otimização mono-objetivo, uma métrica é escolhida inicialmente e o problema é resolvido; uma segunda abordagem consiste em considerar simultaneamente o conjunto de métricas possíveis e procurar um conjunto de soluções representando os diferentes compromissos possíveis.

Várias métricas podem ser consideradas para o problema da reconfiguração. Por exemplo, podemos optar por minimizar o número de enlaces óticos usados, que corresponde a minimizar a carga da rede. Podemos também minimizar o numero de caminhos óticos, que tem uma influência direta no

custo dos roteadores instalados. Podemos diminuir a carga máxima dos enlaces, o que permite distribuir o tráfego na totalidade da rede. Diminuir o número médio de saltos permite de reduzir o tempo de transmissão dos dados na rede. Finalmente, podemos ter por objetivo minimizar o número de reconfigurações.

O problema mono-objetivo da reconfiguração

Num primeiro momento, consideramos o problema como um problema de otimização mono-objetivo. Nós propomos um modelo matemático em programação linear mista e inteira. A maioria das formulações usadas para problemas parecidos são formulações de fluxo "origem-destino". Com tal formulação, um produto é definido para o tráfego associado a cada par "origem-destino". Em consequência, o número de variáveis e de restrições definidas é muito elevado. Com objetivo de obter um modelo menor, nós propomos uma formulação "fonte". Neste caso, um produto é definido para o tráfego associado a cada origem. As duas formulações são equivalentes quando o custo de cada aresta não depende do produto, que é nosso caso.

Nossa formulação matemática inclui restrições que definem a topologia lógica, um conjunto de restrições que efetua o roteamento dos dados, e um conjunto de restrições que define a reconfiguração da topologia lógica. Nós expressamos também as cinco métricas citadas acima. Algumas variáveis do nosso modelo são inteiras, o que torna geralmente o problema mais difícil. Dessas variáveis inteiras, algumas podem ser relaxadas; de qualquer forma elas terão valores inteiros.

Nós propomos também várias extensões para nosso modelo. Um tipo de extensãos é dito *tecnológico*. Ele consiste em tornar o modelo matemático mais realista. Nós propomos três extensões deste tipo. Nós podemos desejar limitar o número de emissores ou receptores a LASERS a serem instalados em cada nó, considerando que são equipamentos caros. Nós propomos uma restrição com este papel que pode ser adicionada a nosso modelo matemático. Nós propomos também uma extensão para não considerar como reconfiguração o fato de deixar de usar um enlace ótico. Finalmente, nós propomos uma extensão impondo uma taxa mínima nos caminhos óticos.

Nós propomos também cortes para reduzir o espaço de soluções a ser explorado durante a busca da solução ótima. Um primeiro corte consiste em relacionar algumas variáveis de fluxo com o número de caminhos óticos definidos; um segundo corte consiste em obter um limite inferior ao número de caminhos óticos saindo de um nó considerando a quantidade total de tráfego que o nó considerado manda. Um outro corte parecido mas agora dando um limite inferior ao número de caminhos óticos chegando em um nó. Finalmente um corte permitindo remover soluções equivalentes a uma solução achada, obtido simplesmente por permutação dos comprimentos de

onda usados.

Nós definimos um limite inferior para nosso problema. Este limite é obtido resolvendo o problema da reconfiguração cuja rede inicial é modificada. A modificação considerada é equivalente a trocar um enlace contendo várias fibras, cada uma capaz de carregar vários comprimentos de onda, por um enlace contendo várias fibras, cada uma capaz de carregar um único comprimento de onda. Isso corresponde a remover a restrição impedindo dois caminhos óticos de mesmo comprimento de onda a usar a mesma fibra.

Apesar do nosso modelo ser relativamente conciso, o problema da reconfiguração é um problema complexo, e é difícil, se não impossível, de resolvê-lo com grandes instâncias. Para resolver este problema, nós propomos duas heurísticas.

A primeira é uma heurística gulosa, ou seja, um algoritmo construtivo. Num passo dado, nós escolhemos a solução que nos parece a mais interessante, sem considerar as possíveis consequências da nossa escolha. Este tipo de algoritmo é muito simples e geralmente rápido. O algoritmo que nós descrevemos foi pensado de forma em favorecer os roteamentos de dados mono-salto.

A segunda heurística proposta é baseada no *Simulated Annealing*. Nós associamos ao problema uma temperatura que diminue ao longo da execução do algoritmo. Fazemos uma busca local e quando encontramos uma solução que melhora as soluções achadas até o momento, ela é guardada. Quando uma solução é pior que a melhor solução achada até agora, ela é guardada com uma certa probabilidade. Esta probabilidade depende do valor da temperatura, de forma que no inicio do algoritmo esta probabilidade é bastante elevada, e que no fim de algoritmo esta probabilidade é baixa. Assim, no início de algoritmo soluções muito diferentes são exploradas, e no fim a exploração das soluções é muito mais seletiva.

Problema mono-objetivo: resultados experimentais

Muitos testes foram feitos com o objetivo de estudar o desempenho da resolução do modelo matemático, dos cortes e do limite inferior propostos. Nós procuramos definir qual é a influência das extensões propostas e se as heurísticas descritas são eficientes. Para isso, nós usamos oito topologias diferentes, a menor tendo 7 nós e 20 enlaces, a maior tendo 50 nós e 250 enlaces. Os tráfegos considerados são gerados aleatoriamente. Nós resolvemos os problemas num PC comum com um gigabyte de memória. Para resolver as instâncias descritas pelo modelo matemático, nós usamos o software Cplex[2] versão 9, com limite de tempo.

Nós comparamos primeiro a formulação fonte que nós propomos com uma formulação mais comum "fonte-destino". Os problemas gerados com

[2]Copyright ©Ilog 1997-2005. Cplex is a registered trademark of Ilog.

a formulação fonte são mais compactos que com a formulação comum. O número de variáveis e de restrições é significativamente menor. O tempo de computação necessário para resolver o problema é mais elevado com a formulação comum. Em tempo limitado, as soluções obtidas com a formulação fonte são melhores na grande maioria das vezes. Com a formulação fonte, nós podemos resolver problemas de tamanho maior que com a formulação comum.

Estudamos a influência das métricas e a eficiência dos cortes apresentados. O tempo de computação preciso para resolver o problema depende muito da métrica escolhida. Se queremos minimizar o número de enlaces óticos ou o número de caminho óticos, o tempo de computação é muito elevado. Ao contrário, com as métricas relacionadas com a carga dos enlaces ou o número de saltos, o tempo de computação é significativamente menor. Minimizar o número de reconfigurações é resolvido rapidamente com instâncias pequenas, mas assim que o tamanho das instâncias aumenta, o tempo de computação se torna muito alto.

O corte relacionado ao número de caminhos óticos e as variáveis de fluxo tem pouco interesse. Em tempo limitado, os resultados obtidos com o corte são piores que sem. Os cortes dando um limite inferior ao número de caminhos óticos entrando e saindo de um nó são interessantes. Eles permitem diminuir o tempo de computação ou melhorar a qualidade da solução obtida em tempo limitado. Além disso, o *gap*, ou seja a incerteza sobre a solução obtida, é geralmente menor que sem o corte. Finalmente, o corte permitindo evitar soluções idênticas com exceção de permutação tem um interesse menor. Uma das formulações possíveis para este corte obtém resultados ruins, e a outra obtém resultados de qualidade equivalente aos obtidos sem cortes.

O limite inferior proposto obtém bons resultados quando a resolução do problema modificado termina. Neste caso, a solução obtida é de melhor qualidade que aquela obtida com a relaxação linear. No entanto, em alguns casos, nós não conseguimos obter a solução ótima do limite inferior, tornando o valor obtido sem interesse. O limite inferior proposto converge com dificuldade quando a métrica usada é o número de enlaces óticos ou o número de caminhos óticos.

Dependendo da métrica escolhida como função objetivo, as características da solução obtida são muito diferentes. Quando minimizamos o número de enlaces óticos, obtemos soluções cuja carga não é distribuída, os caminhos óticos são curtos, a agregação do tráfego é elevada e o número de reconfigurações a serem efetuadas é relativamente baixo. Quando a métrica é o número de caminhos óticos, a solução contem um conjunto de caminhos óticos longos ponta a ponta completamente cheios, e um conjunto de caminhos óticos muito curtos contendo o que sobra do tráfego. Se minimizamos a carga máxima, achamos soluções com a melhor distribuição de carga na rede. No entanto, tais soluções usam muitos recursos e o número de reconfigurações é alto. Minimizar o número médio de saltos tende a gerar caminhos óticos

longos e diretos. Isso se repercute no número de reconfigurações, que é elevado. Finalmente, se procuramos minimizar o número de reconfigurações a serem efetuadas, obtemos soluções usando um número muito alto de recursos, de forma a sempre ter os caminhos óticos disponíveis. Em consequência, o número de reconfigurações a serem efetuadas é nulo.

Nós também testamos a influência da topologia e do tráfego no *solver*. O tipo de tráfego ou a topologia parece não mudar a dificuldade de resolução do problema. Só a densidade do tráfego parece ter uma influência quando nós minimizamos o número de reconfigurações.

Estudamos diferentes extensões tecnológicas descritas anteriormente. Limitar o número de receptores e de transmissores torna o problema mais difícil de ser resolvido, particularmente quando o número de transmissores e de receptores é próximo do mínimo tal que o problema admita uma solução. No entanto, a influência de uma tal restrição é baixa.

Não considerar como uma reconfiguração o fato de liberar recursos tem influência só quando minimizamos o número de reconfigurações. Neste caso, observamos que um número elevado de recursos é alocado no início, durante o primeiro período de tempo, e os recursos são progressivamente liberados quando eles não são mais úteis.

Quando procuramos impor uma taxa de enchimento mínimo para os caminhos óticos, o *solver* aumenta o comprimento do roteamento de forma a aumentar a taxa de ocupação dos caminhos óticos.

Nós comparamos as soluções obtidas pelas heurísticas com aquelas obtidas pelo modelo. O *simulated annealing* é capaz de obter soluções de qualidade comparável àquela que as soluções obtidas pelo modelo matemático, com um tempo de computação significativamente mais baixo. Além disso, o *simulated annealing* é capaz de resolver instâncias muita maiores que o modelo matemático. O algoritmo guloso é muito rápido, mas acha soluções de qualidade inferior ao *simulated annealing* ou ao modelo matemático. Na verdade, as soluções obtidas pelo algoritmo guloso correspondem a um compromisso entre os recursos usados e o número de reconfigurações a serem feitas. Durante a definição da topologia lógica o algoritmo guloso acha uma boa solução. No entanto, durante a transição ao segundo período de tempo, a qualidade da topologia lógica obtida pelo algoritmo guloso é de qualidade inferior em relação àquela achada pelo *solver*.

O problema multiobjetivo da reconfiguração

O problema da reconfiguração admite várias métricas diferentes. Dependendo da métrica escolhida como função objetivo, o resultado obtido apresenta características diferentes. Isso pode ser problemático, e podemos querer impor restrições adicionais ao modelo para evitar soluções pouco interessantes. No entanto, acrescentar tais restrições adicionais precisa um

grande conhecimento do modelo, e pode mesmo assim levar para resultados não desejados.

Além disso, a escolha de uma métrica sendo geralmente feita antes do processo de solução, quando o tomador de decisão não tem nenhuma idéia do desempenho da solução obtida em relação com as outras métricas possíveis. Falta flexibilidade nesta abordagem. A *Otimização multiobjetivo*, também chamada *Otimização vetorial*, permite evitar este problema. Não é uma única solução que é computada, mas um conjunto de "boas" soluções. Uma tal abordagem permite obter informações importantes, como a relação entre as métricas.

Um problema de otimização multiobjetivo é um problema de otimização cuja função objetiva é vetorial, ao contrário dos problemas de otimização mono-objetivo cuja função objetivo é escalar. Em principio não existe "melhor solução", mas um conjunto de boas soluções. Este conjunto é chamado *Conjunto de Pareto*. Um ponto é dito *dominado* se existe um outro ponto obtendo melhor desempenho com todas as métricas consideradas. O conjunto de Pareto é composto dos pontos que não são dominados. Isso significa que se escolhemos dois pontos do conjunto de Pareto, o primeiro tem um melhor desempenho que o outro em relação com pelo menos uma métrica, e um pior desempenho em relação com a outra métrica. Ou seja, é possível melhorar o desempenho de uma solução pertencente ao conjunto de Pareto em relação com uma métrica só diminuindo o seu desempenho em relação com uma outra métrica. O ponto chamado *ponto ideal* é o ponto cujas coordenadas correspondem ao valor ótimo de todas as métricas. Um tal ponto geralmente não admite soluções. Isso significaria que existe uma solução obtendo os melhores desempenhos possíveis simultaneamente com cada métrica.

O problema da reconfiguração é um problema combinatório. Algumas variáveis têm valores discretos. Conseqüentemente, o conjunto de Pareto não é necessariamente contínuo, que pode ser problemático para alguns algoritmos multiobjetivos.

Resolver um problema multiobjetivo corresponde em buscar o conjunto de Pareto. Uma vez este conjunto identificado, o tomador de decisão pode escolher a solução que lhe parece a melhor. Geralmente, não é possível identificar de forma analítica o conjunto de Pareto. Neste caso procuramos obter uma aproximação computando pontos pertecentes a este.

Para isso, podemos resolver problemas de otimização mono-objetivo, modificados de forma em que a solução ótima deles pertence ao conjunto de Pareto. Neste contexto, existe vários métodos. Podemos dar pesos às diferentes métricas e resolver o problema de otimização cuja função objetivo e o somatório com pesos associados a cada métrica. Modificando o valor dos pesos, obtemos diferentes soluções pertencentes ao conjunto de Pareto. Este método é muito simples. No entanto, no caso de um conjunto de Pareto não convexo, alguns pontos nunca serão obtidos.

Uma outra possibilidade consiste em definir a partir do ponto ideal uma direção e minimizar a distância entre o espaço solução e este ponto ideal. Dependente da direção escolhida, um ponto ou um outro do conjunto de Pareto será obtido Este método é chamado *método das relaxações*. No entanto, ele não foi muito adequado aos problemas combinatórios. Ele pode retornar pontos que são dominados. Além disso, para reduzir o número de variáveis inteiras do nosso modelo, nós relaxamos a restrição de integralidade de algumas variáveis. Isso não causa problema quando a função objetivo é um critério de desempenho. Mas o método das relaxações usa uma função objetivo completamente diferente (a distância de um ponto à um conjunto) e a integralidade das variáveis relaxadas não é mais garantida.

Um terceiro método consiste em acrescentar restrições adicionais para "impedir" o *solver* de achar a solução ótima em relação com uma métrica. Este método tem o nome de *método $\epsilon-$restrito*. O problema com este método é que ele pode gerar problemas sem soluções quando são considerados três ou mais objetivos simultaneamente.

Para resolver o problema multiobjetivo da reconfiguração, nós adaptamos o método ϵ-restrito. Nós definimos a noção de "sucessor" e de "predecessor", que nos permite ordenar um conjunto de $\epsilon-$vetores. Nós descrevemos um algoritmo permitindo gerar um conjunto de $\epsilon-$vetores de forma a garantir que isso não torna o problema sem solução. Nós ordenamos este conjunto de $\epsilon-$vetores com o conceito de sucessor precedentemente introduzido. Depois, nós resolvemos um por um os $\epsilon-$problemas, seguindo a ordem. Isso nos permite fornecer uma solução inicial para o problema. Nosso algoritmo melhora de duas formas o $\epsilon-$método padrão, porque ele garante que os problemas gerados tem solução e ele fornece uma solução inicial para cada problema. No entanto, existe o problema da escolha das restrições-ϵ. Um outro problema vem da incerteza em relação com o número de restrições-ϵ preciso para ter uma boa aproximação do conjunto de Pareto. Último ponto de incerteza, não é comprovado que nosso método seja capaz de descrever a integralidade do conjunto de Pareto.

Os métodos descritos acima tem todos o mesmo funcionamento. O problema a ser resolvido é modificado de uma certa forma e resolvido usando métodos de otimização mono-objetivo. Uma tal abordagem pode ser custosa em termos de tempo de computação. Ou seja, podemos definir um processo que retorne vários pontos pertencentes ao conjunto de Pareto numa única execução. Isso é possível usando algoritmos evolutivos, tais como algoritmos genéticos.

Para esta abordagem do problema, nós resolvemos um sub-problema do problema da reconfiguração tal que nós o descrevemos. Nós resolvemos um problema de definição multiperíodo de topologia lógica.

Os algoritmos evolutivos geralmente funcionam da seguinte forma. Definimos primeiro uma *população*, onde cada *indivíduo* da população representa uma possível solução para o problema. Esta população evolui com opera-

ções genéticas. Uma *mutação* é uma alteração aleatória de um indivíduo. A partir de dois indivíduos, um *cruzamento* gera dois novos indivíduos. A qualidade de um indivíduo é avaliada com uma função *fitness*. Em cada iteração do algoritmo, novos indivíduos são gerados. Alguns destes indivíduos são selecionados e servem de população de origem para a iteração seguinte. Com uma função *fitness* adequada, é possível evoluir a população na direção do conjunto de Pareto. O principal problema vindo do uso de algoritmos genéticos é a ausência de garantia em relação com a qualidade das soluções obtidas.

Existem variações de algoritmos evolutivos resolvendo os problemas multiobjetivo. Nós escolhemos usar o algoritmo SPEA (*Strength Pareto Evolutionary Algorithm*), que conserva uma população externa de soluções não-dominadas. As operações genéticas são aplicadas nos indivíduos da população principal e também nos indivíduos da população externa.

Nós descrevemos a forma como codificamos uma solução num cromossomo, e a forma como realisamos as diferentes operações genéticas. A alocação dos comprimentos de onda não é integrada ao cromossomo, mas é computada separadamente usando um algoritmo guloso. Além disso, para evitar que a população externa tenha indivíduos demais, nós usamos uma procedimento que permite a supressão de soluções quando elas são muito similares. A população inicial é gerada aleatoriamente.

Problema multiobjetivo: resultados experimentais

Num primeiro momento, nós fazemos um estudo multiobjetivo superficial do problema da reconfiguração com a ajuda do método dos objetivos com pesos. Nós usamos o *solver* e nossa implementação do algoritmo do *simulated annealing* para resolver os diferentes problemas gerados. Usar como função objetivo um somatório de métricas com pesos torna o problema mais difícil de resolver. No entanto, isso permite obter compromissos interessantes.

Nós fazemos depois um estudo multiobjetivo mais detalhado usando nosso algoritmo baseado no método das $\epsilon-$restrições. Para duas instâncias, nós geramos vários problemas que nós resolvemos. Para cada instância, nós obtemos vários pontos do conjunto de Pareto, mas também soluções dominadas e soluções idênticas. Estas soluções idênticas aparecem quando as restrições-ϵ são muito próximas umas das outras, ou quando o *solver* não consegue achar uma nova solução em tempo limitado. No entanto, nós obtemos várias soluções diferentes apresentando compromissos diferentes.

Nós fazemos também muitos testes com o algoritmo evolutivo SPEA. O número de gerações necessárias para garantia da convergência depende do tamanho da instância. Mil gerações são suficientes para as menores instâncias, 1500 iterações são necessárias para a maior. Observamos que a população inteira se dirige progressivamente em direção do conjunto de Pareto durante

a execução do algoritmo. O compromisso entre o número de recursos usados e o número de reconfigurações não é muito claro, ao contrário daquele entre a carga dos enlaces e o número de reconfigurações. A relação entre o número de recursos usados e a carga dos enlaces é muito forte.

Conclusões

Nesta tese nós nos interessamos pelo problema da reconfiguração nas redes óticas WDM. É um problema complexo que inclui vários parâmetros. Nós propomos um modelo matemático usando uma formulação fonte para o problema, extensões tecnológicas, cortes e um limite inferior. Nós propomos também uma heurística gulosa e uma heurística baseada no *simulated annealing*. Nós fazemos vários testes para identificar a influência e a eficiência das diferentes propostas. No entanto, uma abordagem mono-objetivo não é suficiente para levar em consideração os diferentes compromissos envolvidos para as diferentes métricas. Nós descrevemos uma abordagem multiobjetivo. Nós propomos um algoritmo baseado no método $\epsilon-$restrito, e nós adaptamos o algoritmo SPEA. Nós fazemos testes illustrando a eficiência dos nossos métodos e dando vários compromissos possíveis.

O trabalho desta tese parte da suposição que os tráfegos futuros são conhecidos. Isso é uma hipótese forte que seria interessante relaxar, pelo menos parcialmente, conciliando previsão e adaptação. Nós tratamos a forma como a transição de uma topologia lógica a outra é realizada. Finalmente, seria interessante fornecer um algoritmo auxiliar na escolha dentre as diferentes soluções do conjunto de Pareto aquela que mais interessa o tomador de decisão.

Agradecimentos

Em primeiro lugar, gostaria de agradecer sinceramente à meus dois orientadores que me apoiaram até o fim deste projeto. Um grande obrigado à Afonso Ferreira, graças a ele eu pude trabalhar com pessoas da UFMG e começar esta aventura. Um grande obrigado também a Geraldo Robson Mateus, que soube me orientar durante esta tese apesar do meu português mais ou menos, que sempre dedicou o tempo para me dar orientações ou para ler meus trabalhos.

Meus agradecimentos à Philippe Mahey e Philippe Michelon, por revisar minha tese apesar do curto tempo que nós tínhamos e por participar de minha banca. Quero também expressar minha gratidão aos outros membro da banca, Ricardo H.C. Takahashi e Antônio A.F. Loureiro, que contribuíram com críticas e comentários permitindo destacar perspectivas interessantes sobre meu trabalho.

Durante este doutorado, tive a oportunidade de trabalhar e de evoluir com o grupo do LaPO da UFMG, dirigido pelo Geraldo R. Mateus que não poupa esforços para gerar uma dinâmica no grupo e que nos fornece meios para guiar nossas pesquisas. Expresso minha gratidão a todos as pessoas do LaPO. Trabalhei também com as pessoas do time Mascotte, do INRIA Sophia Antipolis, projeto comum com o I3S e dirigido por Jean-Claude Bermond, que soube tornar este time de pesquisa ao mesmo tempo rico e agradável. Agradeço a Jean-Claude e a todos os membros do time por suas contribuições.

Durante os quatro anos que meu doutorado durou tive o privilégio de assistir a aulas muito interessantes. Quero expressar meus comprimentos a N. Ziviani, Antônio A.F. Loureiro, Newton J. Vieira, Claudionor J.N. Coelho Jr, Ricardo H.C Takahashi, João A. de Vasconcelos, Rodney R. Saldanha, e também a StamFord. Assistindo a estas aulas, aprendi muito e com prazer, as vezes ao preço de grandes momentos de desespero ou de stress.

Minha situação administrativa não foi das mais comuns, o que não simplificou a tarefa das secretárias tanto do INRIA quanto da UNSA ou da UFMG. Eu lhes expresso minha gratidão, particularmente a Éphie Deriche e Renata V.M. Rocha. Agradeço também, aos meus amigos, Jean-François Lalande e Marie-Émilie Voge que aceitaram servir de intermediários quando estava longe de Nice.

Naturalmente, comprimento todos com quem, ao redor de um café ou através de um email, tive a oportunidade de ter conversas enriquecedores. Alguns destes gostaria de citar, como André, Bertrand, David, Fabíola, Fabrice, Filipe, Gustavo, Jean-François, Jean-Marc, João, Loic, Marie-Émilie, Martin, Michel, Nicolas, Paulo, Pável, Pedro(s), Renan, Yann, etc. Comprimento também a sem igual turma dos Gros-Nazes.

Para concluir, agradeço sinceramente à minha família, sem quem não seria nada; família esta, que aumentou de tamanho durante este doutorado.

Gurvan Huiban

The reconfiguration problem in multifiber WDM networks

July 2006

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

In the post-industrialized era, economy tends to develop services, which heavily rely on telecommunication structures. Data transport networks have been developing at a high rate since the end of World war II. The amount of exchanged information constantly evolves and increases. Different technologies have been developed and used as a result of a constant search for higher transmission rates, lower transmission delays and higher reliability. Nowadays, high performance networks are built with optical fibers. Due to various technical qualities, it is predominant for *backbones*, long-distance high-speed networks.

However, installing a large-scale optical network is an expensive and difficult infrastructure operation. The structure of the network as well as its exploitation has to be carefully designed and studied in order to go with the traffic and its evolutions all over the network lifespan. Our work takes place in this context: We are dealing with problems coming from the evolution of traffic over time on an existing optical network.

1.1 General context

In this section, we describe the recent history of telecommunication and how it leads to the use of optical fiber based networks.

1.1.1 Development of networking

The end of the eighteen century defines the beginning of an intense technological development in the area of telecommunications. Claude Chappe presented a project of optical telegraph to the French legislative assembly at the end of the eighteenth century. The French government needed a fast and efficient communication way. This system was composed of poles and mobile arms that could take various positions. Based on a specific code, it allowed the French government to transmit orders in a very short time [EB80].

During the nineteenth century, many technological breakthroughs led to the electrical telegraph as we know it. Samuel Morse achieved the first communication between Washington DC and Baltimore in 1844. Quickly adopted and developed, electrical telegraph contributed towards the first industrial revolution. The first transatlantic cable was installed in 1865. Graham Bell patented the telephone in 1876. In 1878, James Clerk Maxwell's works about electromagnetic radio-waves initiated research about radio-transmission.

In the beginning of the twentieth century, a telecommunication revolution occurred with the introduction of electronics. The transistor appeared in 1947, and the micro-chip in 1960. In 1962 was launched Telstar, the first telecommunication satellite, usable only a few hours per day due to its orbit. During the years 1970-1980, telecommunication networks were based almost exclusively on coaxial cables and radio. The transition between analogue and digital communication happened at the same time. Digital communication allows to increase the number of channels transmitted simultaneously by using efficient multiplexing techniques. It allows to integrate services by transmitting at the same time information of different kinds such as voice, image, text, data [Tél02b].

The LASER (Light Amplification by Stimulated Emission of Radiation) was invented in 1961. It is the dawning of research about optical fibers. Optical high speed networks, of 180 Megabits per second (Mbps) and then 540Mbps, were first installed in the late eighties. The idea of fully optical transport network appeared in the nineties. In 1990 also appeared the World Wide Web. The cellular phones were widely adopted during the nineties.

New trends are continuously appearing such as generalized broadband access, even for cellular and wireless devices, convergence of the media transmission and development of network-based services. The importance of the telecommunication in the world is increasing. The telecommunication market now weights billions of dollars [otEC03], as shown on Figure 1.1. The European Union telecommunication market was estimated to 251 billions euros in 2003. Considerable investments are realized by many companies to build and maintain telecommunication infrastructures. Optical networks represent one of the major elements of the overall telecommunication scheme.

1.1.2 Why optical networks?

The use of optical fiber as a medium really increased in the nineties. As a consequence of fast technological development and large capacity optical systems, first all optical transport networks were built in 1993. Nowadays optical fiber networks have overcome any other transmission technology for large-scale networks [Tél02b].

Basically, the structure of any communication in a network is the same (see Figure 1.2). The data to be transmitted is encoded and sent to a

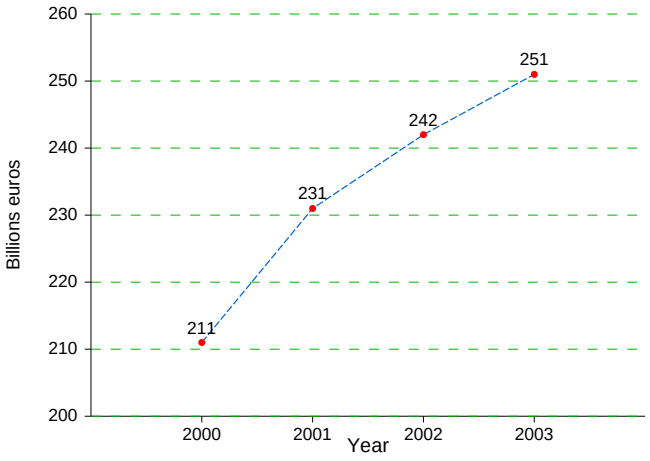


Figure 1.1: European telecommunication market

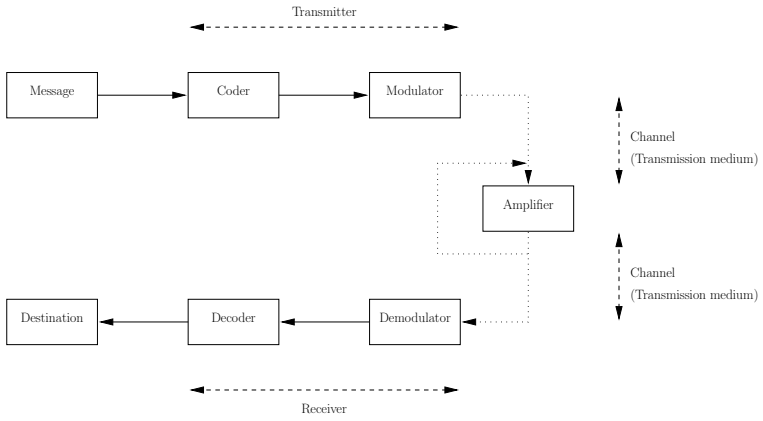


Figure 1.2: Structure of a telecommunication network

transmitter, where it is modulated, and then transmitted in a medium. The signal reaches a receiver where it is demodulated and decoded. Eventually, if the distance to be covered is very long, the signal may go across intermediary amplifiers [EB80].

Optical fiber offers some decisive quality in relation with other technologies:

- The bandwidth theoretically reachable is very high, in the order of 100Tbps (Terabits per second) [MS01];
- The error rate is low, offering a reliable communication medium;
- The energy dissipation is very low, resulting in distant amplifiers and relays (more than 50 kilometers) [RS98];

- The optical signal is insensitive to any kind of electromagnetic interferences;
- An optical fiber is light and thin, allowing to assemble tenth of fibers in a single cable;
- It offers a relative security against active and passive attacks, as it is almost impossible to listen to the canal without interrupting the optical communication.

For all these reasons, the optical fiber is the most used medium for long-distance and high bandwidth networks. The backbones rely on such technology and, as the time goes on, the optical fiber is always closer to the end user. The metropolitan and access networks are more and more based on optical fiber.

1.2 Technological context

In this section, we give an overview of the technological aspects related to optical networks.

1.2.1 Optical fiber technology

In optical networks, information is modulated on coherent monochromatic rays of light. Those rays are emitted by LASER, and have a specific wavelength. We say that the signal is transmitted over a wavelength λ. The transmission medium can be either free space or optical fiber.

Free space is for very small distances (few centimeters). Rays of light are guided with lenses and mirrors. Such technology is used for interconnection systems [MMHE93].

Optical fiber is a glass or plastic wire able to conduct rays of light over long distance. The first generation of optical fibers use geometric optics to conduct the ray of light. Optical fibers are build in a way that avoids the refraction when a ray of light reaches the border of the fiber. It is reflected inside the fiber with almost no energy dissipation, until reaching its destination, as illustrated in Figure 1.3.

Second generation of optical fibers have a core diameter of the same order of the value λ. The geometric optics laws do not apply in this context, and it is necessary to study the propagation of electromagnetic waves, as described by Maxwell's equations, to explain how a ray of light propagate over a long distance. Informally speaking, the index of the material varies smoothly in a way that maintains the ray of light in the center of the fiber.

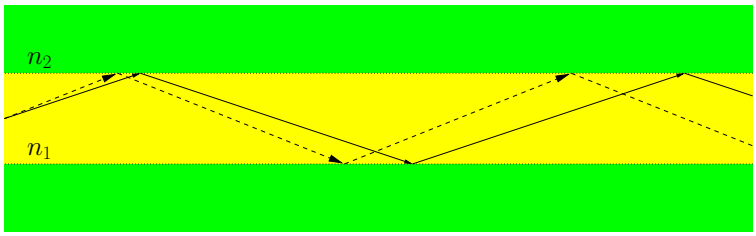


Figure 1.3: Propagation of two rays of light in a first generation optical fiber

1.2.2 Multifiber networks

Installing a large-scale telecommunication network is expensive. For instance the cost of a North-American network covering 15 cities was estimated to 200 millions dollars [Tél02a]. An important part of the expenses comes from the infrastructure operations: digging and installing cables. The thinness of an optical fiber allows a single cable to contain tenth of fibers. Consequently companies generally install many optical fibers at the same time, even if it is not required.

Installing multifiber networks offers interesting possibilities for the carrier companies. Some of the fibers can be dedicated to specific protection schemes. Some other fibers can be rented as dedicated links to large companies or organizations. The carrier can also face a temporary or permanent traffic increasing using those additional fibers.

1.2.3 Connection oriented communication

The bandwidth offered by optical technology is very high, and one limitation comes from the electronic part of the transmission scheme. The devices that are able to deal with tenths of Gigabits per seconds are very expensive. Moreover, high-speed backbone networks deal with aggregated data streams. It provides a certain stability in the traffic and the evolutions in the traffic pattern, if any, occur smoothly. Connection oriented communication with pre-established routes is a way to reduce the amount of processing in this context of slow variations.

There are researches developed in order to implement an *Internet Protocol* (*IP*) layer directly on the optical layer. This would make communications in the optical network much more dynamic, since with IP connections have generally a short lifetime. However, it would require being able to perform efficient optical routing, which is is not yet feasible technologically. IP is a packet-switching protocol. No connection setup is needed before a node sends packets to another node. The intermediate nodes have to perform the routing. Routing optically the packets is not yet feasible technologically, and routing electronically the packets would require very expensive equipments.

1.2.4 Evolving traffic

It has been noted that bandwidth requirements evolve with time, both in short-term and in long-term. During working hours and working days, the traffic is generally higher than during the night or week-end. The emergence of new kinds of applications (multimedia diffusion, voice over IP, etc) and the decreasing price for broadband access causes the traffic to increase continuously on the long-term.

Depending on the time-window considered, the traffic can vary around a base traffic (short time-window) or increase continuously (long time-window). A network has to be able to go along with this evolution all along its lifespan. The fact that the traffic will evolve has to be taken into account in the network design. But as it may be impossible to foresee some evolutions during the network design phase, it also has to be taken into account in the network management phase.

1.2.5 WDM networks

Wavelength Division Multiplexing (*WDM*) revolutionized the use of optical technology for data transmission. It is a frequency multiplexing specific to optical technology. Each signal is modulated with a LASER of specific wavelength and sent in the optical fiber. Various channels can be transmitted simultaneously in a unique fiber as long as the used wavelengths are different. Figure 1.4 represents such multiplexing technique: three signals are modulated in three different colors, but are transmitted at the same time in a unique fiber.

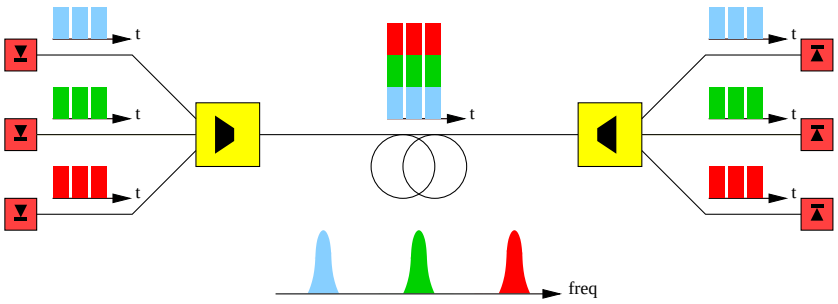


Figure 1.4: A single optical fiber transmits different independent wavelengths

With this technology commercial equipments are able to have an overall throughput greater than one Terabits per second. Transmission rates of 10 Tbps have already been achieved with experimental equipments [FCI+02].

WDM also allows a telecommunication company to expand the capacity of a network without laying more fibers. The capacity of a given link can be expanded by changing the multiplexers and demultiplexers at each end.

1.2.6 Virtual topology

The WDM technology emphasizes the importance of the optical layer responsible for transmitting the signal [RS98]. It is composed of the *virtual topology*, overtaking the point-to-point connection restriction. Within this layer, no electronic processing is performed. Such processing is avoided because it has some serious drawbacks. It requires optical-electrical-optical (O-E-O) conversion which introduces a delay. Such conversion depends on the way the signal has been modulated. Finally electronic devices that are able to process signals at very high bit rates are expensive.

A fully optical layer has some specific characteristics, as:

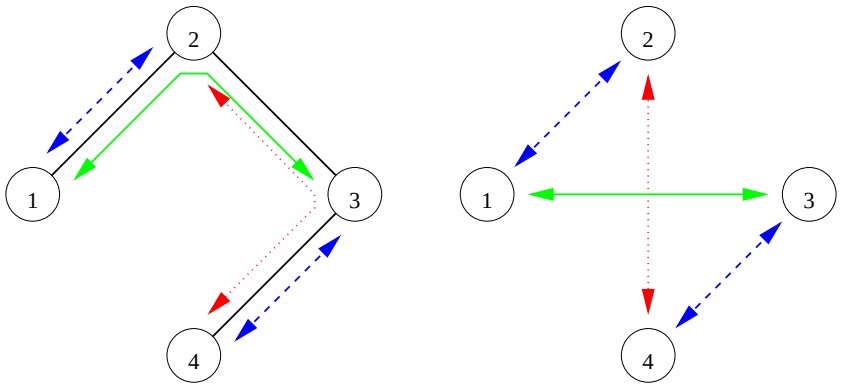
- *Independence of the wavelengths*: Wavelengths are managed independently one from another. The signal contained in one wavelength is completely independent from the signal contained in another wavelength;
- *Wavelength reuse*: The same wavelength can carry two distinct signals in two different places of the network;
- *Transparency of the optical layer*: The optical layer is independent of the modulation of the signal. It treats the signal independently of its content (bit rate, protocol, codification);
- *Reliability and protection mechanisms* in order to provide a robust system against failures.

The virtual topology, also called *logical topology*, is constituted of lightpaths. A lightpath is a connection between a pair of network nodes. It can be direct (point to point) or indirect (the lightpath goes across a succession of intermediate nodes). From a logical point of view, a lightpath from node A to node B represents an indivisible link from A to B, independently from going across intermediate nodes or not. This is illustrated on Figure 1.5(a), which shows a physical topology and the way the lightpaths are defined, and Figure 1.5(b) which shows the resulting virtual topology. Data to be sent from node 2 to node 3 will go through node 1 or node 4, even though there exists a direct link between node 2 and 3 on the physical topology.

Developing an optical layer allows to turn two nodes, non-adjacent in the physical topology, into two adjacent nodes from a logical point of view. Data using a logical direct link will not suffer from O-E-O conversion.

A single optical fiber cannot carry simultaneously two lightpaths using the same wavelength. Moreover, the wavelength remains the same all along the lightpath. This last constraint is known as *wavelength continuity constraint*.

Technically, using different wavelengths along a same lightpath is possible. Converters, which are expensive devices, have to be added. As optical



(a) Physical topology and a set of lightpaths

(b) The associated logical topology

Figure 1.5: From physical to logical topology

converters are still prototypes and are not commercially available, O-E-O conversions are required, which introduces delay.

Technologically, lightpaths are implemented statically *via* add-and-drops (the device responsible for modulating and demodulating the electrical signal on a wavelength) and switches (devices responsible for forwarding the signal). Generally, wavelengths are demultiplexed before entering a switch, where they are split and recombined. Then they are multiplexed in a different manner.

1.3 Document structure

In chapter 2, we describe the main problem that we focus on, the *reconfiguration problem*. It is based on the *Virtual Topology Design and Routing problem* (VTDR problem), which consists in finding a configuration for a network in order to transmit data. It is a core problem for the efficient use in telecommunication backbones.

In chapter 3 we focus on the mono-objective aspect of the reconfiguration problem. We propose a mathematical model to solve it. We also provide some extensions to this model. These extensions can be technological or related to the method chosen to solve the problem. Due to the complexity of the problem, it is unlikely that we succeed in solving large instances of the problem based on exact methods. To address such situation, we propose a simple and straightforward greedy algorithm and a probabilistic metaheuristic based on simulated annealing.

In chapter 4, we present computational results obtained with the mathematical model and the heuristics presented in chapter 3. We make experiments with different network topologies, allowing us to obtain interesting

conclusions about the reconfiguration problem.

The problem involves different metrics, leading us to use *multiobjective optimization*. Instead of focusing on a single objective for the problem, we consider at the same time several objectives. Such approach allows to find relationships between different metrics and provides a large amount of information useful for the decision maker. We describe the mathematical background in chapter 5 and we propose an algorithm based on our mathematical formulation performing such multiobjective optimization. We also adapt an evolutionary algorithm to our problem.

In chapter 6, we report computational results exploring multiobjective optimization. We make experiments in order to identify interesting tradeoffs of our problem. We are able to identify relationship between different metrics, providing valuable knowledge that can help the decision maker when he comes to deal with traffic evolution.

Finally, we conclude this thesis in chapter 7.

1.4 Contributions

The following elements are the main contributions of our thesis:

- we propose a source formulation for the multiperiod reconfiguration problem in multifiber optical networks;
- we adapt some extensions and cuts to our mathematical formulation ;
- we propose a cut to avoid part of the solutions that can be obtained by permutation from a base solution;
- we explore various aspects of the problem, considering various metrics;
- we adapt the simulated annealing metaheuristic to the multiperiod reconfiguration problem;
- we propose an algorithm based on the ϵ-restricted method in order to search for solutions belonging to the Pareto frontier;
- we adapt an evolutionary algorithm for the context of multiperiod virtual topology design and routing problem.

Chapter 2

The reconfiguration problem in multifiber WDM networks

In this chapter, we present the reconfiguration problem in multifiber WDM networks. It comes from the fact that the carried traffic in a network evolves with the time, and consists in studying how to alter the configuration of the network. The problem associated with the configuration of an optical network, the virtual topology design and routing problem, is first described.

Different traffic evolution models, based on collected data, are presented. It allows us to define more precisely the reconfiguration problem. We make a bibliographic study of the problem, emphasizing different points of view from different works from the literature. Many metrics can be considered to measure the quality of a solution. We present the set of metrics that we use in our thesis.

2.1 Related problems

In this section, we present the different optimization problems that lead to the reconfiguration problem, before giving its definition.

2.1.1 The VTD problem

The *Virtual Topology Design* (VTD) problem can be defined as: given a physical topology, what is the set of lightpaths providing the best performance based on a given metric (Cost of the network, used resources, quality of service)? Which path on the physical topology should each lightpath follow? Which wavelength should be assigned to it?

This problem has been proved to be NP-hard and is equivalent to the n-graph coloring problem [CGK92] in the context of monofiber networks. The structure of the problems is modified for a multifiber network. Increasing the number of fibers per link often simplifies the routing of the lightpaths.

The same physical link can be used by more than one lightpaths modulated with the same wavelengths. which reduces the conflicts about the use of a wavelength on a given physical link. However, even in the multifiber case, the VTD problem remains NP-hard in most cases [LS01, FPR$^+$03].

2.1.2 The routing problem

Each node of the network has data to send to other nodes. The *routing problem* consists of defining which lightpaths will be used to transmit those data from the *origin* nodes to the *destination* nodes, optimizing a performance criterion. Each origin-destination pair defines a commodity. The origin node is the node from which the data come from, and the destination node is the node that will receive the data. This problem can be reduced to a multicommodity flow formulation. A network flow is the assignment of a flow to each edge of a graph. Each edge has a capacity, and the flow conversation constraints have to be verified for each commodity. Except in the source or in the demand node, the amount of flow entering a node has to equal to the flow leaving the node. The complexity of the problem is polynomial as long as the flows considered can be described by continuous variables. When we consider integer flow formulation, the problem becomes NP-hard.

2.1.3 The VTDR problem

The relationship between the virtual topology and the routing problems is very strong, and it is common to consider both problems simultaneously. The *Virtual Topology Design and Routing (VTDR) problem* is the union of the virtual topology design and routing problems. Given a physical topology, solving a VTDR problem consists in defining a set of lightpaths, the path followed by each lightpath, the wavelength associated to it, and the lightpaths that will transmit the data from their origin to their destination. The VTDR problem is one of the key problems in the design of a WDM network, and it has been widely studied in the literature. As it is an extension of the virtual topology design problem, it is NP-hard.

There are many variations for this problem depending on the parameters considered such as topology (ring, grid, mesh), the technical details (single-hop or multihop, with or without wavelength converters) and the explored metrics. A common mathematical formulation for this problem considers the lightpaths as a multicommodity flow and the data as another flow on the virtual topology [Ban96]. In the survey [DR00], the authors present a mathematical model, some cuts and various heuristics about this problem.

2.1.4 Traffic models

As mentioned in section 1.2.3, the traffic carried by large scale optical networks is highly aggregated. The traffic sent by a node corresponds to the aggregation of a very high number of small amount of data. Consequently, it is likely that each node sends data to each other nodes and receives data from each other nodes. This leads us to consider all-to-all traffic pattern.

There are various works studying worldwide network traffic. However, there is no widespread agreement about its characteristics, due to the large amount of data flowing on backbones and the difficulty to collect and analyze these data in the nodes. Some models consider that traffic matrix from different periods of time are not correlated and randomly generated (see for instance [NTM00, BR99]). Other models try to simulate different kind of situations by considering a day as a set of working hours, leisure hours, night hours. One interesting work can be found in [RGK+02], where data have been collected on a backbone network during more than one year. This allowed the authors to quantify the traffic into regular predictable, and a stochastic components. According to the authors, it generates good traffic estimations.

Other consequence of the traffic aggregation is its continuous and slow evolution. In this work we discretize such evolution, making it happen step by step. We call *time period* the time between two evolutions, and it is considered sufficiently long to implement a new virtual topology and alter the routing.

We considered two different evolution patterns. In the first one, a base traffic is first defined, and low evolutions are then defined. The second evolution pattern is the same as the first one, except that in the low evolution the traffic increases from one to the following time period.

2.2 The reconfiguration problem

Consequence of such traffic evolution, the initial virtual topology and routing may not remain the optimal one, leading to a loss of performance or congestion. Therefore it becomes necessary to change the routing and the virtual topology and to reconfigure the network. The idea of using optical network configurability, adapting the virtual topology and the routing to the traffic, comes with the first optical networks (see [LA91] for instance).

The reconfiguration problem is finding out how to change the virtual topology and the routing, to keep them optimal or to maintain a good solution regarding to the traffic evolution. It may not be desirable to modify the virtual topology completely due to its implications as it may generate network interruptions which can result in high costs. Considering the huge quantity of data flowing constantly in a backbone, such an interruption must be as short as possible. The reconfiguration has to be sufficient without being

excessive.

Actually, we are facing a problem where we have to consider the trade-off between the number of changes to apply in the reconfiguration of the network and the network performance. The virtual topologies are effectively implemented via the configuration of the add-and-drops and switches. From the description of a virtual topology, the configuration of these devices can be deduced. We do not study how the transition between the different virtual topologies will be carried out. There are different methods that can be used, such as computing a succession of branch exchange operations until reaching the new virtual topology [LHA94].

2.2.1 Unique reconfiguration

A set of works only consider the reconfiguration problem in the case of a unique evolution, and not as a succession of evolutions. This reconfiguration definition with a given all-to-all traffic pattern is considered in [BM00, KO00, GM02]. Note that the VTDR problem can be seen as a reconfiguration from a non existing configuration.

In [BM00] the computation of the new configuration is performed in two phases: a configuration to minimize the resources allocated is computed first, and then is computed a solution using this amount of resources, minimizing the number of changes with respect to the current solution. The same approach has been explored in [RR00]. The reconfiguration of the virtual topology is computed using metaheuristics in [KO00]. However, the authors restrict themselves to regular grid topology. In [GM02] is proposed a *Mixed Integer Linear Programming* (MILP) model performing one reconfiguration. The authors focus on the adaptation of the virtual topology when traffic evolutions happen, which are considered low but very frequent. The computed virtual topology is at most one lightpath different from the previous one. An adaptive heuristic for dynamic traffic is also proposed.

2.2.2 Reconfiguration restricted to virtual topology

Some works in the literature consider all-to-all traffic evolving but focusing only on the reconfiguration of the virtual topology. The routing of data itself is not solved. A two phase algorithm is defined in [SMGM01]. The first phase consists in computing quickly a solution adapted to the new traffic pattern. The obtained solution is improved during the second phase. Both phases use local search. A local search algorithm starts from an existing solution and modify it slightly. If the new solution is better than the previous one, it is kept. The process is repeated various times.

The trade-off between prevision and adaptation of the virtual topology is studied in [GADO01]. Optimizing network costs for all periods at the same time is compared with year by year optimization without any knowledge

about future traffics. The focus of the article is not to solve efficiently the problem, but to compare different strategies from the network management point of view.

2.2.3 Dynamic reconfiguration

Real-time reconfiguration is a possibility to face dynamic traffic. Changing traffic may trigger some events. The configuration of the network is an adaptation in response to the event. Such approach has been chosen in [GPA+02, GM02, YR04] with all-to-all traffic. There is no way to perform in-depth resource optimization with this kind of algorithm. As the algorithm is triggered by an event, the computation of the adaptation has to be fast. Moreover without any knowledge of the future traffics it is not possible to adapt the configuration of the network in a way that eases the future adaptations. To have a knowledge about the future traffics implies making prevision. Such aspect is not present in the cited papers. An interesting discussion on the choice of an adaptation reconfiguration algorithm to be used can be found in [YR02].

2.2.4 Similar problems

In some works, a problem, very similar as the one we consider, is solved but in very specific cases. In [NTM00], the authors develop reconfiguration algorithm for ring networks. The proposed algorithm is based on branch-exchange techniques. In [BR01] a Markovian process is used to study the trade-offs involved by the reconfiguration in single-hop broadcast WDM networks.

Similar problems may also be found in the literature under the name of "dynamic traffic grooming", as [ZZZM02, JPM03]. In both works, the authors modify the initial network graph. The modifications consist in splitting nodes to represent different parts of the optical devices (electronic processing, purely optical router). It allows to use quite simple algorithms based on the shortest path [ZZZM02] or to solve the problem with elegant mathematical model [JPM03]. Unfortunately, this latter mathematical model focuses on the grooming and not on the reconfiguration.

A very interesting survey about the dynamic traffic grooming problem can be found in [HD04]. However, it focuses on the "grooming" part of the problem and does not mention many reconfiguration works. Moreover it does not consider the multifiber case. A mathematical MILP formulation is provided, which generates a very high number of variables and constraints.

2.2.5 Problem solved

In this work, we solve the following reconfiguration problem. We consider a mesh multifiber WDM network and each fiber carries a given number of

optical links. By optical link, we mean a channel able to carry information from a node to one of its neighbors. It corresponds to the use of a wavelength in a fiber linking two nodes. Each optical link can carry a given amount of data. We are given a succession of all-to-all traffic evolution matrices, one for each considered time period. In this work, we consider that the future traffic evolutions are known.

For each time period we compute a virtual topology and a multihop routing for the associated traffic matrix. There is a very large number of different possibilities for these computations and we need a way to decide which solution to consider. The quality of a solution can be evaluated with different metrics taking into account the resources used, the transmission delay and the difficulty to switch from a virtual topology to another.

We adopt two different approaches. We first consider the problem as a mono-objective problem where we choose one metric and search for the best solution according to it. We then consider the multiobjective approach, which consists in taking into account all metrics at the same time. We search for a set of best solutions and the relationship between the different metrics.

To our knowledge, there are few works based on exact methods for the reconfiguration problem. Even though the trade-off between reconfiguration and performance is regularly mentioned, few works make a comprehensive study of this trade-off [GM02, BR01].

2.2.6 Network reconfiguration implementation

The way the network reconfigurations should be implemented and the way the changes are propagated on the network is beyond the scope of this work. Some articles from the literature addressed this problem. In [LHA94] is computed a sequence of branch-exchange operations allowing to perform the reconfiguration step by step in a way that is minimally disruptive to the traffic. Such approach transforms the reconfiguration in a long process. In [BR01] is proposed an algorithm for the context of single-hop optical networks, aiming to minimize the negative effects on network performance while keeping the length of the transition phase relatively small. In [YR02] is proposed a model allowing to measure the consequences of a reconfiguration strategy on both the control plane (changes in the virtual topology) and the data plane (changes in the packet routing).

2.3 Metrics

A problem such as the reconfiguration problem involves many elements, and a solution can be quantified according to various aspects, such as the amount of resources used, the load of the network, the transmission delay or the difficulty to switch from a configuration to other when the traffic matrix changes.

The choice of a metric is a strong hypothesis in an optimization work. In this thesis we consider five metrics for the mono-objective approach and the multiobjective approach. Four of the chosen metrics are related to the physical configuration of the network, while the other one is reflects one aspect of the quality of service offered by the network.

2.3.1 Number of optical links

The number of used optical links is a usual metric and represents directly the load of the network. Each optical link is a resource in the network. If all optical links are being used, no wavelength is available in the network and a new request will be discarded. This metric is used for example in [ZZZM02].

2.3.2 Number of lightpaths

Used in [RR00, ZZZM02], this metric represents the number of lightpaths required to implement the defined virtual topology. For each used lightpath, a transmitter and a receiver is required. This has a direct influence on the cost of the switches.

2.3.3 Maximum link load

Minimizing the maximum link load, expressed in number of lightpaths, allows to balance the load between all the links or in a great part of the links. It avoids having a small set of links carrying all lightpaths. Network evolution and management is more flexible when the load is well-balanced, since there is capacity available in all links. It allows to implement dedicated protection schemes, to face a sudden traffic increase, to rent available dedicated lines. This metric is used in [GM02].

2.3.4 Average number of hops

The number of hops of a demand from node s to node d is the number of lightpaths that the data goes through. It has a direct influence on the transmission time and delay. A signal goes through electronic devices only when it enters or leaves a lightpath. Going through such devices is considered as slow as O-E-O conversions are required. A low value for the average number of hops is interesting from a quality of service point of view, as the transmission delay is directly related with the number of hops a demand makes. This metric is considered in [Ban96, GADO01].

2.3.5 Number of lightpath changes

All metrics above can be used for the VTDR problem. However, the reconfiguration problem introduces some specific metrics such as the number

of changes in the lightpath definitions. Such metric is a computation of the cost implied by the reconfiguration to be carried out, and is used in [BM00, RR00]. The changes in the allocation of the lightpaths is representative of the reconfigurations to be implemented in the optical devices (add-and-drops, switches).

2.4 Conclusion

We describe the reconfiguration problem in multifiber WDM networks and how it is related to other optimization problems in WDM networks. A bibliographical study of this problem is presented. We describe exactly the solved problem and we introduce the metrics we use to evaluate the performance of the solutions.

In the following chapter, we will consider this problem either as a mono-objective optimization problem, for which we propose a mathematical model and heuristics, or as a multiobjective optimization problem, for which we propose an algorithm and adapt some heuristics.

Chapter 3

Mono-objective reconfiguration problem

In this chapter we focus on the reconfiguration problem as a mono-objective optimization approach. One of the metrics described in the previous chapter is chosen and used as objective function to solve the reconfiguration problem. We propose different methods to solve the problem. We first present a mathematical formulation of the reconfiguration. With a dedicated software, we are able to solve optimally instances of the reconfiguration problem.

We also propose some extensions to the mathematical model. Some extensions aim to turn the modeling of the network more realistic. Other extensions aim to reduce the computation time required to solve instances of the reconfiguration problem. We also propose a lower bound for the problem. Having a good lower bound allows to have a better evaluation of the quality of a solution.

As the reconfiguration problem is complex and requires a high computation time, it may not be possible to obtain optimal solution for large instances using optimization software to solve linear programming models. To face such difficulty we propose two heuristics. The first one is a greedy algorithm. It is a simple and straightforward heuristic. The second one is a simulated annealing based heuristic. It is a more complex heuristic than the greedy one able to handle large instances of the reconfiguration problem.

3.1 Mixed integer linear programming model

As a first approach of the problem, we propose a mathematical model for the reconfiguration problem. It models any instance of the problem, and with the use of an appropriate software, a solver, we can search for the optimal solution. The model we propose is a Mixed Integer Linear Problem (MILP), which means that it uses only linear equations or inequations, and

some of its variables are integer.

3.1.1 Source formulation

A common formulation for the VTDR and derived problems is a flow formulation [Ban96]. In such formulation we define a commodity as an origin-destination flow. Therefore, there is a variable making the association between each commodity and each link, indicating if the first one uses the second one. In our case, there is a high number of commodities going through the network. The number of generated variables and constraints would be very high.

We try to obtain the most concise model as possible. The number of variables and constraints can be reduced by aggregating all commodities from a given node [Roc98]. Both approaches are equivalent if the cost associated with each edge does not depend on the commodity, which is the case of the model we propose below. Such commodity aggregation is used for the virtual topology design problem in [TMP02]. This leads us to a source formulation of the reconfiguration problem. Such formulation reduces the computer memory required to search for the optimal solution of the problem, and allows to solve it with less computational efforts.

3.1.2 Notations

We consider a network as a multigraph $\mathcal{P} = (\mathcal{N}, \mathcal{L})$ of $|\mathcal{N}|$ nodes. Each node $n \in \mathcal{N}$ corresponds to a telecommunication center. Each edge $e \in \mathcal{L}$ corresponds to a cable (m, n) between two telecommunication centers. This cable contains $\mathcal{F}_{(m,n)}$ optical fibers from node m to n. The topology is arbitrary (mesh) and not necessary symmetric: we can have $\mathcal{F}_{(m,n)} \neq \mathcal{F}_{(n,m)}$. Each optical fiber transports simultaneously $\mathcal{W}$ wavelengths $\{w \in 1, \ldots, \mathcal{W}\}$. Each one can transport a bandwidth $\mathcal{B}$, expressed in Gbps. We assume that $\mathcal{W}$ and $\mathcal{B}$ are the same on the entire network: many technological parameters (range of frequency used, kind of optical fiber) are involved, and we believe that few telecommunication providers would build an heterogeneous network. A lightpath from node i to node j corresponds to an elementary path on the physical topology.

We consider that traffic evolution occurs step by step. The overall time window is divided in $\mathcal{T}$ time periods $t_1, \ldots, t_{\mathcal{T}}$, and data changes occur when a time period ends and another begins. Each time period is long enough to implement the computed configuration. Real-time changes in the traffic do not occur.

For each origin-destination pair $(s, d) \in \mathcal{N}^2$ and for each time period t, a demand request $\mathcal{D}_{s,d}(t)$, expressed in Gbps, is defined.

We define the following variables:

- $p^i_{(m,n),w}(t)$ is the number of wavelengths w used by lightpaths having node i as origin on physical link $(m,n) \in \mathcal{L}$ during time period t.
- $c^{(i,j)}_w(t)$ is the number of lightpaths from node i to node j using wavelength w during time period t.
- $c^{(i,j)}(t)$ is the number of lightpaths from node i to node j during time period t.
- $f^s_{(i,j)}(t)$ is the flow from origin s using lightpath (i,j) during time period t.
- $\Delta p^i_{(m,n),w}(t)$ is the number of changes for the number of wavelengths w used by lightpaths having node i as an origin on physical link $(m,n) \in \mathcal{L}$, between time period $t-1$ and t.

The overall number of variables is $\Theta\left(|\mathcal{N}|^3 \mathcal{W}\mathcal{T}\right)$.

3.1.3 Virtual topology constraints

The following constraints are associated with the virtual topology design problem. We use a source flow formulation to express it. The lightpaths are considered as commodities that will flow on the physical topology.

$$\sum_{(i,n)\in\mathcal{L}} \sum_{w=1}^{\mathcal{W}} p^i_{(i,n),w}(t) = \sum_{j\in\mathcal{N}} c^{(i,j)}(t), \begin{cases} \forall i \in \mathcal{N} \\ 1 \leqslant t \leqslant \mathcal{T} \end{cases} \quad (3.1)$$

$$\sum_{(m,n)\in\mathcal{L}} p^i_{(m,n),w}(t) - \sum_{(n,p)\in\mathcal{L}} p^i_{(n,p),w}(t) = c^{(i,n)}_w(t), \begin{cases} \forall i,n \in \mathcal{N}^2, i \neq n \\ 1 \leqslant w \leqslant \mathcal{W} \\ 1 \leqslant t \leqslant \mathcal{T} \end{cases} \quad (3.2)$$

$$\sum_{w=1}^{\mathcal{W}} c^{(i,j)}_w(t) = c^{(i,j)}(t), \begin{cases} \forall i,j \in \mathcal{N}^2, i \neq j \\ 1 \leqslant t \leqslant \mathcal{T} \end{cases} \quad (3.3)$$

$$\sum_{i\in\mathcal{N}, i\neq n} p^i_{(m,n),w}(t) \leqslant \mathcal{F}_{(m,n)}, \begin{cases} \forall\,(m,n) \in \mathcal{L} \\ 1 \leqslant w \leqslant \mathcal{W} \\ 1 \leqslant t \leqslant \mathcal{T} \end{cases} \quad (3.4)$$

Constraints (3.1) express the flow conservation for each origin node i. Constraints (3.2) express the flow conservation in destination nodes n, for each wavelength. Constraints (3.3) define the number of lightpath between each pair of node. Constraints (3.4) enforce the link capacity.

As we consider multifiber networks, the wavelength capacity is related with the number of fibers in constraints (3.4). We cannot allow twice the same wavelength in a given fiber, and consequently we cannot allow a number of wavelengths greater than the number of fibers installed. Figure 3.1

illustrates constraints (3.4): It is not possible to allocate more wavelength w_1 between A and B, but there is still capacity available, since it is possible to allocate a wavelength w_2.

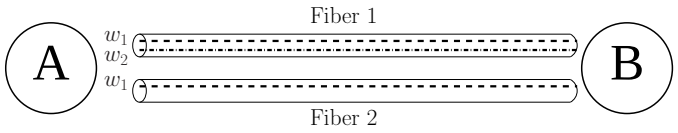


Figure 3.1: Capacity constraints have to be considered for each wavelength

The number of constraints for the virtual topology design problem is $\Theta\left(|\mathcal{N}|^2\mathcal{W}\mathcal{T}\right)$.

3.1.4 Routing constraints

The constraints below are related to the routing of the data on the virtual topology. We use a source flow formulation to express this part of the problem. The data to be transmitted are commodities that will flow on the lightpaths defined.

$$\sum_{j\in\mathcal{N},j\neq s} f^s_{(s,j)}(t) = \sum_{d\in\mathcal{N},d\neq s} \mathcal{D}_{s,d}(t), \left\{ \begin{array}{l} \forall s \in \mathcal{N} \\ 1 \leqslant t \leqslant \mathcal{T} \end{array} \right. \quad (3.5)$$

$$\sum_{i\in\mathcal{N},i\neq s} f^s_{(i,k)}(t) - \sum_{j\in\mathcal{N},j\neq s} f^s_{(k,j)}(t) = \mathcal{D}_{s,k}(t), \left\{ \begin{array}{l} \forall (s,k) \in \mathcal{N}^2, k \neq s \\ 1 \leqslant t \leqslant \mathcal{T} \end{array} \right. \quad (3.6)$$

$$\sum_{s\in\mathcal{N},s\neq j} f^s_{(i,j)}(t) \leqslant \mathcal{B}\sum_{w=1}^{\mathcal{W}} c_w^{(i,j)}(t), \left\{ \begin{array}{l} \forall\,(i,j) \in \mathcal{N}^2 \\ 1 \leqslant t \leqslant \mathcal{T} \end{array} \right. \quad (3.7)$$

Constraints (3.5) correspond to the flow conservation at origin node s. Constraints (3.6) correspond to flow conservation at destination nodes k. Finally, constraints (3.7) are the capacity constraints.

The number of constraints for the routing is $\Theta\left(|\mathcal{N}|^2\mathcal{T}\right)$.

3.1.5 Reconfiguration constraints

We consider the reconfiguration problem as a succession of VTDR problems. We can add the following constraints, in order to express the lightpath changes that occur from a time period to another. Each variation of the allocation variables (the $p^i_{(m,n),w}(t)$ variables) from a time period to another is a change of the virtual topology. Hence, it has to be taken into account. This is done by constraints (3.8) and (3.9).

$$p^i_{(m,n),w}(t) - p^i_{(m,n),w}(t-1) \leqslant \Delta p^i_{(m,n),w}(t), \begin{cases} \forall i \in \mathcal{N}, (m,n) \in \mathcal{L}, i \neq n \\ 1 \leqslant w \leqslant \mathcal{W} \\ 2 \leqslant t \leqslant \mathcal{T} \end{cases} \quad (3.8)$$

$$p^i_{(m,n),w}(t-1) - p^i_{(m,n),w}(t) \leqslant \Delta p^i_{(m,n),w}(t), \begin{cases} \forall i \in \mathcal{N}, (m,n) \in \mathcal{L}, i \neq n \\ 1 \leqslant w \leqslant \mathcal{W} \\ 2 \leqslant t \leqslant \mathcal{T} \end{cases} \quad (3.9)$$

The number of constraints for the reconfiguration is $\Theta\left(|\mathcal{N}|^3 \mathcal{W}\mathcal{T}\right)$.

3.1.6 Objective functions

Various metrics can be used to quantify the quality of a solution. The following functions correspond to the mathematical formulation of the metrics described in section 2.3.

$$O(t) = \sum_{i \in \mathcal{N}} \sum_{(m,n) \in \mathcal{L}} \sum_{w=1}^{\mathcal{W}} p^i_{(m,n),w}(t),\ 1 \leqslant t \leqslant \mathcal{T} \quad (3.10)$$

$$L(t) = \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} c^{(i,j)}(t),\ 1 \leqslant t \leqslant \mathcal{T} \quad (3.11)$$

$$M(t) = M_l(t) \quad (3.12)$$

$$H(t) = \frac{1}{\sum_{s \in \mathcal{N}} \sum_{d \in \mathcal{N}} \mathcal{D}_{s,d}(t)} \sum_{s \in \mathcal{N}} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} f^s_{(i,j)}(t),\ 1 \leqslant t \leqslant \mathcal{T} \quad (3.13)$$

$$C(t) = \sum_{i \in \mathcal{N}} \sum_{(m,n) \in \mathcal{L}} \sum_{w=1}^{\mathcal{W}} \Delta p^i_{(m,n),w}(t),\ 2 \leqslant t \leqslant \mathcal{T} \quad (3.14)$$

$O(t)$ computes the overall number of optical links used for each time period. $L(t)$ corresponds to the number of lightpaths defined for each time period. The maximum link load, expressed in number of lightpaths, is given by $M(t)$. The average number of hops for each time period corresponds to $H(t)$, and the overall number of reconfigurations between two successive virtual topologies is given by $C(t)$.

$M_l(t)$, used to compute the maximum link load, is obtained by adding the constraints (3.15) to the model.

$$\sum_{i \in \mathcal{N}} \sum_{w=1}^{\mathcal{W}} p^i_{(m,n),w}(t) \leqslant M_l(t),\ \forall (m,n) \in \mathcal{L}, 1 \leqslant t \leqslant \mathcal{T} \quad (3.15)$$

3.1.7 Integrality constraints

Some of the variables we use in our model must be integer to have a meaning in relation with the problem modeled. The number of optical links has to be integer, as well as the number of lightpaths and reconfigurations. This leads us to the following integrality constraints:

- $p^i_{(m,n),w}(t) \in \mathbb{N}$
- $c^{(i,j)}_w(t) \in \mathbb{N}$
- $c^{(i,j)}(t) \in \mathbb{N}$
- $\Delta p^i_{(m,n),w}(t) \in \mathbb{N}$

Consequently, the mathematical model proposed is a not a classical linear programing model, but a mixed integer linear programming model. Our mathematical formulation involves continuous and integer variables. Unfortunately, using integer variables generally increases the difficulty to solve a problem.

However, we can relax the integrality constraints for some variables. The $c^{(i,j)}_w(t)$ variables will necessary be integer since they are the sum of integer variables. This is also the case of $c^{(i,j)}(t)$.

Solving the problem without the associated integrality constraints, the $\Delta p^i_{(m,n),w}(t)$ variables may not be integer. However, we generally want to minimize the number of changes. This minimum will be reached only for an integer value. From constraints (3.8) and (3.9) holds

$$|p^i_{(m,n),w}(t-1) - p^i_{(m,n),w}(t)| \leqslant \Delta p^i_{(m,n),w}(t)$$

The minimum of the sum $C(t)$ will be reached when each of its term reaches its minimum value. As constraints (3.8) and (3.9) are the only restrictions assigning variables $\Delta p^i_{(m,n),w}(t)$, this minimum value corresponds to $|p^i_{(m,n),w}(t-1) - p^i_{(m,n),w}(t)|$ which is integer, as the $p^i_{(m,n),w}(t-1)$ are integer variables.

Doing so, the number of integer variables is $\Theta\left(|\mathcal{N}|^3\mathcal{W}\mathcal{T}\right)$, and the number of continuous variables is $\Theta\left(\left(|\mathcal{N}^2|\mathcal{W} + |\mathcal{N}|^3\right)\mathcal{T}\right)$. As the number of constraints is $\Theta\left(\|\mathcal{N}|^3\mathcal{W}\mathcal{T}\right)$, such model can be considered as a large model. However, we could not find in the literature smaller model for the same problem.

3.2 Model extensions

We present in this section some extensions that can be defined for the mathematical model presented in section 3.1. Such extensions may be related to technological or to the optimization aspects. In the first case, we

take into account additional modeling characteristics, while in the second case we define additional constraints in order to solve the problem efficiently.

3.2.1 Technological aspects

We provide below some extensions to the mathematical model defined in section 3.1 in order to turn the modeling more realistic.

Bounded number of receivers and transmitters

The mathematical models found in the literature generally include additional constraints. It is common to find restrictions on the number of receivers and transmitters for each node [LA91, Ban96]. Their number should be limited, considering that they are expensive devices.

If we consider that each node $n \in \mathcal{N}$ is equipped with $\mathcal{R}_n$ receivers and $\mathcal{E}_n$ transmitters, we have to add the following constraints:

$$\sum_{i \in \mathcal{N}} c^{(i,j)}(t) \leqslant \mathcal{R}_j, \begin{cases} \forall j \in \mathcal{N} \\ 1 \leqslant t \leqslant \mathcal{T} \end{cases} \tag{3.16}$$

$$\sum_{j \in \mathcal{N}} c^{(i,j)}(t) \leqslant \mathcal{E}_i, \begin{cases} \forall i \in \mathcal{N} \\ 1 \leqslant t \leqslant \mathcal{T} \end{cases} \tag{3.17}$$

Not considering optical link release as reconfiguration step

One may not want to consider the release of an optical link as a reconfiguration step, arguing that optical link can be used later on and this new assignment will be counted as a reconfiguration step. Moreover, not counting the optical link release will encourage the release of unused resources, and will consequently reduce the overall resource utilization.

To stop considering lightpath link release as a reconfiguration, it is sufficient to remove constraints (3.9) from the mathematical model.

Avoiding almost empty lightpaths

To reduce network resource utilization, the almost empty lightpaths could be avoided by adding a restriction on the lightpath occupation. In our model we do not differentiate the lightpaths, but we consider as a whole the lightpaths having the same origin-destination pair. Consequently, we cannot explicitly avoid that any lightpath will be occupied with less than a given rate, but we can force a set of lightpaths having the same origin-destination pair to have a minimal average occupation. This is expressed by constraints (3.18). $\mathcal{F}$, expressed in Gbps, corresponds to the minimal average occupation rate that a set of lightpaths having the same origin-destination pair must reach.

$$\mathcal{F}\sum_{w=1}^{\mathcal{W}} c_w^{(i,j)}(t) \leqslant \sum_{s\in\mathcal{N},s\neq j} f_{(i,j)}^{s}(t),\ \forall\,(i,j)\in\mathcal{N}^2, 1\leqslant t\leqslant\mathcal{T} \tag{3.18}$$

The $\mathcal{F}$ value has to be carefully defined. If it is too high, the problem can be infeasible.

3.2.2 Cuts

A cut is an additional constraint reducing the solution space without excluding the optimal solution. Adding adapted cuts to the mathematical formulation may help the process in charge for finding a solution or to detect earlier that a part of the exploration tree will not lead to valid integer solution. We focused on the cuts having a meaning with respect the reconfiguration problem and its modeling. We do not consider cuts coming from a polyhedral analysis of the problem.

Flow and number of lightpaths

Constraints relating the flow variables and the number of lightpaths can be defined. It "helps" making the flow variables being equal to zero if the $c_w^{(i,j)}(t)$ is equal to zero. This cut is frequently found in the literature for this kind of problem (in [Ban96] and all works based on it, for instance). It can be expressed in this way:

$$f_{(i,j)}^{s}(t) \leqslant \sum_{d\in\mathcal{N}} \mathcal{D}_{s,d}(t) \sum_{w=1}^{\mathcal{W}} c_w^{(i,j)}(t),\ \forall(s,i,j)\in\mathcal{N}^3, s\neq j, 1\leqslant t\leqslant\mathcal{T} \tag{3.19}$$

Number of lightpaths required

We can compute a lower bound for the number of lightpaths reaching or leaving a node, which allows us to define a cut. Similar cuts are cited in [DR00] in the context of the VTDR problem, and in [JPM03] in the context of dynamic traffic grooming.

Incoming traffic The sum of the demands to node d is a lower bound for the overall traffic arriving in d. As the lightpaths have a fixed capacity this imply a lower bound on the number of lightpaths to d (for each time period t). Such restrictions can be expressed in different ways. Constraints (3.20) and (3.21) give two possible formulations for expressing such restriction.

$$\sum_{i\in\mathcal{N} i\neq j}\sum_{w=1}^{\mathcal{W}} c_w^{(i,j)}(t) \geqslant \left\lceil \frac{\sum_{s\in\mathcal{N}} \mathcal{D}_{i,d}(t)}{\mathcal{B}} \right\rceil, \ \forall j \in \mathcal{N}, 1 \leqslant t \leqslant \mathcal{T} \quad (3.20)$$

$$\sum_{i\in\mathcal{N} i\neq j} c^{(i,j)}(t) \geqslant \left\lceil \frac{\sum_{s\in\mathcal{N}} \mathcal{D}_{i,d}(t)}{\mathcal{B}} \right\rceil, \ \forall j \in \mathcal{N}, 1 \leqslant t \leqslant \mathcal{T} \quad (3.21)$$

Outgoing traffic Similarly, the sum of the demands having node s as an origin is a lower bound for the overall traffic leaving s. Such cut can be expressed in different ways. Constraints (3.22)-(3.24) give three possibilities for expressing such restriction.

$$\sum_{(i,n)\in\mathcal{L}}\sum_{w=1}^{\mathcal{W}} p_{(i,n),w}^{i}(t) \geqslant \left\lceil \frac{\sum_{d\in\mathcal{N}} \mathcal{D}_{i,d}(t)}{\mathcal{B}} \right\rceil, \ \forall i \in \mathcal{N}, 1 \leqslant t \leqslant \mathcal{T} \ (3.22)$$

$$\sum_{(i,j)\in\mathcal{N}^2}\sum_{w=1}^{\mathcal{W}} c_w^{(i,j)}(t) \geqslant \left\lceil \frac{\sum_{d\in\mathcal{N}} \mathcal{D}_{i,d}(t)}{\mathcal{B}} \right\rceil, \ \forall i \in \mathcal{N}, 1 \leqslant t \leqslant \mathcal{T} \ (3.23)$$

$$\sum_{(i,j)\in\mathcal{L}} c^{(i,j)}(t) \geqslant \left\lceil \frac{\sum_{d\in\mathcal{N}} \mathcal{D}_{i,d}(t)}{\mathcal{B}} \right\rceil, \ \forall i \in \mathcal{N}, 1 \leqslant t \leqslant \mathcal{T} \ (3.24)$$

Permutation

Any permutation in the lightpath allocation will give a solution with the same performance, whatever may be the metric chosen. In other words, if a fiber can transport $\mathcal{W}$ wavelengths, there are $\mathcal{W}!$ equivalent solutions, obtained by permutation over the used wavelengths.

Avoiding those almost identical solutions may be useful. This reduces dramatically the size of the solution space, without altering the quality of the solutions.

One possibility to do so is to sort the number of allocated resources: more lightpaths use the first wavelength than using the second one; more lightpaths use the second wavelength than the third one, and so on. As far as we know, such kind of cut has never been proposed in the context of optical telecommunication networks.

Depending on the kind of resources that we want to focus on, we can express such idea in different way: constraints (3.25) refer to the number of optical links, while constraints (3.26) refer to the number of lightpath.

$$\sum_{i\in\mathcal{N}}\sum_{(i,j)\in\mathcal{L}} p_{(i,j),w}^{i}(1) = c_w, \ 1 \leqslant w \leqslant \mathcal{W} \quad (3.25)$$

$$\sum_{(i,j)\in\mathcal{N}^2} c_w^{(i,j)}(1) \quad = \quad c_w,\ 1 \leqslant w \leqslant \mathcal{W} \tag{3.26}$$

We then have to add the constraints (3.27) to the mathematical model to break the possible symmetries.

$$c_{w+1} \leqslant c_w,\ 1 \leqslant w \leqslant \mathcal{W} - 1 \tag{3.27}$$

3.2.3 A lower bound

Solving the reconfiguration problem consists in searching for a solution that is optimal with respect to the metric chosen. The objective functions associated with each metrics described in Section 2.3 are to be minimized. In the context of an objective function to be minimized, a lower bound is a value that is lower than the optimal value. The process of searching for the optimal solution of a MILP optimization problem is illustrated on Figure 3.2.

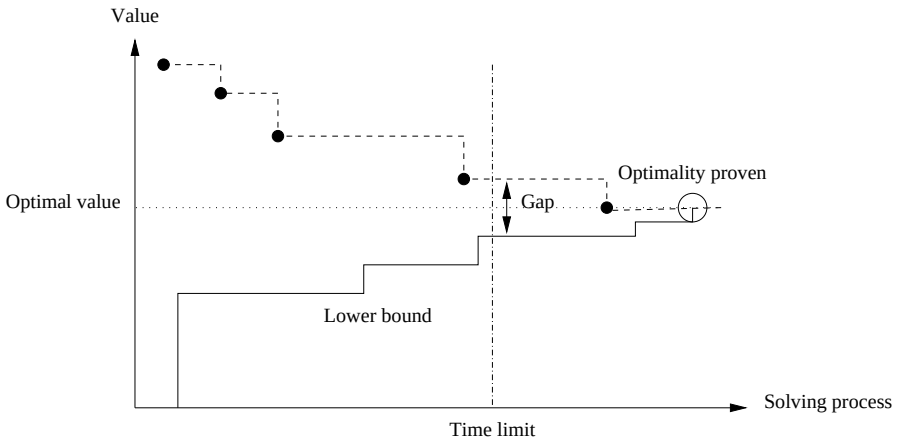
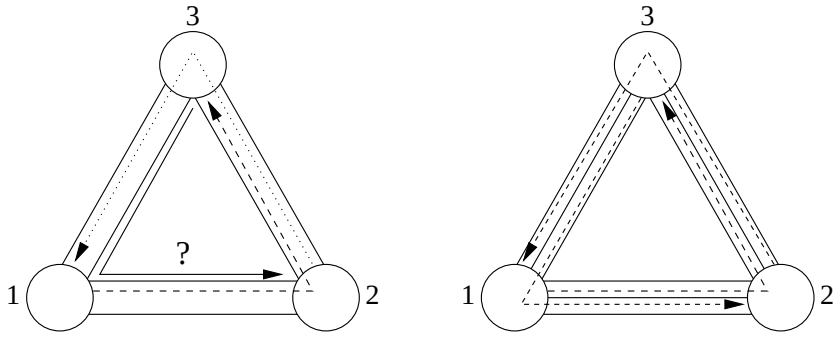


Figure 3.2: Solving process and lower bound

The process of solving an optimization problem generally involves a search for new better solutions (the black dots on the figure) which generally goes along with a search for lower bound improvements. A solution is optimal if it has the same value as the best lower bound found so far. If we decide to stop the solution process before its end, it is still possible to evaluate the quality of the best solution found so far, by giving a maximum value between the current solution and the optimal value (gap). It is known that the optimal value is the fact that it is higher than the lower bound, and consequently use this latter for the computation of the gap. For those reasons, a good lower bound is extremely useful.

In our model, the number of variables and constraints grows linearly with the number of wavelengths that a fiber is able to transmit. Instead of having $\mathcal{F}_{(m,n)}$ fibers of capacity $\mathcal{W}$ from node m to node n, we could consider that

there are $\mathcal{WF}_{(m,n)}$ fibers installed, each one able to transmit one wavelength. The overall capacity of the network remains the same, but it is impossible to have two conflicting wavelengths in the same fiber. Solving such problem should be easier than solving the original problem. The solution obtained may not be feasible for the original problem, but gives a lower bound for our problem [JMT04].



(a) Original model (b) Lower bound

Figure 3.3: Unfeasible solution for the original model

The lower bound consists in solving the same problem as the original one, except that we do not consider the wavelength conflict constraint, as shown on Figure 3.3. Since we consider fibers of unitary capacity, it is impossible to have two lightpaths using the same wavelength conflicting in the same fiber. Let us consider the three-nodes network of the Figure 3.3, with the links (1,2), (2,3) and (3,1) made of one optical fiber. The capacity of each fiber is two wavelengths. It is impossible to have simultaneously a direct lightpaths from node 1 to node 3 through node 2, from node 2 to node 1 through node 3 and from node 3 to node 2 through node 1. One of the three lightpaths would have to use a first wavelength on its first link and a second wavelength on its second one, which is in contradiction with the hypothesis of not using wavelength converters. The network considered by the lower bound has links with two fibers, each fiber having a capacity of one wavelength. With such configuration, it is possible to create the requested set of lightpaths.

With such lower bound, it is possible to have an estimation of the performance for any heuristic.

3.3 Heuristics

Although we propose a concise mathematical model and cuts to reduce the computation time required to solve the reconfiguration problem, it may be difficult or impossible to solve this mathematical formulation for large instances. To address this problem, we present a simple and computationally

efficient greedy heuristic. We also present a simulated annealing heuristics, which has been successfully applied to a virtual topology design and evolution with restoration problem [DSS03]. The main drawback of such kind of method is the lack of guarantees about the solutions quality. It is difficult to know how good is the solution obtained at the end of the process.

3.3.1 Greedy algorithm

A greedy algorithm is a constructive algorithm working step by step. In each step, it takes the decision that appears to be the best immediately and independently of the future discussions. It never goes back in its choices. Once something has been defined (a route to take, an assignment to made), it does not change, avoiding any kind of local search. As a consequence, greedy algorithms are generally simple and straightforward, and are often thought as the more natural way to solve a problem. In general they are also computationally efficient. Unfortunately, they often lead to approximate solutions [Wei97].

Our algorithm solves the reconfiguration problem as a succession of VTDR problems, taking into account the existing configuration of the network. It has been designed to give priority to single-hop routing and the lightpaths are built with the shortest path algorithm. Algorithm 1 gives a high level description of the proposed greedy algorithm. The `Configure` is an algorithm defining a virtual topology and a routing from a physical topology and a traffic matrix. The `Reconfigure` is an algorithm defining a virtual topology and a routing from a physical topology, a traffic matrix and the current virtual topology and routing. The `configure` (respectively `reconfigure`) algorithm is described by Algorithm 2 (respectively Algorithm 3).

Algorithm 1 Greedy algorithm for the reconfiguration problem

Require: A physical topology $\mathcal{P}$, a set of traffic matrices $\mathcal{D}$, one for each time period

Ensure: A solution for the reconfiguration problem or `fail`

```
S(1) = Configure(P, D(1))
if S(1) = fail then
    Return fail and exit
end if
for each time period t, 2 ⩽ t ⩽ T do
    S(2) = reconfigure(P, D(t), S(t − 1))
    if S(t) = fail then
        Return fail and exit
    end if
end for
Return the set of S(t) for each time period
```

Algorithm 2 `Configure` algorithm

Require: A physical topology $\mathcal{P}$ and a traffic matrix $\mathcal{D}(t)$
Ensure: solution $\mathcal{S}(t)$ for the VTDR problem associated or a `fail`
for all demands from s to d greater than the capacity of a lightpath **do**
If possible, build direct lightpaths from s to d and fill them with the demand
end for
for all remaining demands from s to d **do**
Route the demand using, by order of preference:

- A new direct lightpath from s to d;
- A new set of lightpaths from s to d minimizing the number of hops;
- An existing direct lightpath from s to d;
- A path from s to d on the virtual topology using existing lightpaths

end for
if There is still demand to be routed **then**
`fail`
else
Return current solution $\mathcal{S}(t)$
end if

If we consider that the complexity of the shortest path algorithm is $O\left(|\mathcal{N}|^2\right)$, which is the case for a "naive" implementation of the Dijkstra algorithm for the shortest path, the complexity of the `Configure` algorithm is $O\left(|\mathcal{N}|^4\mathcal{W}\right)$. As a consequence, the complexity of the `Reconfigure` algorithm is $O\left(|\mathcal{N}|^4\mathcal{W}\right)$, and the complexity of the overall greedy algorithm is $O\left(|\mathcal{N}|^4\mathcal{W}\mathcal{T}\right)$.

3.3.2 The simulated annealing metaheuristics

Metaheuristics, such as the simulated annealing, are generally able to find good solutions to optimization problems for an affordable computation cost.

Simulated annealing is a Monte Carlo approach for minimizing multivariable functions [KJV83]. It develops an analogy between optimization and statistical mechanics, which is the central discipline of condensed matter physics. When a system temperature decreases, the behaviour of atoms is a major concern in statistical mechanics. Whether the matter will solidify as a crystal or as a glass not only depends on the temperature, but also on the

Algorithm 3 `Reconfigure` algorithm

Require: A physical topology $\mathcal{P}$, a traffic matrix $\mathcal{D}(t)$ and a current solution $\mathcal{S}(t-1)$ for the VTDR problem
Ensure: solution $\mathcal{S}(t)$ for the VTDR problem associated or a `fail`
for All demands which decreased between $t-1$ and t **do**
Desallocate demands to reach the new value of the demand, and delete the empty lightpaths. By order of preference, desallocate:

- Multihop routing, by decreasing number of hop
- routes sharing links with other demands
- routes not filling completely lightpath
- routes filling completely lightpath

Update remaining capacity and demands
end for
for All demands **do**
Increases routing on existing lightpaths with available capacity
end for
Compute the physical topology $\mathcal{P}'$ considering only the available capacity
Compute the remaining demands $\mathcal{D}'$
$\mathcal{S}(t) =$ `configure`$(\mathcal{P}', \mathcal{D}')$
if $\mathcal{S}(t) =$ `fail` **then**
Return `fail` and exit
else
Merge $\mathcal{S}(t)$ with the current solution
Return $\mathcal{S}(t)$
end if

way the temperature is decreased. Decreasing too quickly the temperature will lead to a crystal with many defects or a glass with no crystalline order and only locally optimal structure.

Finding the best low-temperature state of a matter is similar to search for a local optimal solution of an optimization problem. However, there is no equivalent concept for the temperature in the optimization context. An heuristic converging too quickly, only accepting solution improving the overall objective function, is likely to converge toward a local optimal or a good solution.

Algorithms

A temperature for the system is defined. The algorithm progresses by lowering gradually this temperature until the system freezes. At each tem-

perature, a large number of different solutions for the problem is computed, allowing the system to reach a steady state. This process is called *thermalization*.

The system is initialized with a particular configuration. Each new solution is constructed by imposing a displacement. If the energy of this new state is lower than the previous one, this new solution is kept. If not, this new solution is accepted with a given probability. The acceptance probability decreases with the temperature of the system, allowing to explore large portions of the solution space at the beginning of the process. As the temperature decreases, the probability of accepting a bad solution decreases, leading to a local search converging towards the nearest local optima. The probability of acceptance is generally given by $\rho = \exp^{-\delta/KT}$, where K is the Boltzmann's constant, T the temperature and δ the temperature variation. With the execution of the algorithm, the temperature decreases, leading to a more stable system.

There are different possible annealing schemes to update the temperature T. We may use an annealing scheme where the temperature varies as $T_n = \alpha \times T_{n-1}$, where T_n is the temperature at the n^{th} temperature update, and α is an arbitrary constant between 0 and 1. The parameter α decides how slowly T decreases. Typical values of α lie between 0.9 and 0.95. The parameter α and the value of T_0, the initial value, plays a critical role for the performance of the simulated annealing. Annealing scheme where the temperature update is made as $T_n = T_0/(1 + \alpha \times T_{n-1})$ can also be defined. We choose to use this latter proposition. The typical values of α can be of the order of 0.01 to 0.1 to have a graceful degradation of the temperature. We call *transition* the fact that the temperature decreases, and *sub-transition* each time a problem is solved without any modification of the temperature.

We associate to each link $e = (n_1, n_2)$ of the network a weight w_e, creating the link weight vector W. Depending on the weights, different routes will be found by the shortest path algorithm. The weights of the edges are mutated by a factor γ between each sub-transition of the simulated annealing algorithm.

The simulated annealing algorithm is given by Algorithm 4. Our algorithm transforms the set of traffic matrices into an ordered list of requests, and then assign resources to each request. The `Solve` algorithm we use to generate solution is given by Algorithm 5.

The complexity of the `Solve` algorithm is $O\left(|\mathcal{N}|^4 \mathcal{W}\mathcal{T}\right)$. This gives for the Simulated annealing algorithm an overall complexity of $O\left(|\mathcal{N}|^4 \mathcal{W}\mathcal{T}XY\right)$.

3.4 Conclusion

We present a mixed integer linear programming model for the reconfiguration problem in multifiber WDM networks. This model uses a source

Algorithm 4 Simulated annealing for the Reconfiguration problem

Initialize an empty ordered list of requests R
{Transformation of the demands into a set of requests}
for $\forall i, j, t$ **do**
  Add to R $\lfloor \frac{\mathcal{D}_{i,j}(t)}{\mathcal{B}} \rfloor$ requests of size $\mathcal{B}$ and one request with the remaining traffic (lower than $\mathcal{B}$)
end for
Initialize the link weight vector W to 1
Initialize temperature T_0
Compute the initial solution: $\bar{\mathcal{S}} = \texttt{Solve}(\mathcal{P}, W, R)$
for Y transitions **do**
  for X sub-transitions **do**
    Evaluate the hop number h_r of each request $r \in R$
    Reorder the requests $r \in R$ by decreasing h_r
    $\mathcal{S} = \texttt{Solve}(\mathcal{P}, W, R)$
    if Compute $F_{\mathcal{S}} < F_{\bar{\mathcal{S}}}$ **then**
      $\bar{\mathcal{S}} = \mathcal{S}$ (update the best solution found)
    else
      $\bar{\mathcal{S}} = \mathcal{S}$ with a probability of $e^{-\frac{\delta}{KT_n}}$
    end if
  end for
  Update the link weights with $w_l = w_l\,(1-\gamma)\,, \forall l \in \mathcal{L}$
  Scale down temperature: $T_{n+1} = \frac{T_0}{1+\alpha T_n}$
end for

formulation of multiflow constraints more concise than the models for the same problem found in the literature. We provide some additional constraints in order to improve the problem modeling by bounding the number of receivers and transmitters, by not considering optical link release as a reconfiguration step or by avoiding almost empty lightpaths. We also provide some cuts to help the solver. For a given cut, we provide different formulations. We express a cut giving a lower bound for the number of lightpaths arriving in or leaving a node. We also introduce a cut avoiding solutions that can be obtained by permutation from other solutions. We also express a lower bound for the problem, as the solution of a MILP problem.

However, it may be difficult or impossible to solve optimally the problem for large instances. We present a greedy algorithm and a simulated algorithm to solve the reconfiguration problem. Such heuristics are computationally much more efficient than solving problems generated with our mathematical model. However, we do not have any guarantee with respect to the quality of the solution obtained with such heuristics.

Algorithm 5 `Solve(`$\mathcal{P}$`,`W`,`R`)` algorithm

Require: A network $\mathcal{P}$, a link weight vector W and an ordered list of requests R

for all request $r \in R$ **do**

Let s_r, d_r, t_r and v_r be respectively the origin node, the destination node, the time period and the size of r

if $v_r = \mathcal{B}$ **then**

Find the shortest path from s_r to d_r considering the wavelengths available during time period t_r. The cost of a link corresponds to its weight.

Make wavelengths allocation avoiding wavelength changes

Update available wavelengths for the time period t_r

else

if Exist paths p_r from s_r to d_r at time period t_r using only available capacity within the lightpaths able to transport a request of size v_r **then**

Use the shortest of the possible p_r

else

Find the shortest path from s_r to d_r considering the wavelengths available during time period t_r. The cost of a link corresponds to its weight.

Make wavelengths allocation avoiding wavelength changes

end if

end if

Update the available capacity in used links

Update the used link weights with remaining capacity: $w_l = w_l * v_i$

end for

Ensure: A virtual topology for each time period

Chapter 4

Mono-objective optimization: Computational results

We present in chapter 3 a mathematical model based on a source formulation for the reconfiguration problem. We also present a set of cuts to help during the search for the optimal solution. We introduce some extensions in order to turn the modeling of the network more realistic. However, as the problem is NP-hard, it is likely that we are not able to deal with large instances. We design a simple and straightforward greedy heuristic, and we adapt a metaheuristic based on the simulated annealing.

In this chapter, we make experiments in order to evaluate the difficulty of solving exactly the problem, to find out the characteristics of the solutions and to study the performance of the algorithms proposed. The reconfiguration problem admits various solutions of different quality. There may be a significant performance difference between two solutions, leading to significant differences on the network cost or on the quality of service provided. For this reason, we aim to identify the best possible solutions.

We first describe the network topologies that we use in our experiments. Some of these topologies are based on existing networks. We compare the mathematical source formulation proposed with a classical flow formulation. We then experimentally study the performance of the cuts and lower bound. We experiment such cuts on different instances in order to find out their influence on the process of searching for the optimal solution. We solve another set of instances with and without the technological extensions proposed in section 3.2.1 in order to study their influence on the solution.

The heuristics proposed solve the reconfiguration, but we do not have evaluation of their quality. We run experiments in order to compare the quality of the solutions found by the heuristics with the solutions obtained using the mathematical formulation. It allows us to have an idea of the general performance of the heuristics and the quality of the solutions.

4.1 Test instances

In this work, we mainly use four network topologies to run our experiments. Two of these networks are hypothetical small networks (SN1 and SN2), and the other two networks are based on existing networks (NSFNET and Cost239 [BDH+99]). The topology of these networks is represented by Figures 4.1 to 4.4. The characteristics of those networks are given in Table 4.1. The other used parameters depend on the instance solved. We also considered the N20, N30, N40 and N50 hypothetical networks. These topologies are representative Internet topologies.

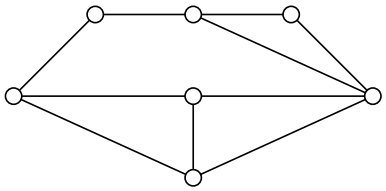


Figure 4.1: Small network 1 (SN1)

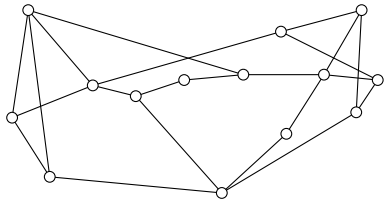


Figure 4.2: NSFNET network

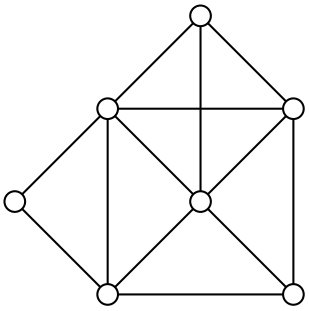


Figure 4.3: Small network 2 (SN2)

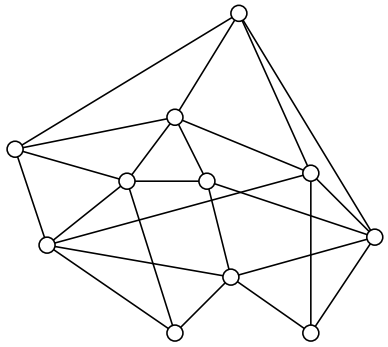


Figure 4.4: Cost239 network

Table 4.1: Networks characteristics

	SN1	SN2	NSFNET	Cost239	N20	N30	N40	N50
$\|\mathcal{N}\|$	7	7	14	11	20	30	40	50
$\|\mathcal{L}\|$	20	26	44	42	68	160	240	250

As mentioned in section 2.1.4, depending on the time scale chosen, the traffic considered can increase and decrease, or can only increase. In our experiments, we considered both cases. The first one is denoted by *var* while the second one is denoted by *incr*. The traffic matrices are generated in the following way: We first generate an initial traffic matrix, the initial demand from a node n_1 to a node n_2 being randomly chosen between 20 and

60Gbps with a uniform distribution. We then randomly choose also with a uniform distribution the evolution of the demand for each time period, based on the value of the demand at previous time period. For the *var* traffic, this evolution is between -10 and 10 Gbps, while for the *incr* traffic it is between 0 and 10Gbps. We generally considered five time periods. For instance, with a *var* traffic, it is possible to have the following traffic from node A to node B over five time period: 57Gbps, 67Gbps, 75Gbps, 68Gbps 77Gbps.

We run our experiments on a desktop PC with one gigabyte of RAM. To solve the mathematical models, we use the commercial software Cplex[1] version 9. To solve the MILP problems, it uses a *branch and bound* algorithm. It consists in exploring a recursively build tree of possibilities. Each node of the tree is obtained solving a relaxed version (e.g. without the integrality constraints) with additional constraints on the value of the variables. Cplex also uses additional cuts for accelerating the search for the optimal value.

For the vast majority of the run experiments, we imposed the solver a time-limit T_L, expressed in seconds. This means that if the computation reaches the time-limit, the solver does not guarantee an optimal solution. It returns the best solution found so far, if any. It also returns an evaluation of the quality of the solution, by the mean of a solution gap. There is no way to know if the solution returned by the solver is the optimal one or not. We use the following convention: The $^{+}$symbol means that the solver hit the time limit imposed, but has already found a feasible solutions. In this case, we report the solution found by the solver and the gap in parenthesis. The ^{0}symbol means that the solver hit the time limit without finding any possible solution. The *symbol means that we are unable to complete the experiment, due to a lack of memory.

4.2 Classical formulation and source formulation

Firstly we want to compare the classical and the source formulation. By classical, we mean the origin-destination flow formulation where a commodity is defined for each origin-destination flow, as evoked in section 3.1.1. This classical formulation can be found in appendix A. Structurally, the main difference between these formulations appears when solving flow problems with a Dantzig-Wolfe decomposition algorithm [DW60]. With an origin-destination formulation, a high number of simple problems (shortest path) are solved; with a source formulation, a lower number of more complex problems (shortest path tree) are solved [JLFP92]. Even though those problems are more complex, they involve less constraints and variables.

The three following aspects are considered: computation time, problem size and solution quality. We run experiments with the SN2, Cost239 and

[1] Copyright ©Ilog 1997-2005. Cplex is a registered trademark of Ilog.

NSFNET instances. The parameters chosen for our experiments are given in Table 4.2. The computation times with the two models are given in Table B.1 and Table B.2. The solutions found are given in Table B.3 and Table B.4, and the problem sizes are given in Table B.5. These tables are located in the appendix B. Figures 4.5 and 4.6 illustrate the computation time required for solving the instance SN2 and Cost239 with the objectives M, C and H. Figures 4.7 and 4.8 represent the number of integer variables, real variables and constraints depending on the formulation, for the SN2 and Cost239 instances.

Table 4.2: Parameters to compare the formulations

	SN2	Cost239	NSFNET	N20
$\mathcal{F}_{(i,j)}$	5	5	5	3
$\mathcal{W}$	8	16	8/16	8
$\mathcal{B}$	40	20	40/20	40
Traffic	*var*	*var*	*var*	*var*
T_L	3600	3600	3600/7200	7200

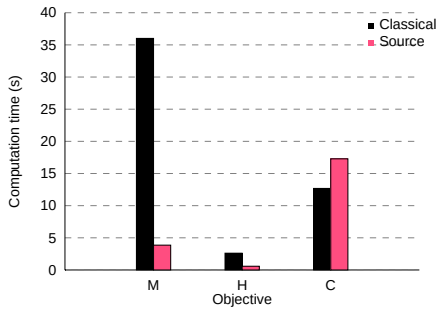


Figure 4.5: SN2 instance, computation time (s)

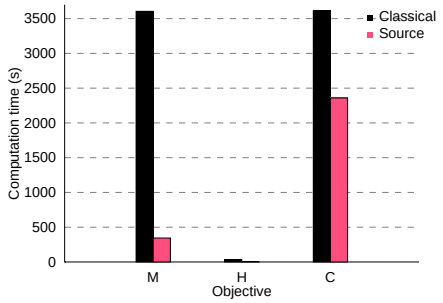


Figure 4.6: Cost239 instance, computation time (s)

The problems generated with our source formulation are much smaller than the ones generated with the classical formulation. The impact on the computation time is difficult to evaluate exactly, since the majority of the computations reaches the time-limit. When it is not reached, we observe that the computation time for the source formulation is significantly lower.

Given the same amount of time, the solution found is generally better with the source formulation, with the exception of the problem solved optimizing the L objective function. In every cases the solver achieved a slightly better lower bound and the number of nodes explored in the branch and bound tree is much higher with the source formulation. The classical formulation find its computational limits for networks that the source formulation is still able to handle, emphasizing the compactness of our formulation.

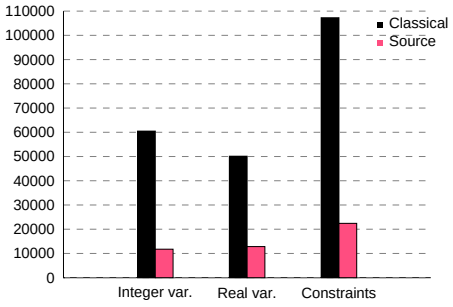


Figure 4.7: SN2 instance, number of variables and constraints

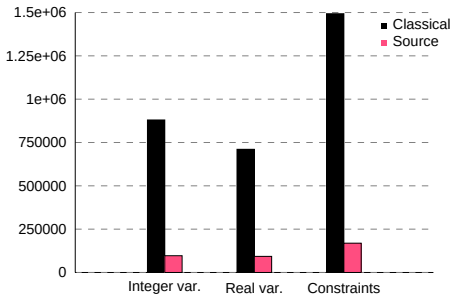


Figure 4.8: Cost239 instance, number of variables and constraints

4.3 Metrics and cut efficiency

In this section we study the efficiency of the cuts proposed in section 3.2.2, when added to the mathematical formulation proposed for the reconfiguration problem.

4.3.1 Parameters chosen

We solve instances of the reconfiguration problem on the four networks described in section 4.1. We also consider a four nodes bidirectional line network *line4*. The parameters chosen for each network are given in Table 4.3.

Table 4.3: Parameters to analyze the cuts efficiency

	Line4	SN1	SN2	NSFNET	Cost239
$\mathcal{F}_{(i,j)}$	2	4	4	5	4
$\mathcal{W}$	5	10	10	40	16
$\mathcal{B}$	40	40	40	10	40
Traffic	*var*	*var*	*var*	*incr*	*incr*
T_L	3600	3600	3600	3600	3600

4.3.2 Computation time

The most striking fact we observe is the evolution of the computation time with respect to the metric chosen. Metrics O and L appear to be the hardest to solve on our test instances. The time-limit is regularly reached, even for small instances. On the other hand, metrics M and H seem to be quite easy to solve, since the time-limit is not hit even for large instances (Cost239 and NSFNET). The situation is different for metric C. When the network is small (SN1 and SN2), the optimal solution is found very quickly. For larger networks (NSFNET and Cost239), the optimal solution is found

after almost one hour of computation for the NSFNET, and no solution is found for the Cost239 network.

4.3.3 Cut performance

We made some experiments in order to evaluate the performance of the cuts defined in section 3.2.2. We solved the reconfiguration problem, adding to the original model one cut. To avoid "interferences" with Cplex's cuts, we disabled all of them.

Figures 4.9 to 4.12, illustrates the computation time and the solution gap for some instances solved. We denote by *nocut* the results obtained without considering any cut. The results obtained considering the cut defined by constraints (3.19) is denoted by *flow*. The cut about the incoming traffic described by constraints (3.20) (respectively (3.21)) is denoted by *in1* (respectively *in2*). Similarly, the cut about the outgoing traffic described by constraints (3.22) (respectively (3.23), (3.24)) is mentioned by *out1* (respectively *out2*, *out3*). Finally the permutation cut described by constraints (3.25) (respectively (3.26)) is denoted by *sym1* (respectively *sym2*). We represent the computation time only for experiments ending before the time limit (Figures 4.9 and 4.11), and the solution gap only for experiments for which the solver hit the time limit (Figures 4.10 and 4.12). The data used to plot the graphs are extracted from the Tables B.6 (respectively Table B.7) containing the computation time (respectively solution gap) for the instances solved. Those tables are located in the appendix B.

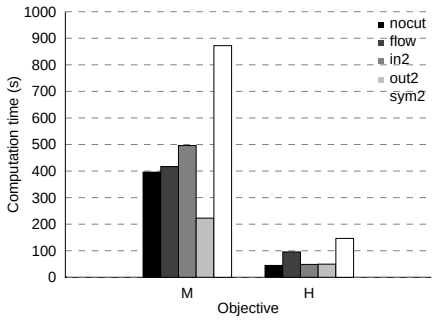


Figure 4.9: NSFNET instance, computation time (s)

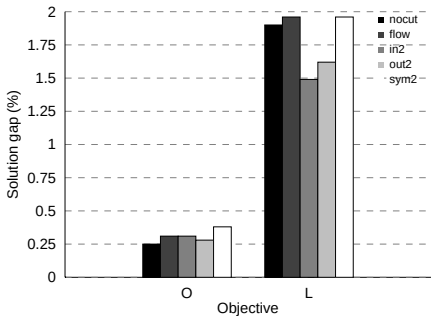


Figure 4.10: NSFNET instance, solution gap (%)

In our experiments, the cut defined by constraints (3.19) has a negative impact on the computation time or on the solution gap. With and without this cut the solver generally finds the same solution after exploring the same number of nodes of the branch and bound. But this exploration is slower with the cut. Due to the additional restriction defined by constraints (3.19) the relaxed problems generated during the branch and bound algorithm are

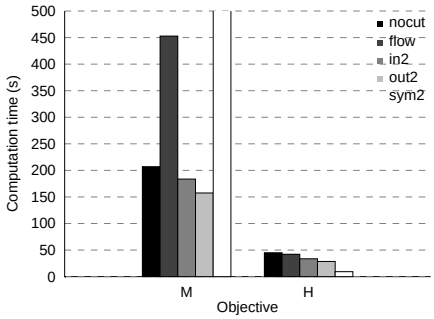


Figure 4.11: Cost239 instance, computation time (s)

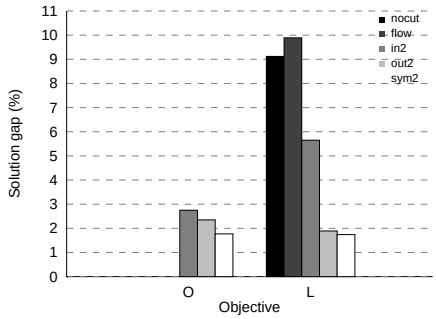


Figure 4.12: Cost239 network, solution gap (%)

harder to solve but lead to the same solution.

When we include the permutation cut described by constraint (3.25), the exploration of the branch and bound tree is much slower than without the cut. This cut may eventually have a positive impact when the solution process is not interrupted, but that has not been verified. We do not know if the gain resulting in a smaller solution space balance or not the longer time required to solve each intermediate linear program. The permutation cut described by restriction (3.26) has better results than the previous one. The solver succeeds in exploring much faster (but still slower than without any cuts), the branch and bound tree. This allows to have computational performance (computation time, solution gap, solution found) very close to the *nocut* formulation.

The cuts about the minimum number of lightpaths required for incoming or outgoing traffic appear to be useful. Each of them improves significantly the lower bound while performing the branch and bound algorithm, resulting in a lower solution gap. However, the exploration of the branch and bound tree is significantly slower with the cut defined by the constraint (3.22) and the lower bound is not improved. With our test instances, it is difficult to decide which formulations are the most efficient, between formulations (3.20) and (3.21), and between formulation (3.23) and (3.24).

The second formulation of the cut avoiding permutations has more or less the same performance that the reference model, while the cuts about the number of lightpaths and the ingoing or outgoing traffic improve the solution gap and have generally a positive impact on the computation time.

4.3.4 Lower bound performance

We make experiments to evaluate the performance of the lower bound described in section 3.2.3. We also show the results obtained by relaxing the integrality constraints (e.g. linear relaxation).

Figures 4.13 and 4.14 illustrate the performance of the lower bound and the linear relaxation for the Cost239 instance. We denote by *ref* the results obtained by our model for the reconfiguration problem, by *lb* the results obtained by the lower bound proposed in section 3.2.3 and by *rel* the results obtained with the linear relaxation. These results are extracted from the Table B.8, located in appendix B, which contains the results and computation time of the instances solved.

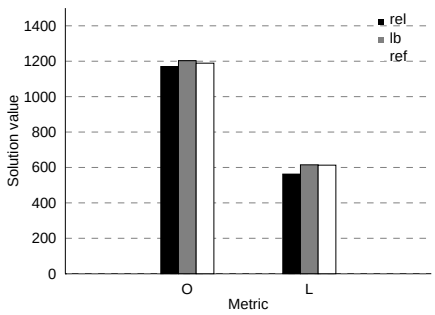


Figure 4.13: Cost239 instance, solution value for metrics O and L

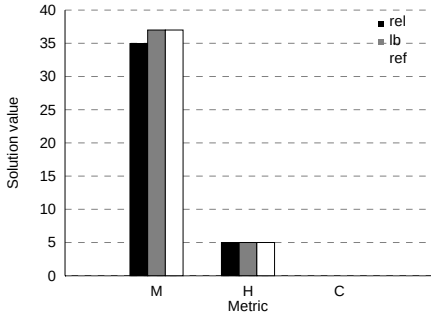


Figure 4.14: Cost239 instance, solution value for metrics M, H and C

On our test instances the performance of the lower bound is good when considering the objective functions H, M and C, since it always reaches the optimal value. The computation time is more or less the same than the one required to solve the linear relaxation of the problem. The value of the lower bound found by the lower bound *lb* are better than the one obtained with the linear relaxation. However, when solving the problem with objective O and L, our lower bound reaches the time-limit of one hour, and consequently, the optimal value may not be found. In this situation, the result provided by the lower bound is useless. Since the optimal value is not reached, there is no guarantee that the obtained value is actually lower than the value obtained by the original reconfiguration algorithm (see lines 1 and 2 of the Cost239 instance in Table B.8).

Figure 4.15 shows the evolution of the solution found during the optimization process, for the reconfiguration model *ref* and for the lower bound *lb*. Remember that the lower bound is valid only when the solver proves the optimality of the obtained solution, which is not the case on this example. An initial solution is found very early with the lower bound *lb* (120 seconds) while it is necessary to wait for 1300 seconds for the model *ref*. The number of branch and bound nodes explored within the time-limit while computing the lower bound is much higher (190000 versus 51000). That is, the intermediate linear programming problems to be solved are much easier to solve. Remains to be solved the convergence problem of the lower bound.

The linear relaxation provided solution of good quality with our test

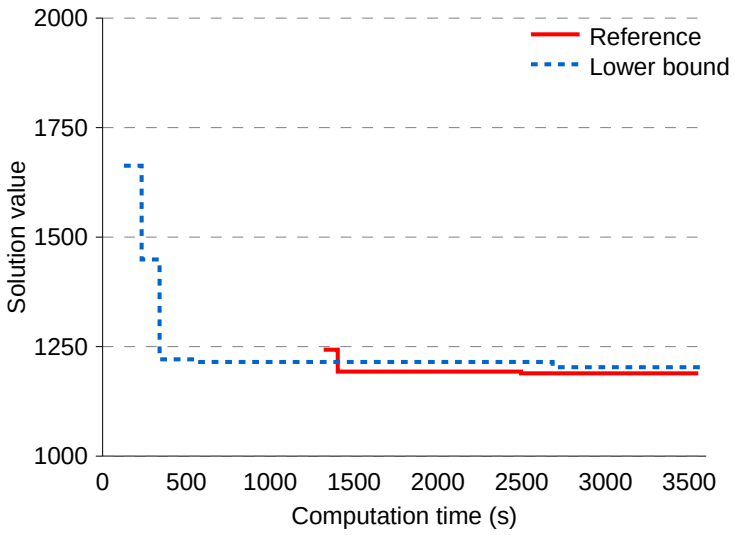


Figure 4.15: Cost239 instance, evolution of the solution found with the computation time

instances (less than 5% below the optimal solution). The computation time is very low (few seconds), except when the objective is C: The solver seems to have difficulty to solve the linear relaxation of the large instances with the objective function C. As solving the linear relaxation of a problem is one of the first steps of a branch and bound algorithm, the computation time it requires has a direct implication in the computation time required by the overall branch and bound algorithm. It is worth noting that with our test instances and for metrics H and C, the linear relaxation finds the optimal solution.

4.3.5 Metric influence

We observe that, depending on the metric chosen, the solutions found and their characteristics are quite different. Table 4.4 gives the value of each metrics, depending on the metric chosen to perform the optimization, for the NSFNET network.

Minimizing the number of optical links makes the lightpaths use the shortest path between each pair of nodes. Consequently, the load is not balanced. This can be particularly observed on a sparse topology like the networks SN1 and NSFNET. There is also a tendency to fill as much as possible each optical link. This generates solutions with short lightpaths (1.56 hops in average) and highly aggregated traffic. As the lightpaths are short and few resources are used, the number of reconfigurations to carry out is small in relation to other metrics (2734 reconfigurations while with

Table 4.4: NSFNET instance, solution depending on the metric optimized

	Metric optimized				
	O	*L*	*M*	*H*	*C*
Num. optical links	3136	3460	4507	3684	7300
Num. lightpaths	2011	1521	2401	1658	3470
Max. physical load	137	133	104	139	292
Avg. number of hops	2.69	2.03	3.02	2	3.98
Num. reconfigurations	2734	3206	4077	3386	0
Avg. lightpath length	1.56	2.27	1.87	2.21	2.1

the other metrics it is above 3000), except when minimizing the number of reconfigurations.

Minimizing the number of lightpaths aims to create as few lightpaths as possible and fill them as much as possible. It creates long direct lightpaths from the origin to the destination as long as they can be completely filled. The remaining traffic is aggregated in a set of short lightpaths. Consequently, the average hop number is low (2.03, almost the optimal value 2), and the average lightpath length is high (2.27 in average).

Minimizing the physical load allows to find the solutions with more balanced load between all metrics. The load of each link is very low (104, while with the other metrics, it is above 130), however the overall resources used is high, as well as the number of reconfigurations. The lightpaths are short and the average number of hops is high (3.02).

Minimizing the average number of hops tends to define only direct point to point lightpaths (2), which has a negative impact on the resources used. There is no traffic aggregation. The lightpaths are long, and a high number of reconfigurations have to be carried out.

Finally, minimizing the number of reconfigurations will allocate all the lightpaths that will be required at a moment or another. Obviously, this reduces greatly the number of reconfigurations to be carried out (0), but also drastically increases the resources used (7300, more than twice the optimal value 3136). In our experiments, it has always been possible to find a solution without any reconfiguration when minimizing the number of virtual topology modifications. Such solutions may include lightpaths that are empty during some time periods. It means that if we can afford using many resources, it is possible to avoid any reconfiguration. In other words, it is possible to avoid the drawbacks of the reconfiguration by oversizing the virtual topology. As the amount of resources allocated is very high, the physical load is very high, but appears to be well balanced.

4.3.6 Other parameters

To evaluate the influence of the network topology, we solve the problem with the same traffic matrix on the networks SN1, SN2 and line7, a seven nodes bidirectional line network. For all metrics, the computational results obtained are very similar from a network to another: The computation time and the gap obtained are almost the same. In other words, the topology does not seem influence much Cplex when it searches for the optimal solution.

We also check the influence of the technical parameters on the problem by experimenting different values for the $\mathcal{F}_{(i,j)}$, $\mathcal{W}$ and $\mathcal{B}$ parameters with the same traffic matrices. As the number of constraints and variables directly depends on the value of $\mathcal{W}$, the memory occupation during the search for the optimal solution is higher with high values of $\mathcal{W}$, resulting in higher computation time. The other parameters do not have significant influence on the solution process: The computation time, number of branch and bound nodes explored and solution gap are almost the same. When the capacity of the optical links is reduced, the number of lightpaths defined increases, but the solutions keep the same profile. The difference between the metrics, as described in section 4.3.5 remains the same. The $\mathcal{F}_{(i,j)}$ parameter do not seem to have a significant influence on the solution process and on the solutions found.

We then focus on the influence of the traffic on the problem. The traffic considered in the vast majority of our experiments can be considered as intermediate - neither light nor heavy. We generate heavy traffics, almost reaching the maximum capacity of the network. The computation times observed are very similar than with our other experiments, except for metric C, where it is significantly higher (118 seconds for the SN2 network, against 20 seconds for the intermediate traffic). As the number of used optical links is much higher, this increases the combinatorial aspects of the lightpaths reconfiguration. We also observe that the solution gaps for metrics O and L are very low (less than 1%). As the traffic is very heavy, there are few possibilities to decrease the number of optical links/lightpaths used.

We also compare the influence of *incr* and *var* traffics. To do so, we generate traffic with comparable parameters, and solve the problem on the same network. The computational efforts required to solve the problem do not seem to be influenced by the traffic type, and the solutions have the same characteristics.

4.4 Technological extensions of the model

4.4.1 Receivers and transmitters

The restrictions (3.16) and (3.17) are regularly present in optimization model related to WDM networks (see [LA91, Ban96, Rou01]). Although

such restriction can improve the realism of the network model, we did not find articles about the consequences of using such restrictions.

We run experiments with and without such restriction and we compare both the quality of the obtained solutions with respect the different metrics presented. We use the SN2 and Cost239 networks to run our experiments. The parameters chosen are given in Table 4.5.

Table 4.5: Parameters to analyze the influence of adding receivers and transmitters

	SN2	Cost239
$\mathcal{F}_{(i,j)}$	5	5
$\mathcal{W}$	8	16
$\mathcal{B}$	40	20
traffic	*var*	*var*
T_L	3600	7200

The number of receivers and transmitters has been homogeneously defined in the network. That is, $\forall n \in \mathcal{N}, \mathcal{R}_j = \mathcal{R}$ and $\mathcal{E}_n = \mathcal{E}$. Table B.9 gives the computation time of the different instances. The solution values are given by Table B.10. These tables are located in appendix B. We present only the values for $\mathcal{R}$ and $\mathcal{E}$ that lead to feasible solutions. For values of $\mathcal{R}$ and $\mathcal{E}$ lower than the ones given in the Table B.10, the problem is infeasible.

The obtained results are illustrated by Figure 4.16 (respectively 4.17) which shows the computation time (respectively solution found) with the SN2 network for different values of $\mathcal{R}$ and $\mathcal{E}$.

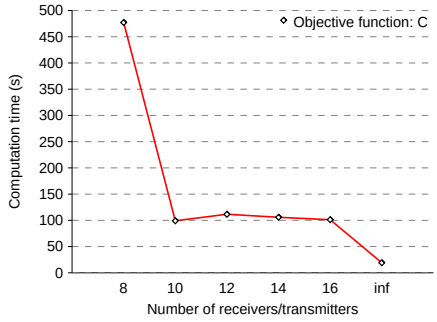


Figure 4.16: SN2 instance, computation time (s)

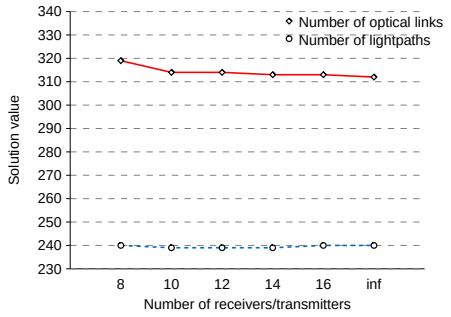


Figure 4.17: SN2 instance, solution value for metrics O and L

It appears that when including the restrictions (3.16) and (3.17), the problem is more difficult to solve, particularly for tight values of $\mathcal{R}$ and $\mathcal{E}$ (e.g. $\mathcal{R} = 8$ for SN2 network, and $\mathcal{R} = 24$ for Cost239 network). However, the influence of such restrictions on the solution value only exists for those tight values of $\mathcal{R}$ and $\mathcal{E}$. In this case, the number of used optical links

is a little higher. The number of used lightpaths does not seems to be assigned. When minimizing the number of optical links, the solver tends to establish lightpaths using the shortest paths. The nodes located in the middle of the networks have a higher load. Limiting the number of receivers and transmitters force the solver to use longer paths, thus increasing the number of optical links used. However, the difference between the solution qualities is low and exists only for the tightest values of $\mathcal{R}$ and $\mathcal{E}$. In the other cases the restriction does not seem to alter the solution quality.

4.4.2 Optical link release

We try to identify the effect of not counting as reconfiguration the deletion of a lightpath. We run experiments with the SN2 instance and with the Cost239 instance. The parameters chosen for our experiments are given in Table 4.6. The solutions found for each time period, and the overall number of reconfiguration are given in Table B.11, located in appendix B, and are illustrated by figures 4.18 and 4.19.

Table 4.6: Parameters to release the optical links

	SN2	Cost239
$\mathcal{F}_{(i,j)}$	5	5
$\mathcal{W}$	8 / 16	8
$\mathcal{B}$	40 /20	40

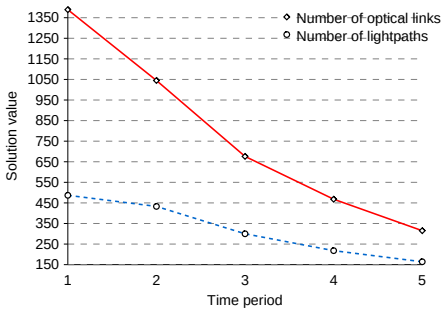


Figure 4.18: SN2 ($\mathcal{W} = 16$) instance, solution value for metrics O and L

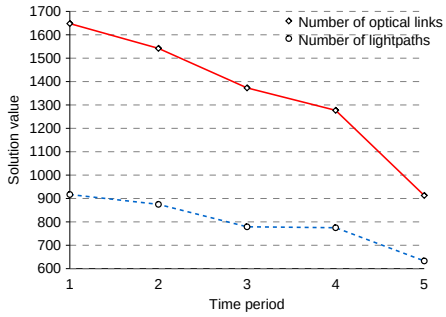


Figure 4.19: Cost239 instance, solution value for metrics O and L

Such modification has a huge impact when minimizing the number of reconfigurations and a low one when minimizing another objective function, since it does not have further influence that the way the reconfigurations are counted. The solver tends to allocate at the beginning all the resources it will need, and then release them when they are not needed anymore.

4.4.3 No empty lightpaths

When we minimize the number of reconfigurations, the solutions found tends to allocate all the lightpaths that will be needed during a time period or another, even if some lightpaths remain empty during some time period. To avoid such behaviour, we define in section 3.2.1 the constraints (3.18) forcing the lightpaths to be filled at a rate of $\mathcal{F}$ or more.

The parameters chosen for our experiments are given in Table 4.7. Table B.12, which is located in appendix B, gives the results we obtained with different values for $\mathcal{F}$. We represent on Figure 4.20 the results obtained with metric H.

Table 4.7: Parameters to avoid empty lightpaths

	SN2	Cost239
$\mathcal{F}_{(i,j)}$	5	5
$\mathcal{W}$	8	16
$\mathcal{B}$	40	20
T_L	3600	7200

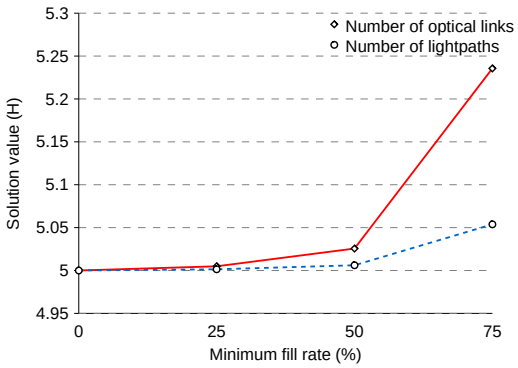


Figure 4.20: SN2 and Cost239 instances, solution value for metric H

The influence of the minimum lightpath fill rate is only noticeable on the average number of hops. The solution value for other metrics do not seem to alter by the constraints forcing the lightpaths not to be empty. With the constraints added, the solver "artificially" increases the length of the path chosen for routing the data, increasing the fill rate of the lightpaths. Instead of using a direct lightpath for routing data between nodes, it makes the data use two or three lightpaths, resulting in an higher average fill rate in order to attend the constraints.

4.5 Heuristic comparisons

We implement the simulated annealing and the greedy algorithms presented in section 3.3 with the C++ language. We run experiments with those algorithms and compare the results obtained with the mathematical model. We compile the programs with the GNU compiler `g++` 4.0[2]. We make our experiments on the small network SN2, Cost239, NSFNET, and on the N20, N30 and N40 networks. The parameters we consider for those experiments are given in Table 4.8.

Table 4.8: Parameters to compare heuristics

	SN2	Cost239	NSFNET	N20	N30	N40
$\mathcal{F}_{(i,j)}$	5	5	5	5	5	5
$\mathcal{W}$	8 /16	8/ 16	8 /16	8/ 16	8/ 16	8 / 16
$\mathcal{B}$	40 /20	40/ 20	40 /20	40/ 20	40/ 20	40 / 20
traffic	*var*	*var*	*var*	*var*	*var*	*var*
T_L	36000	36000	36000	86000	86000	86000

For the simulated annealing experiments, the total number of sub-transitions at a given temperature is chosen between 10-15 and the transitions across different temperatures is considered to be between 30-40 based on the size of the demand sets. These numbers are chosen because they are moderate enough for the simulated annealing to show different possible solution sets. These numbers are empirically chosen.

The K constant is chosen such that, $0 \leq \exp^{-\delta/(K \times T_i)} \leq 1$ where T_i is the temperature at the i^{th} iteration. The temperature mutation parameter α is taken to be 0.005 so that the temperature does not drop abruptly. Higher values of α leads to a fast convergence for the simulated annealing procedure. We mutate the values of α so that the simulated annealing procedure explores the maximal possible solution states, and shows no further improvements. The edge weight mutation parameter γ is chosen to be between 0.5 and 1.0.

4.5.1 Performance Analysis

We denote by *gr* the data obtained with the greedy algorithm. Remember that our greedy algorithm gives priority to single-hop routing and lightpaths following the shortest path. We denote by $sa(O)$ (respectively $sa(L)$ and $sa(C)$) the data obtained when solving an instance with the simulated annealing and using metric O (respectively L and C). In a similar way, we denote by $md(O)$ (respectively $md(L)$ and $md(C)$) the data obtained when solving an instance with the solver and using metric O (respectively L

[2]Copyright ©2006 Free Software Foundation, Inc.

and C). The computation time of the different instances is reported in Table B.13. The number of optical links each solution uses is given in Table B.14. Table B.15 gives the number of lightpaths. Finally, Table B.16 gives the number of reconfigurations that will have to be carried out for each solution. Those tables are located in appendix B, and are illustrated by Figures 4.21, 4.22 and 4.23 Each figure shows the results obtained optimizing one metric, even if we represent the results obtained optimizing each metric.

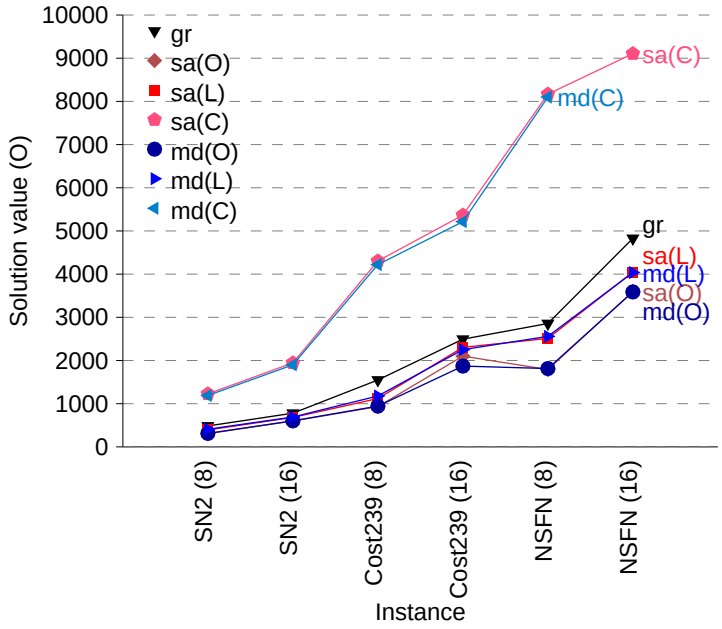


Figure 4.21: Heuristics: Solution value for metric O

The computation times of the greedy algorithm are really low, even for large instances. The quality of the results of the greedy algorithm is variable. However, it generally represents a trade-off, since the solutions found use an intermediate level of resources, triggering a limited number of reconfigurations.

The computation time of the simulated annealing algorithm is also low in comparison with the computation time obtained using the solver. As for the solver, the computation time depends on the metric chosen for performing the optimization. The solutions obtained with the simulated annealing algorithm compete with the one obtained with the solver. However, it is able to handle large instances in a reasonable computation time. The chosen parameters allowed it to explore a part of the solution space large enough to find a good solution. It returns solutions which are within 5% of the optimal solution found by the solver, with computation times at least 4-6 times lower.

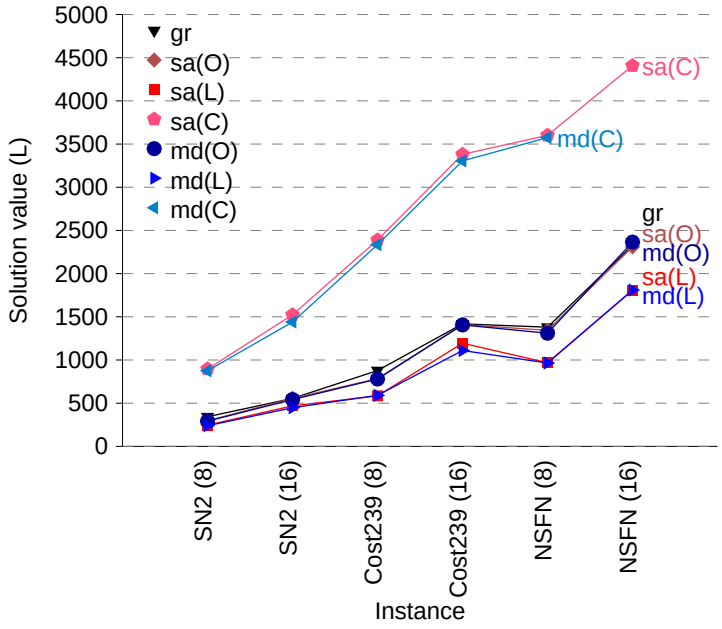


Figure 4.22: Heuristics: Solution value for metric L

4.5.2 Single hop cases

The greedy algorithm has been designed to generate virtual topologies giving priority to single-hop routing, and all the instances generated admit solutions with single-hop routing. For comparison purpose, we add the constraints (4.1) to the mathematical model forcing single-hop routing. We run another set of experiments in order to compare the results obtained with the greedy algorithm and the results obtained with the mathematical model. The parameters chosen for the computation of the results are given in Table 4.9.

$$H(t) = 1,\ 1 \leqslant t \leqslant \mathcal{T} \tag{4.1}$$

Table 4.9: Parameters to evaluate the greedy algorithm performance

	SN1	SN2	Cost239	NSFNET	N20	N30	N40	N50
$\mathcal{F}_{(i,j)}$	5	5	5	5	5	5	5	5
$\mathcal{W}$	8	16	16	8	16	16	16	16
$\mathcal{B}$	40	20	20	40	20	20	20	20
T_L	3600	3600	7200	7200	-	-	-	-

Table B.17 gives the results obtained with the greedy algorithm, compared with the MILP approach. The Table B.18 gives the results obtained

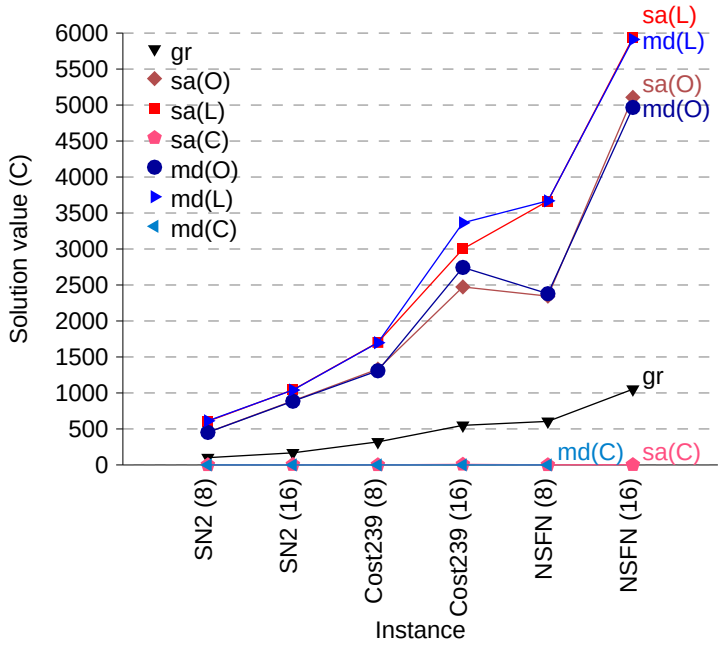


Figure 4.23: Heuristics: Solution value for metric C

for the N20 and larger instances. These tables are located in appendix B, and Figures 4.24, 4.25 and 4.26 illustrate the results obtained with the SN2 and Cost239 instances. Note that the results *md(c)* do not appear on the figure 4.26 because the value equals 0. The greedy algorithm always finds an intermediary solution in comparison with the mathematical model. It uses more optical links than the optimal value, but less than the solutions obtained with the solver when optimizing the objective function L or C. Similarly, the solution obtained with the greedy algorithm generates less reconfigurations than the solutions obtained optimizing the objective function O or L.

We report in Figure 4.27 (respectively Figure 4.28) the number of optical links (respectively lightpaths) used during each time period. We also report on the graphs the resources used with solutions obtained by the solver, restricting the number of reconfiguration to the one obtained with the greedy algorithm. For the Cost239 network, the greedy algorithm finds a solution triggering 758 reconfigurations. Consequently, we also solved the problem with the constraint $C \leqslant 758$. Note that both lines "Solver" and "Solver with $C \leqslant 758$" are overlapping. With our instances the greedy solution does not use more resources during the first time period than the solutions obtained with the solver. However, when it comes to reconfigure, that is after the first reconfiguration step, the level of resources used by the greedy

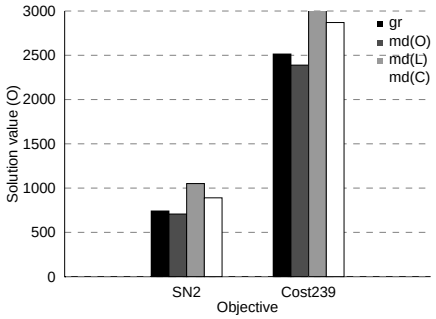


Figure 4.24: SN2 and Cost239 instances, solution value for metric O

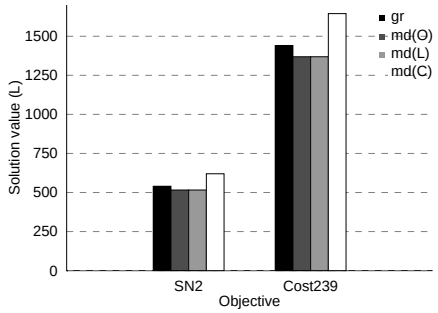


Figure 4.25: SN2 and Cost239 instances, solution value for metric L

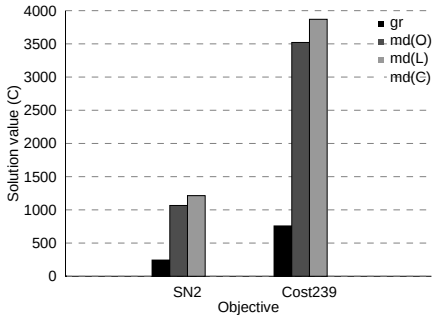


Figure 4.26: SN2 and Cost239 instances, solution value for metric C

algorithm increases while it remains more or less constant with the solver.

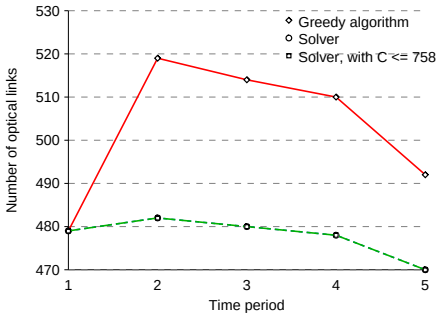


Figure 4.27: Cost239 instance, solution value for metric O

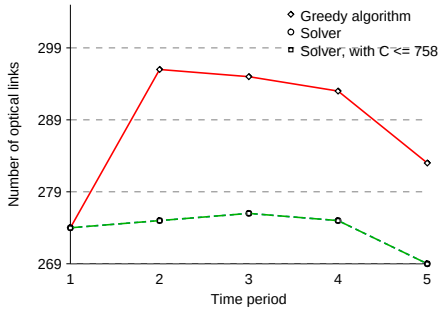


Figure 4.28: Cost239 instance, solution value for metric L

Finally, we generate a test instance with the N50 network, running over 500 time periods. The computation time is still low with respect to the instance size, since it requires a few minutes (12 minutes and 21 seconds)

on our computer to run. The resources used are represented on Figure 4.29. The phenomenon previously noted also happens. The number of resources used notably increases from the first to the second time period, and remains more or less constant during the later time periods.

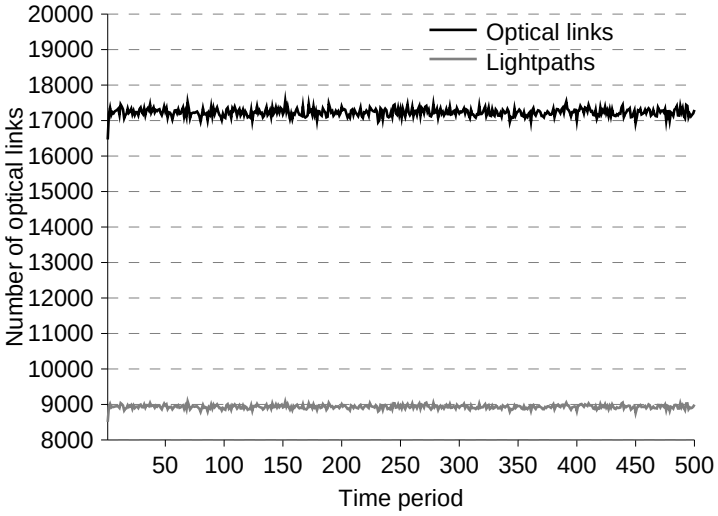


Figure 4.29: Greedy heuristic, Long N50 instance

4.6 Conclusion

We propose a source mathematical formulation for the reconfiguration problem, as well as some possible extensions for the mathematical model. Some cuts are adapted from similar problems or introduced. We also propose a lower bound. As the computation time required to solve exactly different instances of the problem is high, we define a greedy algorithm and adapt a simulated annealing algorithm.

Our formulation is much more compact than the other formulations from the literature and allows to deal with larger instances. For the smaller instances, our formulation generally outperforms the classical formulation.

According to our experiments, the most efficient cuts among the one studied are the cuts about the number of incoming and outgoing lightpaths. They allow to reduce the computation time and the solution gap. One of the two cuts avoiding the permutation seems useless for experiments with a time-limit. The other cut offers a much better performance and actually helps to decrease the computation time in some cases. The cut linking the number of lightpaths and the flow variables, which is commonly found in the literature, has a negative impact with all of our instances.

The lower bound proposed outperforms the linear relaxation, especially when we optimize the objective function H, M or C. For the other two objectives, it suffers from a difficulty to converge towards the optimal solutions.

Depending on the objective function chosen, the computation time required to solve the problem is drastically different. When optimizing metrics O and L, the solver has difficulties to converge toward the optimal solution. The computation time when optimizing metrics M and H is low. The situation is different with the C metric: The computation time is very low for small instances, but the solver is unable to find any solution for large instances, due to the fact that it does not succeed in solving efficiently the linear relaxation of the problem.

The network physical topology does not seem to influence much the complexity to find the optimal solutions with the solver. The traffic used does not have influence on the search for the solution, except with the C objective function. In this case, the solver requires more time to reach the optimal solution if the traffic is heavy. The values for the technological parameters $\mathcal{F}_{(i,j)}$ and $\mathcal{B}$ do not have significant influence on the computation time. On the other hand, high values for the parameter $\mathcal{W}$ increase the computation time, as the number of constraints and variables linearly depends on $\mathcal{W}$.

The characteristics of the solution depend on the objective function chosen. When minimizing O, we obtain a solution with short filled lightpaths, and a controlled number of reconfigurations. Minimizing L, we obtain long direct lightpaths and a set of small lightpaths, resulting in a low average number of hops and a high number of reconfigurations. Minimizing the maximum link load increases the use of resources and the number of reconfigurations to be carried out. Minimizing the average number of hops tends to generate long direct lightpaths, which has a negative impact on the resources used and the number of reconfigurations to be performed. Finally, minimizing the number of reconfigurations generates solutions allocating a very high number of resources, in a way that the virtual topology is able to handle any of the traffics given without having to be reconfigured.

Limiting the number of receivers and transmitters has an impact on the solution value only when the limit is very tight with respect to the minimum value possible without turning the problem infeasible. Not considering the release of a lightpath as a reconfiguration, the solution found will change only when minimizing objective C. However, it does not alter fundamentally the solution found. The resources are massively allocated for the first time period, and released when not needed anymore. Imposing a minimum fill rate for the lightpaths turns out to be useless. The routing of data "artificially" increases the lengths of the paths, in order to increase the overall flow of data and consequently fill the lightpaths.

The greedy algorithm allows to obtain solutions of decent quality with

a very short computation time. The number of resources used during the first time period even compete with the one obtained with the mathematical model. However, the number of resources used after the first reconfiguration step is significantly higher. The solutions obtained with the greedy algorithm represent an interesting trade-off as the number of reconfigurations is moderate. The simulated annealing algorithm allows to obtain solutions which compete, in terms of quality, with the optimal solutions. The computation times are much lower than the ones required with the exact method.

Chapter 5

Multiobjective reconfiguration problem

Many optimization problems accept different metrics to measure the quality of a solution. This is the case of the reconfiguration problem. As we saw in chapter 4, considering separately the different objectives may lead to results that do not consider the trade-off between the different metrics. If we want to reduce the number of reconfigurations, the solutions found oversize the network capacity. On the other hand, if we want to reduce the resources used, the solutions found require a high number of reconfigurations.

A common way to deal with this problem is to add some restrictions to prevent from finding uninteresting solutions, or to give a weight to each metric and solve the optimization problem with the weighted sum as an objective function. However, such approach generally requires a deep knowledge of the problem treated, since the restrictions and the weights cannot be chosen randomly. As we observed in section 4.4, adding restrictions to avoid some undesired behavior may be difficult, as the solver is able to generate not practical solutions in order to attend the restrictions. For instance, when we impose a minimum fill rate for the lightpaths, the solver tends to artificially increase the length of the routes in order to increase the traffic flowing on the network.

Generally, the choice of a metric is made *a priori* before the beginning of the optimization process. This approach lacks of flexibility and lets the decision maker facing a problem to be solved - the choice of a metric - *before* knowing the results of the optimization process. When a solution is obtained, there is no way to evaluate its quality with respect to other performance criterion. In other words, the decision maker has to choose one metric, and hope that the obtained result does not have undesirable performance in relation with other ones.

Multiobjective optimization, also called *vectorial optimization*, avoids this drawback. It does not compute a unique, but a set of "good" solutions.

Carrying out such analysis while designing a large scale network can provide a significant amount of information, the relationship between metrics, to the decision maker *after* the optimization process.

5.1 Mathematical aspects

In this section, we describe the mathematical aspects involved by the multiobjective optimization.

5.1.1 Vectorial optimization

A mono-objective minimization[1] problem can be formulated as:

$$\begin{array}{l} \min_x f^i(x) \\ x \in \mathcal{P} \end{array}$$

where

$$f^i : \left\{ \begin{array}{rcl} \mathbb{R}^n & \rightarrow & \mathbb{R} \\ x & \mapsto & f^i(x) \end{array} \right.$$

is one of the possible metrics of the optimization problem, initially chosen. $\mathcal{P}$ is the set of feasible points.

With multiobjective optimization, the scalar objective function is replaced by a vectorial objective function. The components of the objective function are the different metrics considered for the problem. The optimization is performed in $\mathbb{R}^m$, where m is the number of metrics. A multiobjective optimization problem can be defined in the following way:

$$\begin{array}{l} \min_x F(x) \\ x \in \mathcal{P} \end{array} \tag{5.1}$$

where

$$F : \left\{ \begin{array}{rcl} \mathbb{R}^n & \rightarrow & \mathbb{R}^m \\ & & \\ x & \mapsto & F(x) = \left(\begin{array}{c} f^1(x) \\ \vdots \\ f^m(x) \end{array} \right) \end{array} \right.$$

f^i are the different metrics considered and $\mathcal{P}$ is the set of feasible solutions.

[1] In this chapter, we consider that any objective is to be minimized. This is not a loss of generality, since $\max_x f(x) = -\min_x -f(x)$

5.1.2 Pareto optimal set

In classical optimization, as the objective function is a scalar, the solution space is included in a line. The optimization process aims to identify one of the extremities of this solution space. In multiobjective optimization, the solution space is a part of $\mathbb{R}^m$. There is no total order relation in $\mathbb{R}^m$, and as a consequence there is not a single but many "best solutions", forming a region called *Pareto optimal set* or *Pareto optimal frontier* [Par96].

For all vectors $x, y \in \mathbb{R}^m$, we define the following notation:

$$\begin{aligned} x \leqslant y &\Leftrightarrow \{x_i \leqslant y_i, i = 1, \ldots, n\} \\ x < y &\Leftrightarrow \{x_i < y_i, i = 1, \ldots, n\} \\ x = y &\Leftrightarrow \{x_i = y_i, i = 1, \ldots, n\} \\ x \neq y &\Leftrightarrow \{\exists i / x_i \neq y_i\} \end{aligned}$$

The objective function of our optimization problem is in $\mathbb{R}^m$. We say that a point x *dominates* a point y if $x \leqslant y$ and $\exists j / x_j < y_j$. This is written $y \prec x$.

The set of nondominated points of $F(\mathcal{P})$ is the *Pareto optimal set*. It is included in $\mathbb{R}^m$ and is constituted of the non-dominated points. It has a special interest, since it represents the set of all solutions verifying the following statement: if we want to improve the performance in relation with a metric, we have to decrease the performance in relation with at least another one [CH83].

The *ideal point*, corresponds to the point in the solution space which each component reaches its minimum possible value. Such point may not have an antecedent in $\mathcal{P}$, as a given solution is probably not the best one for all the metrics at the same time. The existence of such point means that there is no "trade-off" between the different metrics. It also means that the Pareto optimal set is a single point. Even in this case, multiobjective optimization brings additional information in relation with mono-objective optimization. With classical optimization, we would find the same solution, but we would not be aware of the situation. The *nadir point* is the point which components are the worst value for each objective among the $\bar{F}^i$.

More formally, if we denote by $\bar{x}^i$ the solution of the following mono-objective problem:

$$\begin{aligned} &\min_x f^i(x) \\ &x \in \mathcal{P} \end{aligned}$$

We define $\bar{f}^i = f^i\left(\bar{x}^i\right)$ and $\bar{F}^i = F\left(\bar{x}^i\right)$. The ideal point can be defined as follows:

$$\bar{F} = \begin{pmatrix} \min_{i=1\ldots m} f^1\left(\bar{x}^i\right) \\ \vdots \\ \min_{i=1\ldots m} f^k\left(\bar{x}^i\right) \\ \vdots \\ \min_{i=1\ldots m} f^m\left(\bar{x}^i\right) \end{pmatrix} = \begin{pmatrix} \bar{f}^1 \\ \vdots \\ \bar{f}^k \\ \vdots \\ \bar{f}^m \end{pmatrix}$$

The nadir point can be defined as follows:

$$\dot{F} = \begin{pmatrix} \max_{i=1\ldots m} f^1\left(\bar{x}^i\right) \\ \vdots \\ \max_{i=1\ldots m} f^k\left(\bar{x}^i\right) \\ \vdots \\ \max_{i=1\ldots m} f^m\left(\bar{x}^i\right) \end{pmatrix} = \begin{pmatrix} \dot{f}^1 \\ \vdots \\ \dot{f}^k \\ \vdots \\ \dot{f}^m \end{pmatrix}$$

Let us give an example for a hypothetical multiobjective optimization problem with two objectives represented by functions f^1 and f^2. The solution space, represented on the Figure 5.1, is included in $\mathbb{R}^2$. $\bar{F}^1$ and $\bar{F}^2$ are points obtained when the minimum of function f^1 and f^2 is reached.

Even though the values $f^1(x)$ and $f^2(x)$ are worst than $\bar{f}^1$ and $\bar{f}^2$, x belongs to the Pareto set, represented by the continuous line: there is no point performing better for both f^1 and f^2 at the same time. On the other hand, y does not belong to the Pareto optimal set, since there are points with lower values for both f^1 and f^2, x for instance. The ideal point and nadir point are also represented.

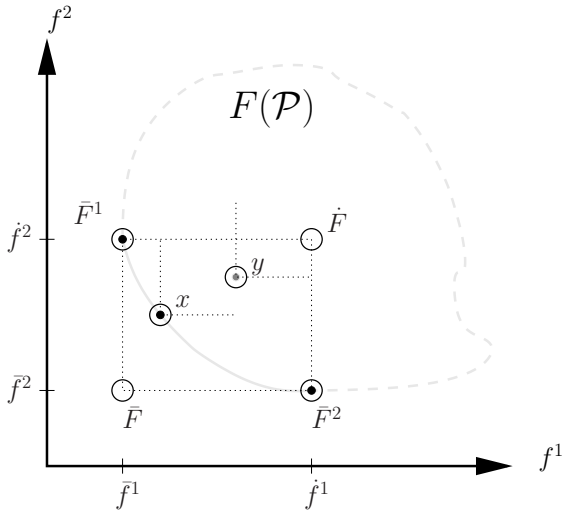


Figure 5.1: Pareto optimal set, ideal point $\bar{F}$ and nadir point $\dot{F}$

5.1.3 Combinatorial optimization problems

We consider in this work a combinatorial problem. Some variables can only take discrete values and the solution set is also discrete. Figure 5.2 illustrates a hypothetical combinatorial optimization problem in $\mathbb{R}^2$. The image by F of each element of $\mathcal{P}$ is represented by a point. The black points represent the Pareto optimal set. The gray points are dominated solutions. The delimitation corresponds to the image of the solutions without considering the integrality restrictions.

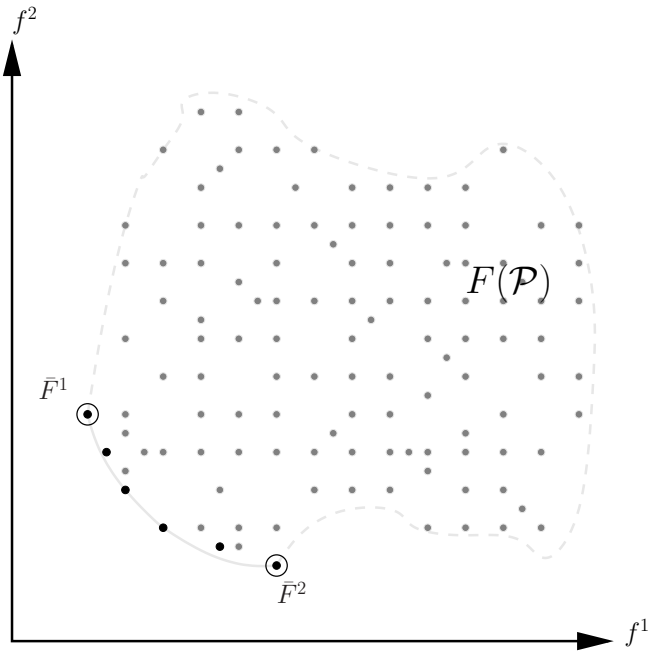


Figure 5.2: Pareto optimal set for combinatorial optimization

Depending on the chosen algorithm to search the Pareto set, the combinatorial nature of the problem may be a difficult task. As for mono-objective optimization, discrete solution sets generally increase the overall solving difficulty. For instance, direction based algorithms are generally avoided in such case, since nothing guarantees that there exists a solution in an arbitrary direction. An interesting survey about multiobjective combinatorial algorithms can be found in [EG00]. The authors first present an exact and an approximation methods for multiobjective combinatorial problems, and then an annotated bibliography problem by problem.

Solving a multiobjective problem consists in identifying the Pareto optimal set. Once the Pareto optimal set is identified, the decision maker can choose the solution that fits the best his needs.

In the vast majority of the cases, it is not possible to identify analytically the Pareto optimal set. We want to have the best approximation possible to

the Pareto optimal set. This can be done by computing one by one points belonging to the Pareto optimal set.

5.2 Mathematical model based methods

Many algorithms have been developed to search for the Pareto set of a multiobjective problem. A generic approach consists in solving many modified mono-objective problems, whose solutions belong to the Pareto optimal set.

5.2.1 Weighted objectives method

In a first classical approach we give weights to each of the f^i and we consider as an objective function the weighted sum of the f^i. If we consider a multiobjective problem described by equation (5.1) and if we define a vector $\lambda = \begin{pmatrix} \lambda^1 \\ \vdots \\ \lambda^m \end{pmatrix}$ such that each λ^i is positive, solving the mono-objective optimization problem

$$\begin{cases} \min_x \sum_{i=1}^m \lambda^i f^i(x) \\ x \in \mathcal{P} \end{cases} \tag{5.2}$$

generates a point belonging to the Pareto optimal set. This is illustrated on Figure 5.3. The direction minimizing the overall objective function changes with the weight associated to each objective function.

This naturally leads to the following algorithm to identify points of the Pareto optimal set: generate a set of $\lambda-$vector and then solve for each $\lambda-$vector the mono-objective problem (5.2) associated.

If the Pareto optimal set is not convex, some of its points may not be found by such algorithm. This is illustrated by Figure 5.4: The y point belongs to the Pareto set but will never be found by the weighted objective method whatever may be the weights.

5.2.2 Relaxation method

This method was first introduced in 1983 [CH83], and detailed in [TPF97]. The idea is to choose a vector pointing toward the Pareto optimal set. From this vector and a scalar, a multidimensional polygon is built. Reducing the scalar reduces the polygon size. The method of relaxation minimizes the value of the scalar such that the intersection between the polygon and the solution space is not empty.

Let be the cone of origin $\bar{F}$ and generated by vectors
$\left(\bar{F}^1 - \bar{F}\right), \ldots, \left(\bar{F}^m - \bar{F}\right)$

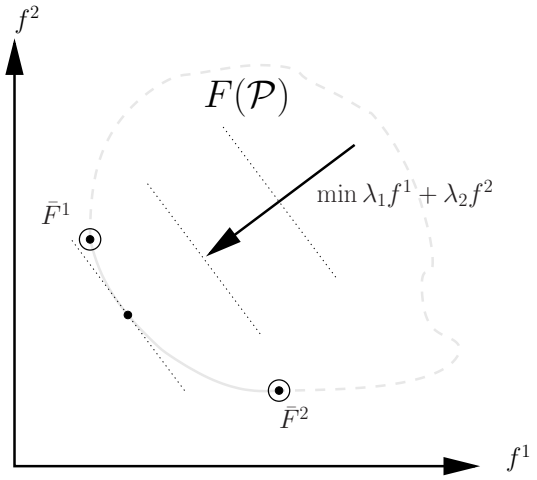


Figure 5.3: Weighted objective method for finding points of the Pareto optimal set

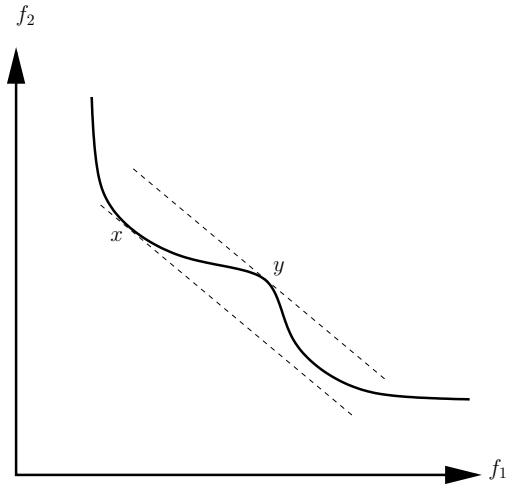


Figure 5.4: Weighted objective methods misses y

Let us define a vector $\eta = \begin{pmatrix} \eta_1 \\ \vdots \\ \eta_m \end{pmatrix}$ such that each η_i is positive, and $w = \sum_{i=1...m} \eta_i \left(\bar{F}^i - \bar{F}\right)$.

We can define the following mono-objective optimization problem:

$$\begin{cases} \min_{x,\eta} \eta \\ F(x) \leqslant \bar{F} + \eta w \\ x \in \mathcal{P} \end{cases} \tag{5.3}$$

The solution of such problem belongs to the Pareto optimal set. A graphical interpretation is shown on Figure 5.5. The brightest gray polygon is a polygon defined by the ηw vector from the $\bar{F}$ point. The intersection between the polygon and the solution space decreases with the value of η. The solution of the optimization problem returns the lower value for η such that the polygon still intersects the solution space. In the case of continuous optimization, this intersection reduces to a unique point of the Pareto optimal set.

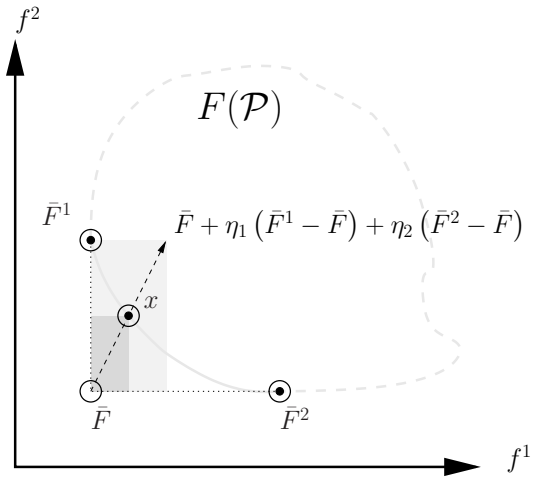


Figure 5.5: Relaxation method

Unfortunately, this method may not be adapted to combinatorial optimization, as shown on Figure 5.6. The algorithm may return dominated points. For the minimum η guaranteeing that the polygon still contains feasible points, there may be more than a unique solution and some of these solutions may be dominated, and consequently not belonging to the Pareto optimal set.

In our case there is a more critical problem with this method. As mentioned in section 3.1.7, some of the integrality restrictions can be relaxed in order to reduce the amount of integer variables, considering that those vari-

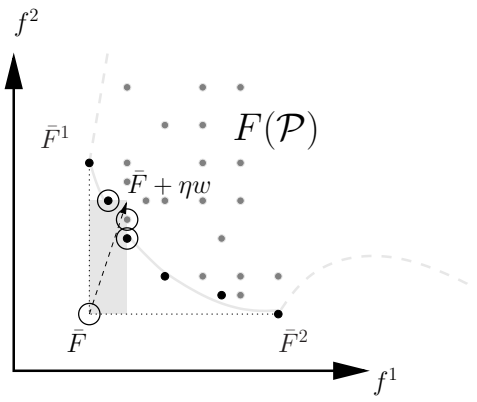


Figure 5.6: The solutions returned may be dominated

ables will have integer values at the optimal solution. Since we completely change the objective function, such assertion does not hold any more. This is illustrated on Figure 5.7. The points represent the fully integer solutions, and the lines the possible solutions when some integrality constraints have been relaxed. It appears that the solver will probably return a solution in which some variable have a fractional value, even it does not make sense "physically".

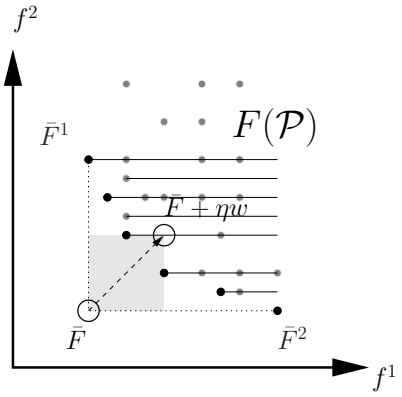


Figure 5.7: The solutions returned may not be integer

5.2.3 ϵ-restricted method

The idea of the ϵ-restricted method is to add additional restrictions preventing the solver to return one of the $\bar{x}^i$ [CH83]. More precisely, an $\epsilon-$restricted method corresponds to generate and solve mono-objective problems under the form:

$$\begin{cases} \min_x f^i \\ x \in \mathcal{P} \\ f^j \leqslant \epsilon^j; j \neq i \end{cases} \tag{5.4}$$

The ϵ^i are chosen such that $\bar{f}^i \leqslant \epsilon^i \leqslant \dot{f}^i$. Figures 5.8 and 5.9 illustrate the key idea of the $\epsilon-$method: minimizing f^1 will give $\bar{x}^1$. If the restriction $f^2(x) \leqslant \epsilon^2$ is added to the problem, minimizing f^1 will not return $\bar{x}^1$ but another points of the Pareto optimal set (see Figure 5.8). The same argument can be applied when minimizing f^2 (see Figure 5.9). The main drawback of this method is that it may generate problems without any feasible solution if $m \geqslant 3$, where m is the number of objective functions.

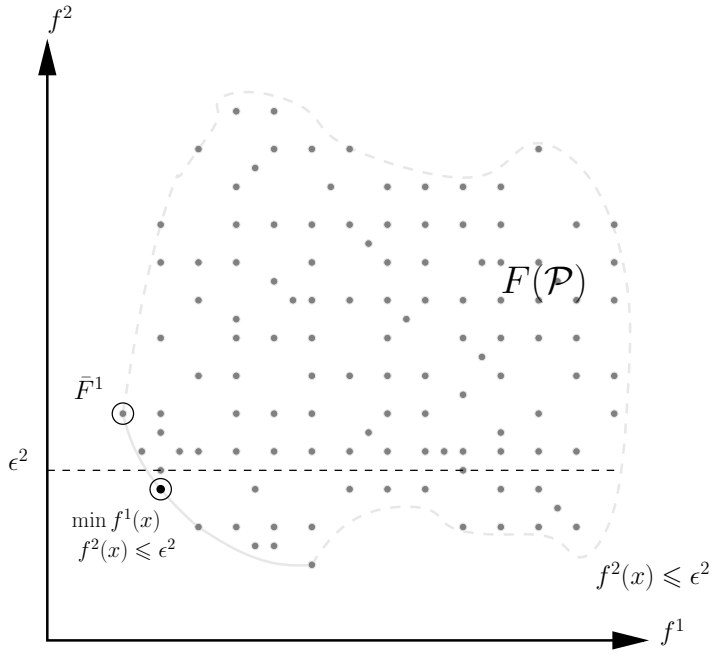


Figure 5.8: ϵ based method minimizing f^1

One may think in adding more restrictive constraints to get a more regular distribution of the Pareto optimal set points, replacing the restrictions :

$$f^j \leqslant \epsilon^j$$

by the restrictions :

$$\epsilon_l^j \leqslant f^j \leqslant \epsilon_u^j$$

where ϵ_l^j and ϵ_u^j are respectively a lower and an upper bound for function f^j, while optimizing with objective function f^i. ϵ_l^j and ϵ_u^j are chosen in a way such that at the end of the algorithm, all possible values for f^j have

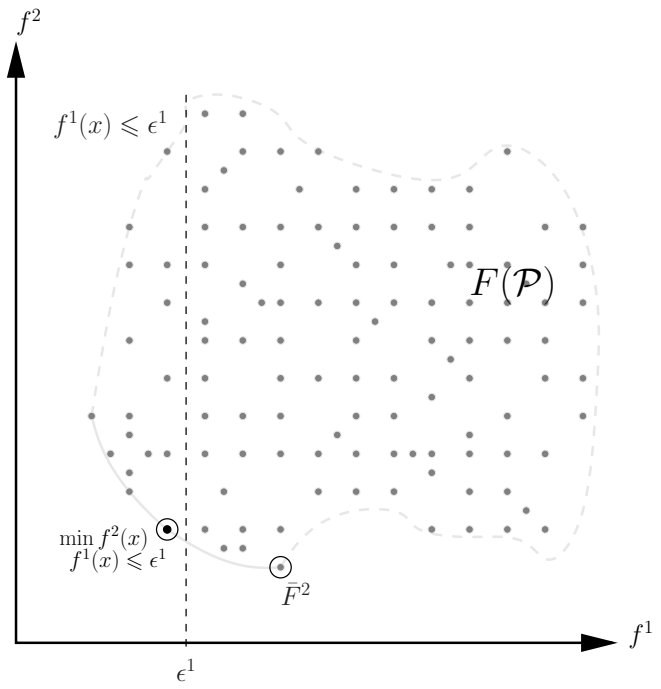


Figure 5.9: ϵ based method minimizing f^2

been covered.

Figure 5.10 illustrates this in the case of a two dimensional problem. However when the dimension $m \geqslant 3$, such algorithm would generate much more empty problems than the original method. As we do not know *a priori* if the problem is empty or not, it may require a high computation time to find it out. Even with non-empty problems, the solver may have great difficulties to find solutions, since the solution set is very restricted. Moreover, points returned by such an algorithm would not necessary belong to the Pareto optimal set if ϵ_l^j and ϵ_u^j define a very narrow solution set. As showed on Figure 5.11, both x_1 and x_2 belongs to the Pareto optimal set, and will be found by the algorithm in other step. But point y, which does not belong to the Pareto optimal set, will also be found when the restriction $\epsilon_l^2 \leqslant f^2 \leqslant \epsilon_u^2$ will be considered. In other words, being too restrictive with respect the ϵ restriction can decrease the method efficiency. In other words, such modifications would not improve the original $\epsilon-$restricted method and consequently will not be further considered.

5.2.4 Chosen method and used algorithm

We described above three different methods to search for the points of the Pareto optimal set. As mentioned in section 5.2.2, the relaxation method is not well adapted for MILP problem such as our, since it may generate

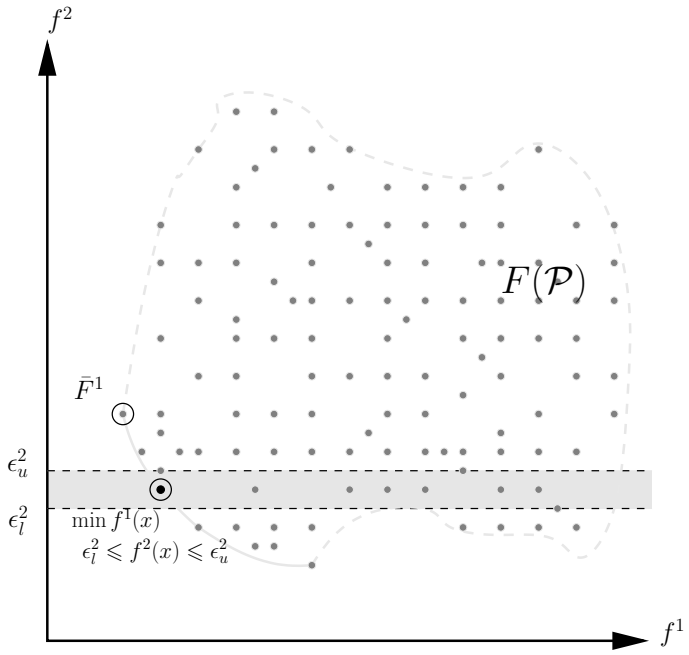


Figure 5.10: More restrictive $\epsilon-$ constraints

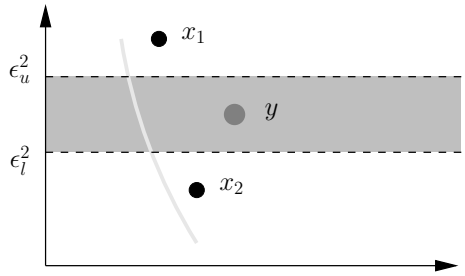


Figure 5.11: Some points may not belong to the Pareto optimal set

dominated and non-integer solutions.

We choose to adapt and use the ϵ-restricted method. The search for the Pareto optimal set will be made by generating various mono-objective optimization problems, whose solutions should belong to the Pareto optimal set. Each generated problem remains a MILP problem.

The algorithm we propose has been designed to generate feasible problems and to use already computed solutions as initial solutions, easing the solving of the problem. The notations used are described below. We also sort the $\epsilon-$restrictions used using a concept of *predecessor* and *successor* defined below.

We denote by $\bar{f}_j^i$ the j^{th} component of $\bar{F}^i$. As we will generate several ϵ-restricted problems, we call ϵ_j^i the j^{th} ϵ restriction associated with metric i.

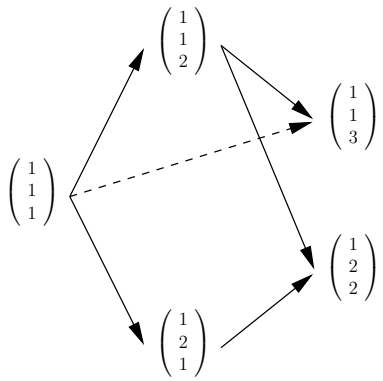


Figure 5.12: Successor/predecessor definition

We say that a vector v^1 is a *successor* of the vector v^0 when $\exists! i / v_i^0 < v_i^1$ and $v_j^0 = v_j^1, \forall i \neq j$. Reciprocally, we say that v^0 is a *predecessor* of v^1. We say that vector v^k is a *close successor* of vector v^0 among a set of vector $v^1 \dots v^j$ if v^k is a successor of v^0 and $\nexists v \neq v^k \in v^1 \dots v^j / v^0 \leqslant v \leqslant v^k$. In this case, v^0 is a close predecessor of v^k. Note that a vector may have up to m close successors or close predecessors. Those definitions are represented on Figure 5.12. A dashed arrow represents the "successor" relationship and a continuous arrow represents the "close successor" relationship. We say that we *sort* a set of vector v when we associate to each vector its close successors and its close predecessors.

Algorithm 6 describes the algorithm we used to search for the Pareto optimal set. Informally speaking, it consists in computing ϵ-vectors, generating and solving ϵ-restricted problems in a way that we can always use as an initial solution one of the solutions recently computed. We do not compute the complexity of the algorithm, since the complexity of generating the $\epsilon-$vectors is negligible with respect to the complexity of solving the optimization problems.

Figure 5.13 illustrates the idea of Algorithm 6 for a problem with two metrics. We only represented the ϵ generated to solve problems optimizing metric f^1. The ϵ_l^2 are represented by lines. The dotted arrows indicate which solution will be used as starting point for each problem generated from an ϵ. The dashed lines give the succession of the Pareto optimal points that will be computed. The Figure also illustrates the fact that if the epsilon are chosen too close one to another, two ϵ-generated problems will give the same point.

A possible problem of our method is the difficulty to choose interesting ϵ-vectors. If the $\epsilon-$vectors are too close one to another, it is likely that we obtain the same solutions. On the other hand, having ϵ-vectors different one from another may result in largely unexplored regions of the Pareto set. As far as we know, the majority of the multiobjective algorithms based on mono-objective optimization suffer with the problem of choosing different

Algorithm 6 $\epsilon-$restricted method based algorithm

Require: An optimization problem $\mathbb{R}^n \to \mathbb{R}^m$

1: Solve each mono-objective problem, to obtain each of the $\bar{F}^k, 1 \leqslant k \leqslant m$.
2: **for** i from 1 to m **do**
3: **for** For each metric j from 1 to $m, j \neq i$ **do**
4: **for** For each metric l from 1 to $m, l \neq i$ **do**
5: Generate a set $E^{i,j,k}$ containing the ϵ_l^j values to be used as ϵ-constraint with respect to the metric k, such that $\bar{f}_k^j \leqslant \epsilon_l^j \leqslant \bar{f}_k^i$
6: **end for**
7: Compute $E^{i,j} = \prod_{1 \leqslant k \leqslant m, i \neq k} E^{i,j,k}$, where $\prod$ corresponds to the Cartesian product
8: **end for**
9: Compute $E^i = \bigcup_{1 \leqslant j \leqslant m, i \neq j} E^{j,l}$
10: Sort E^i and remove duplicates, if any
11: **for** each ϵ vector in E^i **do**
12: build optimization problem (5.4) and solve it using as initial solution the solution obtained solving the problem build with one of ϵ close predecessor. If ϵ does not have close predecessor, it means that we can use one of the $\bar{x}^j$.
13: **end for**
14: **end for**

Ensure: Points of $\mathbb{R}^m$ belonging to the Pareto optimal set

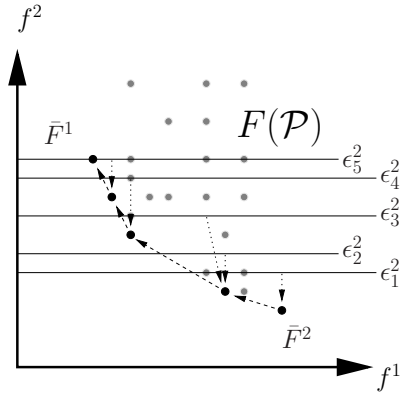


Figure 5.13: ϵ-based algorithm

values for a parameter, in order to generate different solutions. We do not perform an analysis of the number of $\epsilon-$vectors to be generated or of the values to use to generate such vectors.

We also face the absence of guarantee that a solution is the best one according to the metrics that are not the objective function. When we minimize a metric, the solver improves its solutions until finding the optimal one. When it proves that a solution is optimal, it returns it, whatever may be the value of the solution with respect to another metric. Carefully chosen ϵ-vectors provide a workaround to this problem. If the ϵ-vector is sufficiently tight, the solver will not have the possibility to return a solution of low quality in relation with the other metrics than the one being optimized.

Finally remains as open question whether or not our method is able to find any point belonging to the Pareto set. We generate the ϵ-vectors in a way that we are able to provide an initial solution. The generation process of the $\epsilon-$restriction discards at least ϵ-vectors that would lead to empty problem. Remains the question about whether or not it also discards some $\epsilon-$vectors leading to Pareto solutions unreachable with $\epsilon-$vectors generated with our method.

5.3 Evolutionary algorithm

The previous section describes different methods using mathematical model to identify the Pareto optimal set. Those methods have the following approach in common: they solve many mono-objective problems. For each problem solved, we hopefully obtain a nondominated point. However, such approach may require a very high computation time, particularly if solving one problem already requires hours of computation. An other approach to identify the Pareto optimal set is to compute simultaneously various points of it. This is possible using *population* based algorithms, such as *evolutionary algorithms* [SMK00].

5.3.1 Problem solved

In this section, we focus on a subproblem of the reconfiguration problem. We do not consider the overall reconfiguration problem described in section 2, but a multiperiod virtual topology design problem. Solving the routing of the data traffic with evolutionary algorithm would significantly increase the complexity of the problem modeling and the solutions search.

From a physical topology and a succession of traffic matrices, a virtual topology adapted to each traffic matrix is defined. The wavelength allocation is also solved. The difference with the reconfiguration problem as defined in section 2.2 lies in the fact that we do not deal with the routing of the data traffic. The traffic matrices which serve as input of our problem are expressed in number of lightpaths.

The quality of the solutions obtained is evaluated with respect to the amount of resources used (number of optical links), to the balance of the load over the network (maximum link load) and to the number of reconfigurations triggered by the solution (number of reconfiguration).

5.3.2 The Strength Pareto Evolutionary Algorithm

Evolutionary algorithms start with an initial set of solutions and try to improve them by applying *genetic* operations, simulating a general evolution process. The population is a set of solutions called *individuals*. Each individual represents a possible solution for the considered problem and is encoded as a long string called *chromosome*, divided in many *genes* representing unitary information unit.

The population evolves through different genetic operations, such as *mutation*, *crossover* and *selection*. The mutation is the operation introducing new genes in the population. It consists in modifying randomly a gene. The *crossover* operation combines two individuals to generate two other individuals. The main purpose of the crossover is to recombine the existing elements. Through a *fitness* function, a value is assigned to each individual, representative of its quality. The selection consists in choosing the individuals based on their fitness, to create a new population on which the genetic operations will be applied again.

With an adequate fitness function, it is possible to make the overall population map effectively the Pareto optimal set. Such approach may be flexible and computationally efficient. The main drawback of such kind of methods is the lack of guarantees about the solutions quality. It is difficult to know how good is the approximation of the Pareto optimal set obtained at the end of the process.

There are different variants of population algorithms solving multiobjective algorithms [Hor97]. We chose the *Strength Pareto Evolutionary Algorithm* (SPEA) [ZT99]. It maintains a set of nondominated solutions in an external population and uses a fitness function based on the dominance. A mating pool is filled for each generation with individuals from both the current population and the external population. The genetic operations will be applied to the elements of the mating pool. As a consequence, the solutions stored in the external population participate in the selection.

The steps of the SPEA algorithm are described on Algorithm 7 and will be described more precisely from sections 5.3.6 to 5.3.10. We note P the current population and P' the external population. N is the maximum size of P while N' is the maximum size of the external population.

The overall complexity of the algorithm, including the complexity of the sub-procedures, is $O\left(GN\left(mN^2 + |\mathcal{N}|^2 L\right)\right)$, where G is the maximum number of generations and $L = \sum_{1 \leqslant t \leqslant \mathcal{T}} \sum_{(o,d) \in \mathcal{N}^2, o \neq d} \mathcal{D}_{o,d}(t)$ is the overall number of lightpaths defined.

Algorithm 7 The Strength Pareto Evolutionary Algorithm

1: Generate initial population P and create the empty external set of nondominated individuals P'
2: **repeat**
3: Evaluate objective function for each individual in P
4: Copy nondominated members of P to P'
5: Remove solutions of P' which are dominated by any other solution of P'
6: **if** $|P'| \geqslant N'$ **then**
7: prune P' using a clustering algorithm
8: **end if**
9: Compute the fitness function of each individual in $P \cup P'$
10: Select individuals through binary tournament from $P \cup P'$ until the mating pool is filled
11: Apply crossover and mutation to members of the mating pool in order to create a new population P
12: **until** Maximum number of generations G reached

5.3.3 SPEA application to the reconfiguration problem

The SPEA is a metaheuristic that can be applied to any multiobjective optimization problem. In the classical genetic algorithms, the operations of mutation and crossover are generic, the specificity of the problem lying in a binary chromosome encoding and the fitness function. The mutation only consists in changing a "0" into a "1" or reciprocally and the crossover consists in cutting two chromosomes in two at the same position and swapping the end of the chromosomes [Hol75].

Such generic approaches work fine when any possible combination in the chromosome leads to a feasible solution. In the case of constrained problems, finding a chromosome encoding that allows random mutation or crossover is a complicated task. We did not succeed in finding such chromosome encoding for our problem. Consequently the mutation and crossover procedures depends specifically on the problem.

5.3.4 Chromosomes encoding

The choice of the chromosome codification is an important step in the use of evolutionary algorithm. It has to allow the representation of any possible solution for the problem and to allow quite easily to perform the genetic operations.

The codification of the chromosomes we retain is the following: Each chromosome $\mathcal{C}$ is divided in $\mathcal{T}$ parts of the same size, one for each time period. Each of these parts $\mathcal{C}(t)$ represents the virtual topology associated with the corresponding time period t and is divided in $|\mathcal{N}|(|\mathcal{N}|-1)$ segments $\mathcal{C}(t)_{(o,d)}$,

associated with all the possible origin-destination pairs $(o,d)\,, o \neq d$. Each segment $\mathcal{C}(t)_{(o,d)}$ contains $\mathcal{K}$ values or less if so many paths do not exist. The value $\mathcal{C}(t)_{(o,d)}(k)$ represents the number of times path k is defined for origin-destination pair (o,d) during time period t in the solution proposed by this chromosome. Consequently, the size of the overall chromosome is $O\left(\mathcal{T}|\mathcal{N}|^2\mathcal{K}\right)$. Figure 5.14 illustrates this chromosome construction. Note that the encoding chosen is not a binary one.

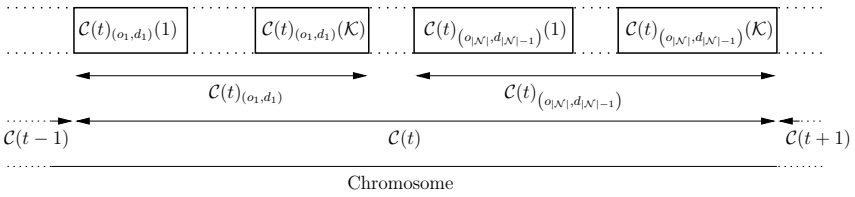


Figure 5.14: Chromosome encoding

As mentioned, we only allow a lightpath to use one of the $\mathcal{K}$-shortest paths between each pair origin-destination. There are various works about the theoretical aspects of the $\mathcal{K}$-shortest path problem, and numerous algorithms exist. In this work we chose to implement the algorithm presented in [MP03], as it is simple and quite efficient.

5.3.5 Wavelength assignment

The chromosome encoding described in section 5.3.4 does not include information about the wavelength allocation. It would result in much longer chromosomes. The genetic operations do not consider the wavelength allocation.

In evolutionary algorithm, the solution space is represented by the number of valid combinations a chromosome can have. Longer chromosomes implies larger solution space. To explore efficiently a larger solution space, we need a larger population. Consequently, longer chromosomes have negative consequences on the overall computation process on two aspects: longer chromosomes means higher memory requirements, and larger population means more computations and also larger memory requirements.

Fore those reasons, we decided not to include the wavelength allocation in the chromosome, but to solve the wavelength allocation problem for each individuals using a fast heuristic, described by Algorithm 8. The algorithm used is a greedy algorithm. The main problem of this algorithm is the following: it may not find any solution in some cases when solutions actually exist. However, we believe that such problem do not happen frequently and the computation time gain over more clever algorithm is worth it.

The complexity of the wavelength allocation algorithm is $O\left(|\mathcal{N}|^2 L\mathcal{W}\right)$.

Algorithm 8 Wavelength allocation algorithm

```
1: for Each node do
2:   for Each time period do
3:     Initialize the list of the available wavelengths
4:     Sort the lightpaths having the current node as origin, from the
       longest to the shortest
5:     for Each lightpath do
6:       repeat
7:         Associate a wavelengths to the lightpath
8:         Check if it is possible to allocate the wavelengths all along the
           path
9:       until A wavelength has been allocated or all available wavelength
         have been experimented
10:      if All wavelengths have been unsuccessfully experimented then
11:        Fail
12:      else
13:        Update the list of available wavelengths
14:      end if
15:    end for
16:  end for
17: end for
```

5.3.6 Mutation

We perform the mutation the following way: We select one by one all individuals. We decide with a probability α if the individual is mutated. If this is the case, we generate a mutation in a randomly chosen gene $\mathcal{C}(t)_{(o,d)}(k)$. The mutation is made the way it is described by Algorithm 9. Basically, we choose randomly a gene, reset its value, and reallocate the lightpaths carried among all the genes of its segment.

We generate a mutation guarantying that the resulting solution is still feasible. This algorithm calls $N + N'$ times the wavelength allocation algorithm, leading to a complexity of $O\left((N + N')\,|\mathcal{N}|^2 L\mathcal{W}\right)$.

5.3.7 Crossover

From two chromosomes from the mating pool we generate two additional chromosomes. The crossover process is done as described by Algorithm 10. This combines the characteristics of two different individuals. Hopefully, this allows to combine the strengths of the two original individuals. We do not replace the two original individuals, as they may be better than the two individuals generated by the crossover operation. As a consequence of the recombination, some genes can appear in the recombined chromosomes.

When the recombination does not preserve the feasibility of the two

Algorithm 9 Mutation algorithm

for Each chromosome in the mating pool **do**
  Randomly decide with a probability α if there will be a mutation
  if the mutation happens **then**
    Chose randomly a gene $\mathcal{C}(t)_{(o,d)}(k)$. Its value is v
    Reset its value: $\mathcal{C}(t)_{(o,d)}(k) = 0$
    for v times **do**
      Chose randomly one of the $\mathcal{K}$ genes in segment $\mathcal{C}(t)_{(o,d)}$ with probability $\frac{1}{\mathcal{K}}$. Let say we chose gene $\mathcal{C}(t)_{(o,d)}(l)$
      Increment of 1 the value of $\mathcal{C}(t)_{(o,d)}(l)$
    end for
    Make wavelength assignment with Algorithm 8
  end if
end for

Algorithm 10 Crossover algorithm

Divide randomly the individuals of the mating pool in two lists of equal number of individuals
repeat
  Take the first individual of each list and remove each one from its list
  Randomly decide with a probability β if there will be crossover
  if The crossover happens **then**
    Copy the two parents individuals
    Chose randomly a gene $\mathcal{C}(t)_{(o,d)}(k)$
    swap from the two chromosomes the value of all genes following $\mathcal{C}(t)_{(o,d)}(k)$
    if the resulting $\sum_{i=1}^{\mathcal{K}} \mathcal{C}(t)_{(o,d)}(i) \neq \mathcal{D}_{o,d}(t)$ **then**
      Also permute the genes of segment $\mathcal{C}(t)_{(o,d)}$ that have not been permuted yet
    end if
    Make wavelength assignment for each new individuals with Algorithm 8
  end if
until One of the two lists is empty

solutions, the place where the chromosomes are "cut" is deported to the beginning of the segment containing the cut, guarantying the feasibility of the two resulting two individuals. The overall complexity of the crossover algorithm is $O\left((N+N')\left(\mathcal{T}|\mathcal{N}|^2\mathcal{K}\right)\right)$.

5.3.8 Clustering

We cannot simply copy each nondominated solution to an external population and keep it. It could lead to an overpopulated external population. Moreover it is likely that it would contain many very similar solutions. On the other hand we aim to have an external population as representative as possible of the diversity of the solutions contained in the Pareto optimal set. As some external population individuals also participate to the genetic operations, maintaining the diversity of the external population, they help to maintain the diversity of the overall population.

Such objective are achieved by means of clustering. It allows us to regroup very similar solutions, at the same time reducing the population size and preserving its diversity. We apply the clustering algorithm described in [ZT99] (originally defined in [Mor80]), described by Algorithm 11, where $|C|$ corresponds to the number of elements of C and $||i_1 - i_2|| = \sqrt{\sum_{l=1}^{m}\left(i_1^l - i_2^l\right)^2}$, with m the dimension of the solution space and i_m^l the value of the l^{th} objective of individual i_m.

Algorithm 11 Clustering algorithm

1: Create a cluster c_i for each external nondominated point $i \in P' : c_i = \{i\}$
2: Initialize a set of clusters $C = \bigcup_{i \in P'} \{c_i\}$
3: **while** $|C| \geqslant N'$ **do**
4: Calculate the distance of all possible pair of clusters. The distance d of two clusters c_1 and c_2 is the average distance between pairs of individuals across the two clusters:

$$d = \frac{1}{|c_1|.|c_2|} \sum_{i_1 \in c_1, i_2 \in c_2} ||i_1 - i_2||$$

5: Merge the two clusters c_1 and c_2 that have the minimum distance: $C = C \setminus \{c_1, c_2\} \cup \{c_1 \cup c_2\}$
6: **end while**
7: Compute the reduced nondominated set by selecting a representative individual per cluster

The complexity of the clustering algorithm is $O\left(N'^3 m\right)$.

5.3.9 Fitness

As mentioned in section 5.1.1, when we deal with multiobjective optimization, the objective function is a vector. Moreover, there is no direct mapping between the objective function and the fitness function, as in classical mono-objective optimization. Consequently it can be difficult to evaluate and compare the individuals.

In [ZT99] is proposed along with the SPEA algorithm a fitness function for multiobjective problems. The fitness function is based on the dominance concept. It attracts individuals near the Pareto optimal front and tends to distribute them at the same time along the trade-off surface. Generally the method guarantying the diversity of the population - also called niching method - is distance based. This is not the case here, since the diversity is guaranteed by the fitness function.

As mentioned in section 5.3.5, the wavelength allocation algorithm chosen may not be able to find a solution in some cases. When this happens, we discard the individual.

The fitness assignment is a two-stage process described by Algorithm 12. The fitness is to be minimized. It first computes the fitness of the individuals of the external population, based on the number of elements of the population it dominates. We then sum for each individual of the population P the fitness of the individuals that dominate it. Such fitness function favors individuals near the Pareto-set and distributed along the trade-off surface.

Algorithm 12 Fitness assignment

1: **for** Each chromosome $i \in P'$ **do**
2: Let d be the number of chromosomes in P dominated by i
3: Assign $s_i = \frac{d}{N+1}$
4: **if** Wavelength allocation is possible for this individual **then**
5: Assign $f_i = s_i$
6: **else**
7: Discard individual
8: **end if**
9: **end for**
10: **for** Each chromosome $j \in P$ **do**
11: **if** Wavelength allocation is possible for this individual **then**
12: $f_j = 1 + \sum_{i \in P', j \prec i} s_i$
13: **else**
14: Discard the individual
15: **end if**
16: **end for**

The complexity of the fitness assignment algorithm is $O(NN')$.

5.3.10 Initial population

The initial population is generated randomly. Each chromosome is created in the following way: For each pair origin-destination (o, d) and for each time period t, we choose randomly $\mathcal{D}_{o,d}(t)$ times with a probability $\frac{1}{\mathcal{K}}$ one of the $\mathcal{K}$ possible paths from o to d and initialize the $\mathcal{C}(t)_{(o,d)}(k)$ according to it. This way, each chromosome is a randomly generated feasible solution for the problem. The solutions are possible since each demand is distributed among the different possible paths from each origin to each destination. The complexity of the generation of the initial population is $O(NL)$.

5.4 Conclusion

In this chapter, we focus on the multiobjective aspects of the reconfiguration problem. We present an algorithm based on the $\epsilon-$restricted method in order to generate a set of optimization problems, allowing us to obtain points belonging to the Pareto set. Our algorithm generates only feasible problems and uses already computed solutions as initial solutions.

We also adapt the Strength Pareto Evolutionary Algorithm for a subproblem of the reconfiguration problem. This population based algorithm compute simultaneously various non-dominated points, hopefully belonging to the Pareto set, in a single run. However, the quality of the solutions obtained is not guaranteed.

Chapter 6

Multiobjective optimization: computational results

In the previous chapter, we introduce different methods to find out the Pareto set of the reconfiguration problem. In this chapter, we perform experiments in order to search for the Pareto set of different instances of our problem.

We use the simulated annealing heuristic and a weighted objective method as a first approach of multiobjective optimization. We then use the algorithm based on the use of ϵ-vectors we proposed in section 5.2.4. We run a set of experiments with two different networks, allowing us to generate many nondominated points. We conclude this chapter running a large number of experiments with the SPEA algorithm.

6.1 Weighted method with the simulated annealing algorithm

We superficially apply the weighted objective method described in section 5.2.1 with the simulated annealing algorithm presented in section 3.3.2 and solving the mathematical model presented in chapter 3. We use the same instances and the same traffic matrix as in section 4.5. We consider the small network SN2, the Cost239, NSFNET, N20, N30 and N40 networks. The considered parameters for these experiments are given in Table 6.1. Moreover, we use the same parameters for the simulated annealing algorithm as in section 4.5. The temperature mutation parameter α is 0.005 and the edge weight mutation parameter γ is between 0.5 and 1.

We consider α_O (respectively α_L and α_C) the weight associated with the O (respectively L and C) metric. Depending on the value of each parameter, more importance is given to one or another aspect. The objective function being optimized is given by equation (6.1).

Table 6.1: Parameters to apply the weighted objective method

	SN2	Cost239	NSFNET	N20	N30	N40
$\mathcal{F}_{(i,j)}$	5	5	5	5	5	5
$\mathcal{W}$	8 /16	8/ 16	8 /16	8/ 16	8/ 16	8 / 16
$\mathcal{B}$	40 /20	40/ 20	40 /20	40/ 20	40/ 20	40 / 20
traffic	*var*	*var*	*var*	*var*	*var*	*var*
T_L	36000	36000	36000	86000	86000	86000

$$F = \alpha_O O + \alpha_L L + \alpha_C C \tag{6.1}$$

We study some combinations for the $(\alpha_O, \alpha_L, \alpha_C)$ triple. Note that setting $(\alpha_O = 1, \alpha_L = 0, \alpha_C = 0)$ is equivalent as optimizing the metric O. Similarly, $(\alpha_O = 0, \alpha_L = 1, \alpha_C = 0)$ corresponds to minimizing L, and minimizing C corresponds to $(\alpha_O = 0, \alpha_L = 0, \alpha_C = 1)$. In these cases, we use the results obtained in section 4.5. Tables C.1 to C.4, located in appendix C, give the results obtained with the solver and the simulated annealing algorithm. With the instances N30 and N40, we could not obtain any result with the solver. We report in Table C.5 the results obtained with the simulated annealing algorithm only.

Mixing the resources and reconfiguration objectives turns the problem much more difficult to solve than considering a single objective. This can be observed for the MILP and for the simulated annealing approaches. If we consider only one metric, the results obtained with the other metrics are generally very bad. On the other hand, if we consider the performance-reconfiguration trade-off, that is using at the same time two metrics, we obtain good solutions in relation with both metrics.

For instance, with the SN2 network ($\mathcal{W} = 16$), minimizing objective function O gives a solution using 603 optical links, but triggering 885 reconfigurations. On the other hand, the solution obtained minimizing C uses 1900 optical links and triggers 0 reconfigurations. Mixing the objectives with an equal weight, we obtain a solution using 611 optical links and triggering only 11 reconfigurations. A little flexibility with respect with a metric allows to drastically improve the quality of the solution with respect to other metrics.

6.2 Results with the ϵ-restricted method

We apply the method presented in section 5.2.4 to search for the Pareto set of the reconfiguration problem. Due to the high computational cost, we only apply this method to two instances. We choose the SN2 and cost239 networks, with the parameters given in Table 6.2.

We apply Algorithm 6. We first solve the five classical mono-objective problems without any modifications. We obtain the solutions given in Ta-

Table 6.2: Parameters to apply the ϵ-restricted method

	SN2	Cost239
$\mathcal{F}_{(i,j)}$	5	5
$\mathcal{W}$	8	8
$\mathcal{B}$	40	40
traffic	*var*	*var*
T_L	3600	7200

ble C.6, located in appendix C. Each cell of the table corresponds to a solution and the value of the objective vector. Each value of this vector corresponds to the quality of the solution with respect to the metrics given at the extreme left of the table. For instance, one of the solutions obtains the following objective vector:

$$\begin{pmatrix} O \\ L \\ M \\ H \\ C \end{pmatrix} = \begin{pmatrix} 312 \\ 292 \\ 23 \\ 6.70742 \\ 452 \end{pmatrix}$$

From these solutions, we defined 48 ϵ-vectors for the SN2 network, and 40 for the Cost239 network. We believe that such number of vectors is enough to have a correct mapping of the Pareto set without being excessively time-consuming. We solve with a two hours time-limit each of the generated optimization problems, using a previously computed solution as described by Algorithm 6.

Except for the original five mono-objective problems, for each problem an initial solution is provided. However, it may happen that the solver returns as final solution the initial solution. This happens if the solver hits the time-limit before finding a better solution. As our mathematical model is a MILP model, the Pareto set is not continuous. Consequently, there is no guarantee that each different ϵ-vector leads to a different Pareto point.

For the SN2 network, we solve 53 problems, the 5 mono-objective ones and the 48 $\epsilon-$problems. We obtained 38 different solutions. From those, 23 are nondominated. For the Cost239 network, we solve 45 problems, the 5 mono-objective ones and the 40 $\epsilon-$problems. We obtained 24 different solutions, from those, 12 are nondominated. The nondominated solutions for the SN2 (respectively Cost239) network are given in Table C.7 (respectively Table C.8) located in the appendix.

We obtain dominated solutions. This is consequence of solving the problem with a time-limit. When it is reached, the solver returns the best solution found so far, which may not be optimal. In the case of our $\epsilon-$restricted method, this means that the solution found may not belong to the Pareto set and may be dominated.

By design, our $\epsilon-$algorithm only generates feasible optimization problems, even if finding the optimal solution remains a difficult task. This solves one of the main problems of the original ϵ-method. The main problem faced is related with the choice of the number of $\epsilon-$vectors, and of the vectors themselves. The experiment we run, we use "manually" chosen ϵ-vectors, according to the values obtained solving the mono-objective problems.

For both instances, our algorithm is able to return solutions making controlled use of resources and having low number of reconfigurations. The SN2 instance admits a solution with $O = 485$ and triggering only 30 reconfigurations. The Cost239 network admits a solution with $O = 1185$ and triggering no reconfiguration. However, it suffers from a very high value for $H = 7.04$. Such solutions are interesting from a designer point of view, since making good use of the resources while avoiding large reconfiguration steps, but may be a problem in terms of quality of service.

6.3 The SPEA algorithm

We make experiments in order to evaluate the performance of the evolutionary algorithm described in section 5.3 and its ability to identify the Pareto set. The results obtained allow us to study the relationship between the different metrics chosen.

6.3.1 Computational results

We use the SN1, SN2, Cost239 and NSFNET networks described in section 4.1. We implement the algorithms presented in section 5.3 in Java and run our test instances on a desktop PC. The traffics are randomly generated. We consider 5 time periods, and the demand from a node o to a node d during time period t is randomly defined such that $\mathcal{D}_{o,d}(t) \in [2, 10]$. The metrics considered to compare the quality of our results are the ones described in section 5.3.1, that is the number of optical links, the maximum link load and the number of reconfigurations.

The chosen parameters for the experiments are given in Table 6.3. Choosing the value for the parameters to be used is always a difficult question when dealing with probabilistic algorithms. We make various experiments modifying the values of the different parameters, and it appears to us that this set of parameters is quite good. In relation with other sets of parameters, the number of generations required for the algorithm to converge as well as the computation time are low. Figures 6.1 to 6.4 represent the evolution of the solutions found depending on the generation for instances SN1, SN2, Cost239 and NSFNET.

The computation time goes from 5 minutes for the smallest instance to 16 hours for the largest instances with the largest population. It appears that with this problem and our test instances, the clustering is not really

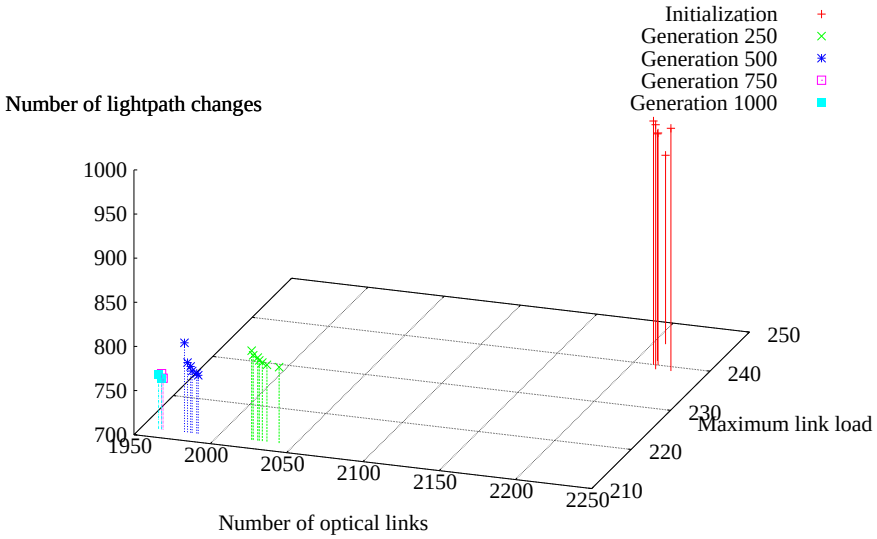


Figure 6.1: SN1 instance: obtained results

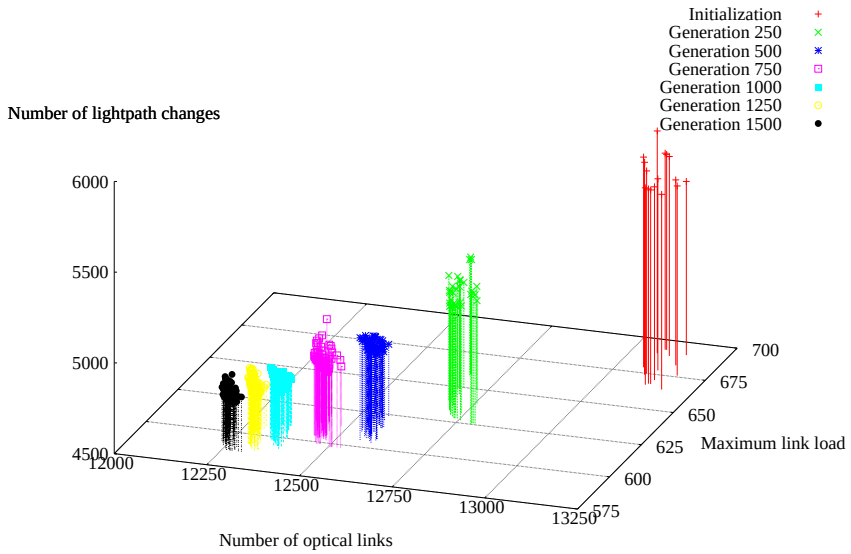


Figure 6.2: NSFNET instance: obtained results

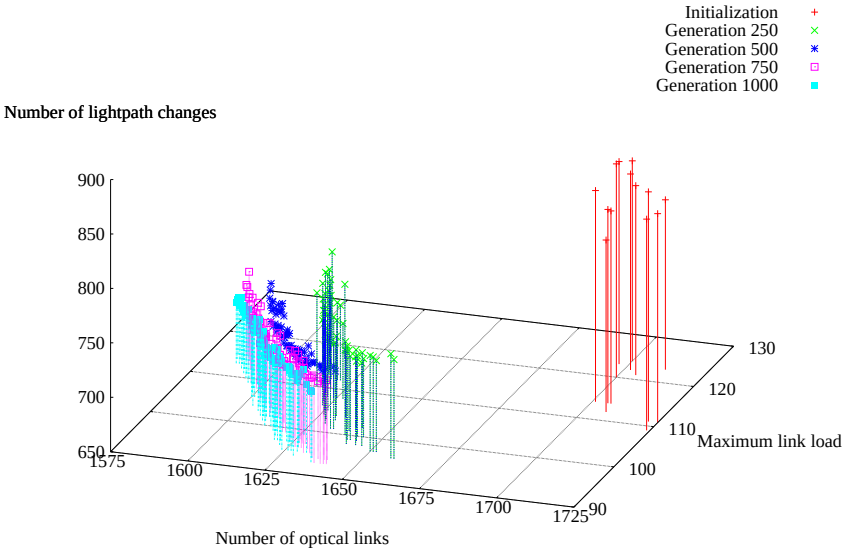


Figure 6.3: SN2 instance: obtained results

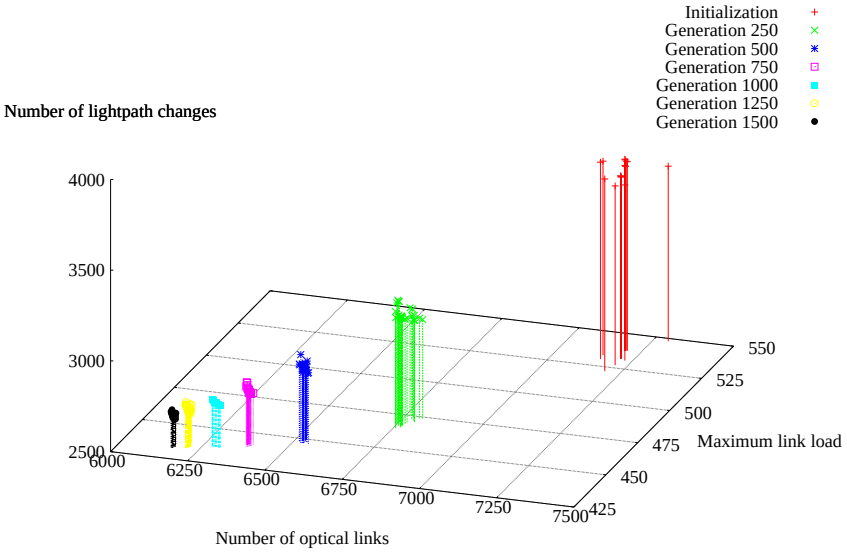


Figure 6.4: Cost239 instance: obtained results

Table 6.3: Parameters to apply the SPEA algorithm

	SN1	**SN2**	**Cost239**	**NSFNET**
$\mathcal{W}$	8	8	24	32
N	50	50	50	50
N'	70	70	70	70
Mating pool size	30	30	30	30
α	5%	5%	5%	5%
β	75%	75%	75%	75%
$\mathcal{K}$	4	4	4	4
Number of generations	1000	1000	1500	1500

necessary, as the size of P' remains small (less than 30 elements) during the execution of the algorithm. The only instance for which the clustering is required is the NSFNET instance.

The SPEA algorithm succeeds in improving significantly the quality of the solutions between the first and the last iteration. None of the chromosomes of the initial population belongs to P' after 250 generations. Similarly, almost none of the chromosomes of P' after 250 generations belongs to P' after 500 generations. The population converges toward the Pareto set globally and progressively. The SPEA algorithm succeeds in maintaining a diversity among the population even though the solution quality constantly improves. The improvements are made in every directions towards the Pareto set, and not in a single direction. It seems that with our parameters 1000 iterations are enough for the algorithm to converge with the small networks (SN1 and SN2). However, the larger networks, more than 1500 generations are required.

Figure 6.5 shows some of the results we obtained with experiments in order to evaluate the influence of number of paths considered $\mathcal{K}$. For $\mathcal{K} = 3$, the algorithm tends to find low quality solutions. Increasing the value to $\mathcal{K} = 4$ significantly improves the quality of the solutions found. When we consider larger values for $\mathcal{K}$ we obtain solutions of lower quality. The size of the solution space is related with the value of the $\mathcal{K}$ parameter and with the best solutions achievable. Increasing $\mathcal{K}$ increases the size of the solution space and the size of the chromosomes, resulting in better solutions, that are harder to find.

The variation of the metrics among the obtained solutions is quite low (less than 5%) with our test instances. However we believe that such variation already have a significant impact in the network use.

The SN1 network is small and quite sparse. There are few different paths from a node to other one. Consequently the solution space is small and there are few nondominated results. This appears clearly on the results obtained. The final population contains very few elements. This is not the case for the other networks having a much larger solution space.

Figures 6.6 (respectively 6.7) represents solutions for the NSFNET net-

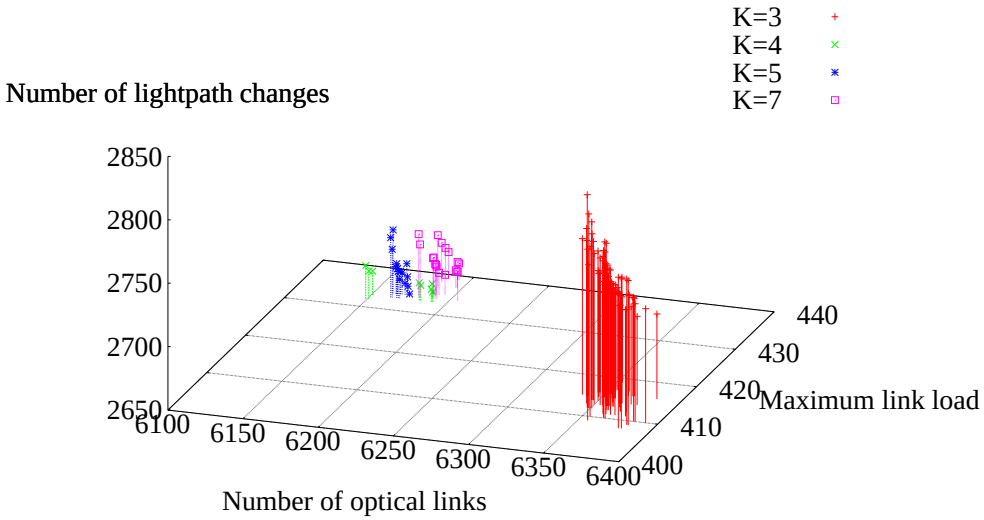


Figure 6.5: Cost239 instance: Influence of the $\mathcal{K}$ parameter

work projected on the plan defined by the metric "number of optical links" (respectively "maximum link load") and "number of lightpath changes". We call those projections "Projection 1" (respectively "Projection 2"). The "Projection 3" is given by the Number of optical links and the maximum link load, and is represented on Figure 6.8.

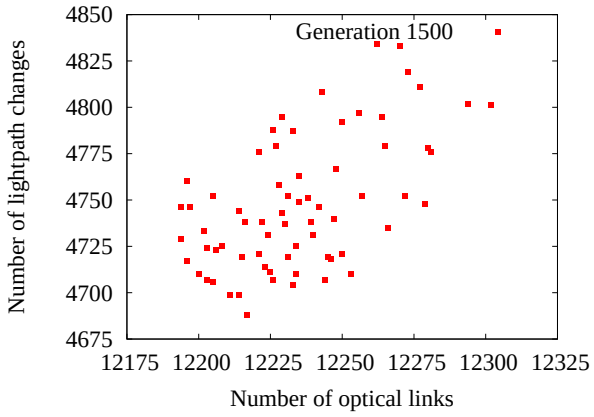


Figure 6.6: NSFNET instance: Projection 1

There is a small trade-off between the use of the resource and the number of reconfigurations to be carried out. But another aspect to be considered is that using long lightpaths, and consequently more optical links, results in

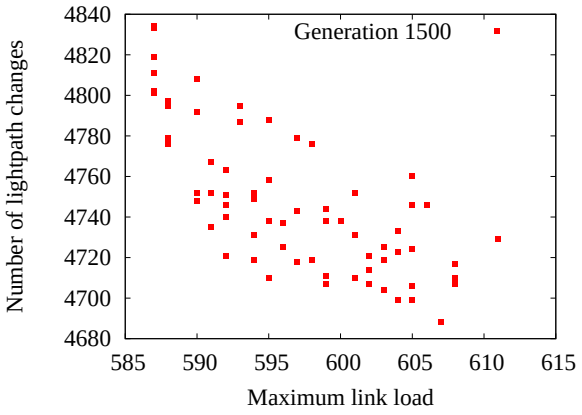


Figure 6.7: NSFNET instance: Projection 2

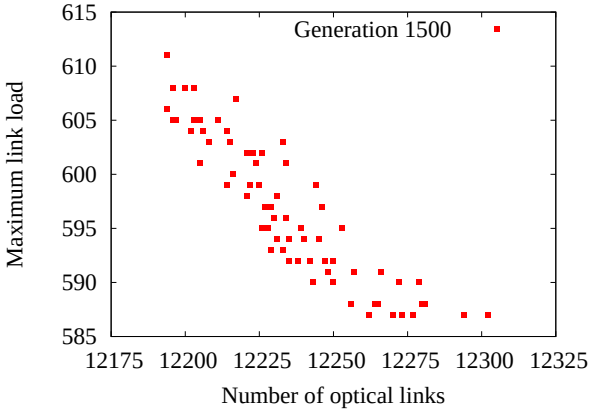


Figure 6.8: NSFNET instance: Projection 3

a high number of reconfigurations. This can be observed on figure 6.6. The solutions using the higher amount of optical links are the ones requiring the higher number of reconfigurations.

The trade-off between the number of reconfigurations and the maximum link load is apparent on figure 6.7: Decreasing the number of reconfigurations results in an increasing maximum link load.

The solutions on projection 1 and 2 are much less organized that on Projection 3. It appears that the relationship between the maximum link load and the number of optical links used is tight: for a given value of optical links used, there are few variations on the maximum link load. Moreover, the trade-off between the two involved metric clearly appears: Using a large number of optical links allows to use longer lightpaths, and consequently to decrease the maximum link load.

6.4 Conclusion

We apply three different methods to perform a multiobjective study of the reconfiguration problem. We apply superficially a weighted objective method with the mathematical model and the simulated annealing heuristics. It allows us to find out some interesting trade-offs, and the computation time is low when used with the simulated annealing algorithm.

We also make experiments with an ϵ-based method. Our method solves one of the possible problems of ϵ-based methods. By design, our algorithm guarantees the existence of a feasible solution to the problem. However, this method is more complex to handle than the weighted objective method, and requires a high computation time.

Finally, we apply the Strength Pareto Evolutionary Algorithm to a subproblem of the reconfiguration problem. The algorithm generates a set of various non-dominated solutions in a run. It appears to be able to maintain a good diversity among the population. However, it suffers from the lack of evaluation of the solution quality.

Carrying out a multiobjective analysis allows to obtain solutions presenting interesting trade-offs. For instance there are solutions using few resources and triggering few reconfigurations. But those solutions suffer from bad quality with respect to the average number of hops or to the maximum link load.

Performing a multiobjective study of a problem as complex as the reconfiguration problem brings valuable information about the different possible solutions and their relative performance. However, it may be difficult for a decision maker to find out what is the best solution according to its own criteria. Choosing a solution among tenth of different possibilities is a delicate task. It would be particularly interesting to study and define procedures to help the decision making.

Chapter 7

Conclusions and perspectives

7.1 Conclusion

In this thesis, we address the reconfiguration problem in multifiber WDM networks. This problem comes from the evolution of the traffic carried by the network during its lifetime. It is a complex problem involving various parameters.

We propose for this problem a mixed integer linear programming model, based on a source formulation. The model is more compact than other models found in the literature. It has been published in the literature [HM05a]. Some technological extensions and cuts are adapted from similar problems, or even introduced. We also propose a lower bound for the problem. However, as the problem is difficult to solve exactly, we propose a greedy and a simulated annealing heuristics. Technical report presenting the simulated annealing have been written [HD05].

We make experiments in order to evaluate the performance of our model with respect to different metrics. We also study the influence of the cuts and extensions on the difficulty to solve the problem and on the solution characteristics. Technical report presenting this work have been written [HM05c]. We also run a set of experiments in order to evaluate the quality of the proposed algorithm. For a very low computation time, the greedy algorithm finds a solution of decent quality. The simulated annealing represents an interesting trade-off between the computation time and the solution quality, while the mathematical model finds the best solutions, but may difficulties converge or even to handle large instances.

As the mono-objective approach is not sufficient to capture the different trade-offs involved by this problem, we perform a multiobjective study of the problem. We propose an algorithm based on the ϵ-restricted method. Our algorithm does not have one of the main drawbacks of classical ϵ-restricted method, since the problems generated always admit a solution. Preliminary works about this methods have been published in an international confer-

ence [HM05b]. We also adapt the Strength Pareto Evolutionary Algorithm to a subproblem of the reconfiguration problem.

We perform different experiments allowing us to validate those algorithms and to extract various solutions covering different aspects of the involved trade-off.

7.2 Perspective and future works

Part of this work is based on the hypothesis that the future traffic evolutions are known, or at least that the differences between the foreseen traffic evolution and the effective future traffic evolution are negligible. This hypothesis is very strong, particularly if we want to consider long term traffic evolution. It would be particularly valuable to make a sensibility analysis of the solutions obtained with different methods, and to include some elements of the adaptation algorithms. This would allow us to compute initially a solution for the overall reconfiguration problem considering estimated traffic matrices, but also to adapt the solution found to traffic evolution that has not been foreseen.

By minimizing the number of reconfigurations, we facilitate the implementation of each new virtual topology. However, we left aside the way to carry out this implementation. The way the information regarding the new virtual topology and routing is transmitted also has to be carefully studied. The cost of carrying out a reconfiguration also has to be taken into account.

With respect to the multiobjective analysis, it would be particularly interesting to study and define procedures to help the decision making. The proposed algorithms aim to identify the Pareto frontier of the problem, where are located the interesting trade-offs. However, it can contains a high number of different solutions and it can be confusing for the decision maker to extract one and only one solution from this set of interesting trade-offs.

Bibliography

[Ban96] D. Banerjee. *Design and analysis of wavelength-routed optical networks*. PhD thesis, University of California, Davis, 1996.

[BDH+99] P. Batchelor, B. Daino, P. Heinzmann, C. Weinert, J. Späth, B. Van Caenegem, D.R. Hjelme, R. Inkret, H.A. Jäger, M. Joindot, A. Kuchar, E. Le Coquil, P. Leuthold, G. de Marchis, F. Materaand, B. Mikac, H.-P. Nolting, F. Tillerot, and N. Wauters. Ultra high capacity optical transmission networks : Final report of action cost 239. Technical Report ISBN 953-184-013-X, Faculty of Electrical Engineering and Computing, HR, Zagreb, 1999.

[BM00] D. Banerjee and B. Mukherjee. Wavelength-routed optical networks: Linear formulation, resource budgeting tradeoffs, and a reconfiguration study. *IEEE/ACM Transactions on Networking*, 8(5):598–607, 2000.

[BR99] I. Baldine and G. Rouskas. Reconfiguration and dynamic load balancing in broadcast WDM networks. *Photonic Network Communications Journal*, 1(1):49–64, June 1999.

[BR01] I. Baldine and G.N. Rouskas. Traffic adaptive WDM networks: A study of reconfiguration issues. *Journal of Lightwave Technology*, 19(4):433–455, April 2001.

[CGK92] I. Chlamtac, A. Ganz, and G. Karmi. Lightpath communications: An approach to high-bandwidth optical WAN's. *lEEE Transactions on Communications*, 40(7):1171–1182, July 1992.

[CH83] V. Chankong and Y.Y. Haimes. *Multiobjective decision making: Theory and methodology*. Elsevier, New York, North-Holland edition, 1983.

[DR00] R. Dutta and G.N. Rouskas. A survey of virtual topology design algorithms for wavelength routed optical networks. *Optical Networks Magazine*, 1(1):73–89, January 2000.

[DSS03] P. Datta, M. Sridharan, and A.K. Somani. A simulated annealing approach for topology planning and evolution of mesh-restorable optical networks. In *8th IFIP Working conference on optical networks design and modeling (ONDM)*, February 2003.

[DW60] G.B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operation Research*, 8:101–111, 1960.

[EB80] Inc Encyclopædia Britannica, editor. *Encyclopædia Britannica*, volume 18, chapter Telephony and telecommunication systems, pages 82–96. Encyclopædia Britannica, Inc, 1980.

[EG00] M. Ehrgott and X. Gandibleux. A survey and annotated bibliography of multiobjective combinatorial optimization. *OR Spektrum*, 2000.

[FCI+02] Y. Frignac, G. Charlet, W. Idler, R. Dischler, P. Tran, S. Lanne, S. Borne, C. Martinelli, G. Veith, A. Jourdan, J.-P. Hamaide, and S. Bigo. Transmission of 256 wavelength-division and polarization-division-multiplexed channels at 42.7gb/s (10.2tb/s capacity) over 3x100km of teralight fiber. In *Optical Fiber Communication Conference and Exhibit (OFC)*, pages FC5–1 – FC5–3, 2002. DOI: 10.1109/OFC.2002.1036776.

[FPR+03] A. Ferreira, S. Pérennes, H. Rivano, A.W. Richa, and N. Stier Moses3. Models, complexity and algorithms for the design of multi-fiber WDM networks. *Telecommunication Systems*, 24(2-4):123–138, October 2003. DOI: 10.1023/A:1026158611840.

[GADO01] N. Geary, A. Antonopoulos, E. Drakopoulos, and J. O'Reilly. Analysis of optimization issues in multi-period DWDM network planning. In *INFOCOM*, pages 152–158. IEEE, 2001.

[GM02] A.E. Gençata and B. Mukherjee. Virtual-topology adaptation for WDM mesh networks under dynamic traffic. In *INFOCOM*, volume 1, pages 48–56. IEEE, June 2002.

[GPA+02] N. Geary, N. Parnis, A. Antonopoulos, E. Drakopoulos, and J. O'Reilly. The benefits of reconfiguration in optical networks. In *10th International Telecommunication Network Strategy and Planning Symposium (Networks)*, pages 373–378, June 2002.

[HD04] S. Huang and R. Dutta. Research problems in dynamic traffic grooming in optical network. Traffic grooming workshop, co-located with IEEE/Create-Net BROADNETS 2004, October 2004.

[HD05] G. Huiban and P. Datta. Virtual topology reconfiguration issues in evolution of WDM optical networks. Research report 5711, INRIA, 2005.

[HM05a] G. Huiban and G. R. Mateus. A MILP model for the reconfiguration problem in multi-fiber WDM networks. In *SBRC Simpósio Brasileiro de Redes de Computadores*, May 2005.

[HM05b] G. Huiban and G. R. Mateus. A multiobjective approach of the virtual topology design and routing problem in WDM networks. In IEEE, editor, *ICT International Conference on Telecommunications*. IEEE, May 2005.

[HM05c] G. Huiban and G.R. Mateus. Optimization aspects of the reconfiguration problem in WDM networks. Research report 5730, INRIA, 2005.

[Hol75] J.H. Holland. *Adaptation in natural and artificial systems*. University of Michigan Press, December 1975. ASIN: 0472084607.

[Hor97] J. Horn. *Handbook of evolutionary computation*, volume 1, chapter Multicriterion decision making, pages F1.9:1–F1.9:15. IOP Publishing and Oxford University Press, 1997. Thomas Bäck, David Fogel and Zbigniew Michalewicz editors.

[JLFP92] K.L. Jones, I.J. Lustig, J.M. Farvolden, and W.B. Powell. Multicommodity networks flows: The impact of formulation on decomposition. Technical Report SOR-91-23, Department of Civil Engineering and Operations Research, Princeton University, April 1992.

[JMT04] B. Jaumard, C. Meyer, and B. Thiongane. ILP formulations for the RWA problem-symmetric systems. In *Globecom*, volume 3, pages 1918–1924. IEEE, November 2004. DOI: 10.1109/GLOCOM.2004.1378328.

[JPM03] Z.K.G. Patrocínio Jr., P.P.R.Teixeira, and G.R. Mateus. Traffic grooming and reconfiguration for incremental traffic in WDM optical networks. In *International Network Optimization Conference (INOC)*, pages 454–459. Informs, October 2003.

[KJV83] S. Kirkpatrick, C.D. Gelatt Jr., and M.P. Vecchi. Optimization by simulated annealing. *Science*, 1983.

[KO00] M. Kato and Y. Oie. Reconfiguration algorithms based on metaheuristics for multihop WDM lightwave networks. In *IEEE International Conference on Communications (ICC)*, pages 1638–1644, June 2000.

[LA91] JF.P. Labourdette and A.S. Acampora. Logically rearrangeable multihop lightwave networks. *IEEE Transactions on Communications*, 39(8):1223–1230, august 1991.

[LHA94] JF.P. Labourdette, G.W. Hart, and A.S. Acampora. Branch-exchange sequences for reconfiguration of lightwave networks. *IEEE transactions on communications*, 42(10):2822–2832, October 1994.

[LS01] G. Li and R. Simha. On the wavelength assignment problem in multifiber WDM star and ring networks. *IEEE/ACM Transactions on Networking*, 9(1):60–68, February 2001.

[MMHE93] G. Marsden, P. Marchand, P. Harvey, and S. Esener. Optical transpose interconnection system architectures. *OSA Optics letters*, 18(13):1083–1085, July 1993.

[Mor80] J.N. Morse. Reducing the size of the nondominated set: Pruning by clustering. *Computers and Operations Research*, 7(1-2):55–60, 1980.

[MP03] R. Mewanou and S. Pierre. Dynamic routing algorithms in all-optical networks. In *Canadian Conference on Electrical and Computer Engineering (CCECE)*, volume 2, pages 772–776. IEEE, May 2003. DOI: 10.1109/CCECE.2003.1226009.

[MS01] P.P. Mitra and J. B. Starke. Nonlinear limits to the information capacity of optical fiber communications. *Nature*, 41:1027–1030, June 2001.

[NTM00] A. Narula-Tam and E. Modiano. Dynamic load balancing in WDM packet networks with and without wavelength constraints. *IEEE Journal on Selected Areas in Communications*, 18(10):1972–1979, 2000.

[otEC03] Commission of the European Communities. Report on the implementation of the eu electronic communication regulatory package. Technical report, European Union, 2003.

[Par96] V. Pareto. *Cours d'économie politique*. F. Rouge, Lausanne, 1896.

[RGK+02] M. Roughan, A. Greenberg, C. Kalmanek, M. Rumsewicz, J. Yates, and Y. Zhang. Experience in measuring backbone traffic variability: models, metrics, measurements and meaning. In *ACM SIGCOMM Internet Measurement Workshop (IMW'2002)*, November 2002. (extended abstract).

[Roc98] R.T. Rockafellar. *Network flows and monotropic optimization*. Athena scientific, 1998.

[Rou01] G.N. Rouskas. *Wiley Encyclopedia of Telecommunications*, chapter Routing and Wavelength Assignment in Optical WDM Networks. John Wiley & Sons, 2001.

[RR00] B. Ramamurthy and A. Ramakrishnan. Virtual topology reconfiguration of wavelength-routed optical WDM networks. In *Globecom*, volume 2, pages 1269–1275. IEEE, November 2000.

[RS98] R. Ramaswami and K.N. Sivarajan. *Optical networks. A practical perspective*. Academic press / Morgan Kaufmann Publishers, 1998.

[SMGM01] N. Sreenath, C.S.R. Murthy, B.H. Gurucharan, and G. Mohan. A two-stage approach for virtual topology reconfiguration of WDM optical networks. *Optical Networks Magazine*, may/june 2001.

[SMK00] F. Sbalzarini, S. D. Mueller, and P. Koumoutsakos. Multiobjective optimization using evolutionary algorithms. In *CTR summer program*. Center for Turbulence Research, Stanford University, 2000.

[TMP02] M. Tornatore, G. Maier, and A. Pattavina. WDM network optimization by ILP based on source formulation. In *INFOCOM*, volume 3, pages 1813–1821. IEEE, June 2002. DOI 10.1109/INFCOM.2002.1019435.

[TPF97] R.H.C. Takahashi, P.L.D. Peres, and P.A.V. Ferreira. Multiobjective $\mathcal{H}_2/\mathcal{H}_\infty$ guaranteed cost PID design. *IEEE Control Systems Magazine*, 1997.

[Tél02a] France Télécom. Document de référence 2001. Technical report, France Télécom, 2002.

[Tél02b] France Télécom, editor. *Les communications optiques du futur*, June 2002. Mémento technique France Télécom 19. http://www.rd.francetelecom.fr/fr/conseil/mento19/index.html.

[Wei97] M.A. Weiss. *Data structures and algorithm analysis in C*. Addison-Wesley, second edition, 1997. ISBN: 0-201-49840-5.

[YR02] X. Yang and B. Ramamurthy. An analytical model for virtual topology reconfiguration in optical networks and a case study. In *International Conference on Computer Communications and Networks (ICCCN)*, pages 302–306. IEEE, October 2002. DOI: Digital Object Identifier 10.1109/ICCCN.2002.1043082.

[YR04] W. Yao and B. Ramamurthy. Dynamic traffic grooming using fixed alternate routing in WDM mesh optical networks. Traffic grooming workshop, co-located with IEEE/Create-Net BROADNETS 2004, October 2004.

[ZT99] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: A comparative case study and the Strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, November 1999.

[ZZZM02] H. Zhu, H. Zang, K. Zhu, and B. Mukherjee. Dynamic traffic grooming in WDM mesh networks using a novel graph model. In *Globecom 2002*, pages 2681–2685, 2002.

Appendix A

Classical formulation for the reconfiguration problem

In section 4.2, we compare our source formulation to the "classical" formulation. Follows this classical formulation. We use similar notations as the one defined in section 3.

A.1 Variables

We use the following variables:

- $p^{i,j}_{m,n,w}(t) \in \mathbb{N}$: number of wavelengths w used by lightpaths between i and j on physical link (m,n) during time period t;
- $c_{i,j}(t) \in \mathbb{N}$: number of lightpaths from node i to node j during time period t;
- $f^{s,d}_{i,j}(t)$: part of demand from node s to node d using a lightpath from i to j during time period t.
- $\Delta p^{i,j}_{m,n,w}(t) \in \mathbb{N}$ is the number of changes for the number of wavelengths w used by lightpaths from node i to node j physical link $(m,n) \in \mathcal{L}$, between time period $t-1$ and t.

A.2 Virtual topology constraints

$$\sum_{m=1}^{N} p^{i,j}_{m,k,w}(t) - \sum_{n=1}^{N} p^{i,j}_{k,n,w}(t) = 0, \quad \begin{array}{l}(i,k,j) \in N^3, k \neq i,j \\ 1 \leqslant w \leqslant \mathcal{W} \\ 1 \leqslant t \leqslant \mathcal{T}\end{array} \tag{A.1}$$

$$\sum_{n=1}^{N} \sum_{w=1}^{\mathcal{W}} p^{i,j}_{i,n,w}(t) - c_{i,j}(t) = 0, \quad \begin{array}{l}\forall (i,j) \in N^2 \\ 1 \leqslant t \leqslant \mathcal{T}\end{array} \tag{A.2}$$

$$\sum_{m=1}^{N}\sum_{w=1}^{\mathcal{W}} p_{m,j,w}^{i,j}(t) - c_{i,j}(t) = 0, \forall\,(i,j) \in N^2 \quad \begin{matrix}\forall (i,j) \in N^2 \\ 1 \leqslant t \leqslant \mathcal{T}\end{matrix} \tag{A.3}$$

$$\sum_{(i,j)} p_{m,n,w}^{i,j}(t) \leqslant P_{(m,n)}, \quad \begin{matrix}\forall (i,j) \in N^2 \\ 1 \leqslant w \leqslant \mathcal{W} \\ 1 \leqslant t \leqslant \mathcal{T}\end{matrix} \tag{A.4}$$

A.3 Routing constraints

$$\sum_{i=1}^{N} f_{i,k}^{s,d}(t) - \sum_{j=1}^{N} f_{k,j}^{s,d}(t) = 0, \quad \begin{matrix}\forall (s,d,k) \in N^3, k \neq s,d \\ 1 \leqslant t \leqslant \mathcal{T}\end{matrix} \tag{A.5}$$

$$\sum_{j=1}^{N} f_{s,j}^{s,d}(t) = \mathcal{D}_{s,d}(t), \quad \begin{matrix}\forall (s,d) \in N^2 \\ 1 \leqslant t \leqslant \mathcal{T}\end{matrix} \tag{A.6}$$

$$\sum_{i=1}^{N} f_{i,d}^{s,d}(t) = \mathcal{D}_{s,d}(t), \quad \begin{matrix}\forall (s,d) \in N^2 \\ 1 \leqslant t \leqslant \mathcal{T}\end{matrix} \tag{A.7}$$

$$\sum_{(s,d)} f_{i,j}^{s,d}(t) - \mathcal{B}c_{i,j}(t) \leqslant 0 \quad \begin{matrix}\forall (i,j) \in N^2 \\ 1 \leqslant t \leqslant \mathcal{T}\end{matrix} \tag{A.8}$$

A.4 Reconfiguration constraints

$$p_{m,n,w}^{i,j}(t) - p_{m,n,w}^{i,j}(t-1) \leqslant \Delta p_{m,n,w}^{i,j}(t), \quad \begin{matrix}(m,n,i,j) \in N^4 \\ 1 \leqslant w \leqslant \mathcal{W} \\ 1 \leqslant t \leqslant \mathcal{T}\end{matrix} \tag{A.9}$$

$$p_{m,n,w}^{i,j}(t-1) - p_{m,n,w}^{i,j}(t) \leqslant \Delta p_{m,n,w}^{i,j}(t), \quad \begin{matrix}(m,n,i,j) \in N^4 \\ 1 \leqslant w \leqslant \mathcal{W} \\ 1 \leqslant t \leqslant \mathcal{T}\end{matrix} \tag{A.10}$$

A.5 Objective functions

A.5.1 Number of optical links

$$O(t) = \sum_{(m,n)}\sum_{(i,j)}\sum_{w=1}^{\mathcal{W}} p_{m,n,w}^{i,j}(t), \; 1 \leqslant t \leqslant \mathcal{T} \tag{A.11}$$

A.5.2 Number of lightpaths

$$L(t) = \sum_{(i,j)} c_{i,j}(t),\ 1 \leqslant t \leqslant \mathcal{T} \tag{A.12}$$

A.5.3 Maximum physical link load

$$M(t) = M_l(t),\ 1 \leqslant t \leqslant \mathcal{T} \tag{A.13}$$

with

$$\sum_{(i,j)} \sum_{w=1}^{\mathcal{W}} p^{i,j}_{m,n,w}(t) \leqslant M_l(t),\ \forall\,(m,n) \in N^2 \tag{A.14}$$

A.5.4 Average number of hops

$$H(t) = \frac{1}{\sum_{s,d} \mathcal{D}_{s,d}(t)} \sum_{i,j} \sum_{s,d} f^{s,d}_{i,j}(t),\ 1 \leqslant t \leqslant \mathcal{T} \tag{A.15}$$

A.5.5 Number of reconfigurations

$$C(t) = \sum_{(m,n)} \sum_{(i,j)} \sum_{w=1}^{\mathcal{W}} \Delta p^{i,j}_{m,n,w}(t),\ 1 \leqslant t \leqslant \mathcal{T} \tag{A.16}$$

Appendix B

Mono-objective optimization: Computational results

We provide in this chapter the numerical values for the experiments regarding the mono-objective approach. In some of the following tables appear percentages in parenthesis. This percentage corresponds to the solution gap (when applicable).

B.1 Classical and source formulation: computational results

Table B.1: SN2, Cost239 and NSFNET ($\mathcal{W} = 8$) instances, computation time (s)

	"classical" formulation	**Source formulation**
	SN2 network	
O	3652^{+}	3628^{+}
L	3638^{+}	3623^{+}
M	36.01	3.86
H	2.609	0.58
C	12.68	17.28
	Cost239 network	
O	3636^{+}	3624^{+}
L	3626^{+}	3613^{+}
M	3605^{0}	344.45
H	31.81	5.55
C	3614^{0}	2359
	NSFNET network ($\mathcal{W} = 8$)	
O	*	3618^{+}
L	3626^{+}	3608^{+}
M	3605^{0}	642.4
H	31.81	12.71
C	3614^{0}	3605^{0}

Table B.2: NSFNET ($\mathcal{W} = 16$) and N20 instances, Computation time (s)

	"classical" formulation	**Source formulation**
	NSFNET network ($\mathcal{W} = 16$)	
O	*	7223^{0}
L	*	7212^{+}
M	*	2377
H	*	16.96
C	*	7209^{0}
	N20 network	
O	*	7242^{0}
L	*	7235^{0}
M	*	2494
H	*	67.60
C	*	7235^{0}

Table B.3: SN2, Cost239 and NSFNET ($\mathcal{W} = 8$) instances, solution value

	"classical" formulation	**Source formulation**
	SN2 network	
O	314 (5.03%)	312 (4.88%)
L	239 (3.68%)	240 (4.06%)
M	20	20
H	5.00	5.00
C	0	0
	Cost239 network	
O	1896 (2.31%)	1871 (1.00%)
L	1111 (1.24%)	1111 (1.24%)
M	-	56
H	5.00	5.00
C	-[0]	0
	NSFNET network ($\mathcal{W} = 8$)	
O	-	1894 (5.99%)
L	953 (1.66%)	967 (3.08%)
M	-	60
H	5.00	5.00
C	-	-

Table B.4: NSFNET ($\mathcal{W} = 16$) and N20 instances, solution value

	"classical" formulation	**Source formulation**
	NSFNET network ($\mathcal{W} = 16$)	
O	-	-
L	-	1812 (0.8%)
M	-	120
H	-	5.00
C	-	-
	N20 network	
O	-	-
L	-	-
M	-	62
H	-	3.00
C	-	-

Table B.5: SN2, Cost239, NSFNET and N20 instances, problem size

	"classical" formulation	**Source formulation**
	SN2 network	
integer var.	60480	11760
continuous var.	50128	12832
constraints num.	107272	22425
	Cost239 network	
integer var.	880000	96800
continuous var.	710684	92924
constraints num.	1493104	169089
	NSFNET network ($\mathcal{W} = 8$)	
integer var.	1230320	101920
continuous var.	998005	102565
constraints num.	2061656	175296
	NSFNET network ($\mathcal{W} = 16$)	
integer var.	-	203840
continuous var.	-	191381
constraints num.	-	347408
	N20 network	
integer var.	-	182400
continuous var.	-	154739
constraints num.	-	257772

B.2 Metrics and cut efficiency: Computational results

B.2.1 Cut performance

Table B.6: SN1, SN2, Cost239 and NSFNET instances, computation time (s)

	nocut	**flow**	**in1**	**in2**	**out1**	**out2**	**out3**	**sym1**	**sym2**
	Line4 network								
O	3632^{+}	3673^{+}	3677^{+}	3666^{+}	3628^{+}	3637^{+}	3630^{+}	3637^{+}	3628^{+}
L	3624^{+}	3629^{+}	3630^{+}	1701	3632^{+}	3634^{+}	714	3631^{+}	3626^{+}
M	0.43	0.73	0.56	0.45	0.48	0.60	0.35	0.83	0.48
H	0.23	0.36	0.31	0.31	0.24	0.25	0.24	0.32	0.24
C	0.09	0.11	0.14	0.11	0.12	0.11	0.08	3.35	0.11
	SN1 network								
O	3632^{+}	3648^{+}	3642^{+}	3642^{+}	3650^{+}	3648^{+}	3635^{+}	3628^{+}	3631^{+}
L	3624^{+}	3640^{+}	3632^{+}	3639^{+}	3646^{+}	3643^{+}	3622^{+}	3616^{+}	3621^{+}
M	33.7	22.9	71.1	3635^{+}	32.5	71.8	37.6	118.	66.1
H	8.17	12.3	10.1	9.61	9.61	9.42	8.17	9.64	9.22
C	12.7	18.1	13.6	17.8	21.3	170	13.5	197	28.9
	SN2 network								
O	3627^{+}	3634^{+}	3634^{+}	3628^{+}	3636^{+}	3631^{+}	3632^{+}	3627^{+}	3632^{+}
L	3622^{+}	3629^{+}	3630^{+}	3624^{+}	3638^{+}	3632^{+}	3624^{+}	3616^{+}	3623^{+}
M	82.3	112	34.1	46.0	28.8	79.5	12.0	93.5	45.5
H	8.95	12.5	9.11	8.57	9.64	9.83	9.02	14.1	9.73
C	17.8	32.3	18.6	20.8	28.7	18.5	21.2	134	34.5
	NSFNET network								
O	3629^{+}	3642^{+}	3640^{+}	3632^{+}	3629^{+}	3629^{+}	3629^{+}	3606^{0}	3619^{+}
L	3618^{+}	3633^{+}	3624^{+}	3621^{+}	3614^{+}	3619^{+}	3617^{+}	3606^{0}	3613^{+}
M	395	416	207	495	311	222	484	3604^{+}	872
H	44	95	74	48	46	49	63	1834	146
C	3572	3622^{0}	3205	3607^{0}	3603^{0}	3134	3128	3607^{0}	3605^{0}
	Cost239 network								
O	3632^{0}	3646^{0}	3637^{+}	3639^{+}	3611^{+}	3633^{+}	3630^{+}	3624^{0}	3633^{+}
L	3627^{+}	3628^{+}	3624^{+}	3618^{+}	3616^{+}	3614^{+}	3618^{+}	3615^{+}	3611^{+}
M	207	452	156	183	176	157	313	1795	2935
H	45	42	67	33	28	28	6	108	9
C	3615^{0}	3624^{0}	3614^{0}	3612^{0}	3607^{0}	3615^{0}	3606^{0}	3619^{0}	3605^{0}

Table B.7: SN1, SN2, Cost239 and NSFNET instances, solution gap (%)

	nocut	**flow**	**in1**	**in2**	**out1**	**out2**	**out3**	**sym1**	**sym2**
	Line4 network								
O	11.76	11.48	7.8	6.91	9.1	8.52	7.61	12.48	11.55
L	7.0	7.25	4.87	1.67	6.23	4.62	1.67	7.25	7.04
	SN1 network								
O	7.16	7.25	6.12	6.79	6.56	6.00	6.44	7.03	7.82
L	11.71	11.20	6.56	6.39	5.43	4.40	4.92	11.72	11.25
M	0	0	0	4.17	0	0	0	0	0
	SN2 network								
O	6.06	6.28	4.88	4.56	4.76	5.08	4.76	6.33	6.90
L	11.34	12.13	4.99	5.35	7.09	7.04	6.99	11.78	11.88
	NSFNET network								
O	0.25	0.31	0.28	0.31	0.25	0.28	0.28	∞	0.38
L	1.90	1.96	1.49	1.49	1.50	1.62	1.62	∞	1.96
C	0	∞	0	∞	∞	0	0	∞	∞
	Cost239 network								
O	∞	∞	2.67	2.75	6.11	2.35	2.67	∞	1.77
L	9.12	9.89	5.92	5.65	∞	1.89	5.36	9.28	1.74
C	∞	∞	∞	∞	∞	∞	∞	∞	∞

B.2.2 Lower bound performance

Table B.8: SN1, SN2, Cost239 and NSFNET instances, computation time (s) and solution value

	Computation time (s)			**Solution value**		
	ref	**lb**	**rel**	**ref**	**lb**	**rel**
	line4 network					
O	3641^+	3629^+	0.019	118	118	102.69
L	1045	15.5	0.016	60	60	48.699
M	0.07	0.02	0.019	25	25	21.525
H	0.05	0.01	0.017	5	5	5
C	0.09	0.01	0.050	0	0	0
	SN1 network					
O	3649^+	3617^+	0.42	325	329	300.775
L	3635^+	3612^+	0.30	182	186	157.5
M	14.8	0.58	1.58	23	23	20.4833
H	0.71	0.12	0.28	5	5	4.99999
C	14.1	0.21	11.3	0	0	0
	SN2 network					
O	3631	3619^+	0.43	295	300	276.874
L	3622	3611^+	0.30	180	185	155.524
M	8.17	0.73	2.10	16	16	14.4
H	0.69	0.13	0.28	5	5	4.99999
C	18.2	0.24	15.5	0	0	0
	NSFNET network					
O	3627^+	3605^+	8.92	3136	3137	3127.2
L	3657^+	3603^+	7.02	802	803	783.8
M	1030	4.63	158	104	104	103.2
H	8.30	0.35	4.97	2	2	2
C	3587	3.36	3605^+	0	0	0
	Cost239 network					
O	3620^+	3609^+	4.53	1189	1203	1169.90
L	3615^+	3604^+	2.34	613	615	562.47
M	265	12.5	51.5	37	37	34.9649
H	5.96	0.55	2.03	4.99	5	5
C	3603^+	4.74	3606^+	-	0	0

B.3 Technological extensions of the model: Computational results

B.3.1 Receivers and transmitters

Table B.9: SN2 and Cost239 instances, computation time (s)

	Number of receivers $\mathcal{R}$ and transmitters $\mathcal{E}$					
	SN2 network					
	8	**10**	**12**	**14**	**16**	∞
O	3644.5^{+}	3630.2^{+}	3630.1^{+}	3627.0^{+}	3629.9^{+}	3646.8^{+}
L	3620.7^{+}	3621.2^{+}	3620.3^{+}	3920.8^{+}	3621.8^{+}	3637.7^{+}
C	477.07	99.1	111.6	105.8	101.2	19.1
	Cost239 network					
	24	**28**	**32**	**36**	**40**	∞
O	7256^{+}	7252^{+}	7253^{+}	7286^{+}	7261^{+}	7280^{+}
L	7230^{+}	7219^{+}	7224^{+}	7261^{+}	7226^{+}	7237^{+}
C	7209^{0}	7213^{0}	7209^{0}	7228^{0}	7211^{0}	7243^{0}

Table B.10: SN2 and Cost239 instances, solution value

	Number of receivers $\mathcal{R}$ and transmitters $\mathcal{E}$		
	SN2 network		
	8	**10**	**12**
O	319 (5.72%)	314 (5.04%)	314 (5.33%)
L	240 (3.99%)	239 (3.66%)	239 (3.67%)
C	0	0	0
	14	**16**	∞
O	313 (5.01%)	313 (4.99%)	312(4.89%)
L	239 (3.64%)	240 (4.02%)	240(4.07%)
C	0	0	0
	Cost239 network		
	24	**28**	**32**
O	1890 (2%)	1872 (1.06%)	1869 (0.9%)
L	1111 (1.22%)	1110 (1.15%)	1110 (1.13%)
C	-	-	-
	36	**40**	∞
O	1875 (1.22%)	1870 (0.95%)	1870 (0.95%)
L	1110 (1.15%)	1111 (1.23%)	1111 (1.23%)
C	-	-	-

B.3.2 Optical link release

Table B.11: SN2 and Cost239 instances, solution value

	Time period				
	1	**2**	**3**	**4**	**5**
	SN2 network ($\mathcal{W} = 8, \mathcal{B} = 40$)				
O	430	345	283	212	135
L	198	172	158	130	91
	SN2 network ($\mathcal{W} = 16, \mathcal{B} = 20$)				
O	1389	1045	676	468	315
L	487	433	300	218	164
	Cost239 network ($\mathcal{W} = 8, \mathcal{B} = 40$)				
O	1648	1542	1373	1277	913
L	917	875	779	775	633

B.3.3 No empty lightpaths

Table B.12: SN2 and Cost239 instances, solution value

	Minimum fill rate $\mathcal{F}$			
	00%	**25%**	**50%**	**75%**
	SN2 network			
O	$312^{+}(4.88\%)$	$312^{+}(4.43\%)$	$313^{+}(4.72\%)$	$312^{+}(4.42\%)$
L	$240^{+}(4.06\%)$	$239^{+}(3.63\%)$	$240^{+}(4.08\%)$	$241^{+}(4.47\%)$
M	20	20	20	20
H	5.00	5.0049	5.0256	5.2356^{+}
C	0	0	0	0
	Cost239 network			
O	$1871^{+}(1.00\%)$	$1872^{+}(1.06\%)$	$1872^{+}(1.06\%)$	$1876^{+}(1.27\%)$
L	$1111^{+}(1.23\%)$	$1111^{+}(1.24\%)$	$1111^{+}(1.22\%)$	$1111^{+}(1.24\%)$
M	56	56	56	56
H	5.0000	5.0014	5.0061	5.0538
C	0	-	-	-

B.4 Comparison with the heuristics: Computational results

Table B.13: Heuristics experiment, computation time (s)

	gr	**sa**(O)	**sa**(L)	**sa**(C)	**md**(O)	**md**(L)	**md**(C)
SN2 (8)	0.04	234	298	18	36241^{+}	36173^{+}	15.75
SN2 (16)	0.08	304	331	41	36333^{+}	36238^{+}	54.44
Cost239 (8)	0.016	642	620	221	36309^{+}	36086^{+}	767.8
Cost239 (16)	0.08	800	813	340	36299^{+}	36093^{+}	2243
NSFN. (8)	0.016	1023	1088	1496	36191^{+}	36054^{+}	8100
NSFN. (16)	0.032	2044	2181	2102	36262^{+}	36060^{+}	36065^{0}
N20 (8)	0.02	3145	3089	3278	86799^{+}	86811^{0}	12236
N20 (16)	0.068	3578	3610	3888	86690^{+}	86521^{0}	86491^{0}
N30 (8)	0.148	3452	3312	3441	86562^{0}	86516^{0}	86562^{0}
N30 (16)	0.332	3609	3549	3456	-*	-*	-*
N40 (8)	0.448	3588	3455	3491	-*	-*	-*
N40 (16)	1.008	3718	3655	3722	-*	-*	-*

Table B.14: Heuristics experiment, solution value with metric O

	gr	**sa**(O)	**sa**(L)	**sa**(C)	**md**(O)	**md**(L)	**md**(C)
SN2 (8)	485	307	392	1228	312 (4.79%)	409	1190
SN2 (16)	780	599	682	1945	603 (1.63%)	691	1900
Cost239 (8)	1549	933	1119	4304	944 (1.90%)	1179	4220
Cost239 (16)	2494	2102	2304	5367	1871 (1.00%)	2252	5215
NSFN (8)	2857	1791	2512	8168	1814 (1.85%)	2559	8100
NSFN (16)	4828	3601	4047	9106	3588 (0.75%)	4038	-
N20 (8)	4122	2682	3189	7810	2655 (1.34%)	-	7782
N20 (16)	6821	5307	5489	15043	5266 (0.53%)	-	-
N30 (8)	14215	3019	3488	8311	-	-	-
N30 (16)	23604	5987	6759	15835	-	-	-
N40 (8)	26735	3141	3790	8671	-	-	-
N40 (16)	44367	6018	7211	16093	-	-	-

Table B.15: Heuristics experiments, solution value with metric L

	gr	**sa**(O)	**sa**(L)	**sa**(C)	**md**(O)	**md**(L)	**md**(C)
SN2 (8)	343	297	246	896	292	239 (3.59%)	875
SN2 (16)	555	556	469	1523	542	447 (1.66%)	1440
Cost239 (8)	878	784	585	2389	780	591 (3.24%)	2335
Cost239 (16)	1418	1410	1194	3381	1406	1110 (1.14%)	3305
NSFN (8)	1380	1343	968	3601	1310	965 (2.87%)	3575
NSFN (16)	2326	2310	1806	4408	2365	1810 (0.69%)	-
N20 (8)	1785	1668	1677	3409	1719	-	3339
N20 (16)	2951	3166	3105	6465	3139	-	-
N30 (8)	6774	2018	2011	3672	-	-	-
N30 (16)	11231	3976	4000	7188	-	-	-
N40 (8)	12206	2231	2467	4005	-	-	-
N40 (16)	20256	4151	4879	8173	-	-	-

Table B.16: Heuristics experiment, solution value with metric C

	gr	**sa**(O)	**sa**(L)	**sa**(C)	**md**(O)	**md**(L)	**md**(C)
SN2 (8)	101	454	606	1	452	613	0
SN2 (16)	168	891	1043	4	885	1040	0
Cost239 (8)	320	1331	1702	1	1308	1698	0
Cost239 (16)	550	2473	3001	8	2744	3364	0
NSFN (8)	605	2345	3666	1	2379	3669	0
NSFN (16)	1052	5107	5934	3	4967	5912	-
N20 (8)	1410	2534	2872	2	2622	-	0
N20 (16)	2350	5575	5891	4	5613	-	-
N30 (8)	2983	2834	3109	6	-	-	-
N30 (16)	5186	5811	6079	11	-	-	-
N40 (8)	5600	2984	3451	9	-	-	-
N40 (16)	9822	5567	6671	15	-	-	-

B.4.1 Single hop cases

Table B.17: SN1, SN2, Cost239 and NSFNET single hop instances, solution value

	Greedy	**md**(O)	**md**(L)	**md**(C)
	Number of optical links			
SN1	591	484	628	655
SN2	741	707	1051	890
Cost239	2514	2389	3062	2870
NSFNET	3379	2801	3208	4700
	Number of lightpaths			
SN1	370	305	305	410
SN2	540	516	516	620
Cost239	1441	1369	1369	1645
NSFNET	1607	1331	1331	1805
	Number of reconfigurations			
SN1	139	664	636	0
SN2	244	1066	1214	0
Cost239	758	3521	3870	0
NSFNET	778	3673	3557	0

Table B.18: N20, N30, N40 and N50 instances, greedy solution value

	N20	N30	N40	N50
O	11342	23439	43905	60573
L	4897	11150	19969	31394
C	3469	7286	13870	18890

Appendix C

Multiobjective optimization: Computational results

C.1 Weighted method with the simulated annealing algorithm: Computational results

Table C.1: SN2 instances, solution (O, L, C) found depending on $(\alpha_O, \alpha_L, \alpha_C)$

	Solver (O, L, C) (gap)	**Simulated annealing** (O, L, C)
$(\alpha_O, \alpha_L, \alpha_C)$	SN2 network ($\mathcal{W} = 8$)	
$(1, 0, 0)$	$(312, 292, 452)$ (4.79%)	$(307, 297, 454)$
$(0, 1, 0)$	$(409, 239, 613)$ (3.59%)	$(392, 246, 606)$
$(0, 0, 1)$	$(1190, 875, 0)$ $(0\ \%)$	$(1228, 896, 1)$
$(1, 0, 1)$	$(318, 304, 5)$ (5.05%)	$(324, 307, 5)$
$(0, 1, 1)$	$(5137, 250, 8)$ $(8.1\ \%)$	$(5188, 253, 8)$
	SN2 network ($\mathcal{W} = 16$)	
$(1, 0, 0)$	$(603, 542, 885)$ (1.63%)	$(599, 556, 891)$
$(0, 1, 0)$	$(691, 447, 1040)$ (1.66%)	$(682, 469, 1043)$
$(0, 0, 1)$	$(1900, 1440, 0)$ $(0\ \%)$	$(1945, 1523, 4)$
$(1, 0, 1)$	$(611, 544, 11)$ (1.58%)	$(616, 584, 13)$
$(0, 1, 1)$	$(953, 459, 5)$ (2.07%)	$(874, 533, 6)$

Table C.2: Cost239 instances, solution (O, L, C) found depending on $(\alpha_O, \alpha_L, \alpha_C)$

	Solver (O, L, C) (gap)	**Simulated annealing** (O, L, C)
$(\alpha_O, \alpha_L, \alpha_C)$	Cost239 network ($\mathcal{W} = 8$)	
$(1, 0, 0)$	$(944, 780, 1308)$ (1.90%)	$(933, 784, 1331)$
$(0, 1, 0)$	$(1179, 591, 1698)$ (3.24%)	$(1119, 585, 1702)$
$(0, 0, 1)$	$(4220, 2335, 0)$ (0 %)	$(4304, 2389, 1)$
$(1, 0, 1)$	$(-, -, -)$	$(1136, 884, 1304)$
$(0, 1, 1)$	$(-, -, -)$	$(1236, 778, 8)$
	Cost239 network ($\mathcal{W} = 16$)	
$(1, 0, 0)$	$(1871, 1406, 2744)$ (1.00%)	$(2102, 1410, 2473)$
$(0, 1, 0)$	$(2252, 1110, 3364)$ (1.14%)	$(2304, 1194, 3001)$
$(0, 0, 1)$	$(5215, 3305, 0)$ (0 %)	$(5367, 3381, 8)$
$(1, 0, 1)$	$(-, -, -)$	$(1069, 863, 11)$
$(0, 1, 1)$	$(-, -, -)$	$(1322, 699, 12)$

Table C.3: NSFNET instances, solution (O, L, C) found depending on $(\alpha_O, \alpha_L, \alpha_C)$

	Solver (O, L, C) (gap)	**Simulated annealing** (O, L, C)
$(\alpha_O, \alpha_L, \alpha_C)$	NSFNET network ($\mathcal{W} = 8$)	
$(1, 0, 0)$	$(1814, 1310, 2379)$ (1.85%)	$(1791, 1343, 2345)$
$(0, 1, 0)$	$(2559, 965, 3669)$ (2.87%)	$(2512, 968, 3666)$
$(0, 0, 1)$	$(8100, 3575, 0)$ (0 %)	$(8168, 3601, 1)$
$(1, 0, 1)$	$(1883, 1249, 12)$ (4.82%)	$(1798, 1210, 11)$
$(0, 1, 1)$	$(-, -, -)$	$(1801, 1306, 2279)$
	NSFNET network ($\mathcal{W} = 16$)	
$(1, 0, 0)$	$(3588, 2365, 4967)$ (0.75%)	$(3601, 2310, 5107)$
$(0, 1, 0)$	$(4038, 1810, 5912)$ (0.69%)	$(4047, 1806, 5934)$
$(0, 0, 1)$	$(-, -, -)$	$(9106, 4408, 3)$
$(1, 0, 1)$	$(-, -, -)$	$(1914, 1223, 11)$
$(0, 1, 1)$	$(-, -, -)$	$(1943, 1405, 2406)$

Table C.4: N20 instances, solution (O, L, C) found depending on $(\alpha_O, \alpha_L, \alpha_C)$

	Solver (O, L, C) (gap)	**Simulated annealing** (O, L, C)
$(\alpha_O, \alpha_L, \alpha_C)$	N20 network ($\mathcal{W} = 8$)	
$(1, 0, 0)$	$(2655, 1719, 2622)$ (1.34%)	$(2682, 1668, 2534)$
$(0, 1, 0)$	$(-, -, -)$	$(3189, 1677, 2872)$
$(0, 0, 1)$	$(7782, 3339, 0)$ (0%)	$(7810, 3409, 2)$
$(1, 0, 1)$	$(-, -, -)$	$(2694, 1745, 2676)$
$(0, 1, 1)$	$(-, -, -)$	$(2781, 1783, 2760)$
	N20 network ($\mathcal{W} = 16$)	
$(1, 0, 0)$	$(5266, 3139, 5613)$ (0.53%)	$(5307, 3166, 5575)$
$(0, 1, 0)$	$(-, -, -)$	$(5489, 3105, 5891)$
$(0, 0, 1)$	$(-, -, -)$	$(15043, 6465, 4)$
$(1, 0, 1)$	$(-, -, -)$	$(5038, 3104, 5428)$
$(0, 1, 1)$	$(-, -, -)$	$(5120, 3165, 5603)$

Table C.5: N30 and N40 instances, solution (O, L, C) found depending on $(\alpha_O, \alpha_L, \alpha_C)$

	$\mathcal{W} = 8$ (O, L, C)	$\mathcal{W} = 16$ (O, L, C)
$(\alpha_O, \alpha_L, \alpha_C)$	N30 network	
$(1, 0, 0)$	$(3019, 2018, 2834)$	$(5987, 3976, 5811)$
$(0, 1, 0)$	$(3488, 2011, 3109)$	$(6759, 4000, 6079)$
$(0, 0, 1)$	$(8311, 3672, 6)$	$(15835, 7188, 11)$
$(1, 0, 1)$	$(3096, 2113, 2901)$	$(6192, 4014, 5689)$
$(0, 1, 1)$	$(3200, 2103, 2944)$	$(6371, 4218, 5763)$
	N40 network	
$(1, 0, 0)$	$(3141, 2231, 2984)$	$(6018, 4151, 5567)$
$(0, 1, 0)$	$(3790, 2467, 3451)$	$(7211, 4879, 6671)$
$(0, 0, 1)$	$(8671, 4005, 9)$	$(16093, 8173, 15)$
$(1, 0, 1)$	$(3134, 2345, 3024)$	$(6094, 4487, 5831)$
$(0, 1, 1)$	$(3410, 2409, 3310)$	$(6652, 4674, 6598)$

C.2 Results with the ϵ-restricted method: Computational results

Table C.6: SN2 and Cost239 instances, mono-objective solution value

	SN2				
O	312	411	502	519	1190
L	292	240	359	314	875
M	23	41	20	45	75
H	6.70742	5.51315	6.8943	5	10.0027
C	452	618	721	762	0
	Cost239				
O	947	1179	1210	1620	4220
L	770	591	804	798	2335
M	48	58	30	71	195
H	7.22059	5.50564	7.04223	5	11.6793
C	1296	1698	1698	2310	0

Table C.7: SN2 instance, obtained nondominated points

	Solutions					
O	443	356	327	327	334	328
L	314	263	287	287	283	292
M	45	45	45	24	20	25
H	5	5.5	6.5	6.62003	6.42093	6.61749
C	762	762	762	762	721	721
O	480	480	517	408	610	610
L	465	465	314	265	355	355
M	20	20	29	20	75	30
H	10.0027	8.33728	5	5.51315	5	5
C	0	250	762	618	0	250
O	411	404	312	400	485	515
L	240	250	292	289	314	254
M	41	41	23	23	45	45
H	5.46235	5.2761	6.68327	5.06639	5	5.25
C	618	618	452	452	30	762
O	329	502	500	490	395	
L	250	328	358	355	295	
M	23	20	30	20	41	
H	5.51315	5.06873	5	6.8943	5.51315	
C	618	721	721	0	0	

Table C.8: Cost239 instance, obtained nondominated points

	Solutions					
$\begin{pmatrix} O \\ L \\ M \\ H \\ C \end{pmatrix}$	$\begin{pmatrix} 1422 \\ 798 \\ 71 \\ 5 \\ 2310 \end{pmatrix}$	$\begin{pmatrix} 1595 \\ 798 \\ 44 \\ 5 \\ 2310 \end{pmatrix}$	$\begin{pmatrix} 4210 \\ 895 \\ 195 \\ 5.00165 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1210 \\ 804 \\ 30 \\ 6.41182 \\ 1698 \end{pmatrix}$	$\begin{pmatrix} 1210 \\ 698 \\ 75 \\ 5.09596 \\ 1698 \end{pmatrix}$	$\begin{pmatrix} 1179 \\ 591 \\ 58 \\ 5.47011 \\ 1698 \end{pmatrix}$
$\begin{pmatrix} O \\ L \\ M \\ H \\ C \end{pmatrix}$	$\begin{pmatrix} 1179 \\ 683 \\ 58 \\ 5.12357 \\ 1698 \end{pmatrix}$	$\begin{pmatrix} 947 \\ 770 \\ 48 \\ 7.18322 \\ 1296 \end{pmatrix}$	$\begin{pmatrix} 1250 \\ 719 \\ 48 \\ 5.06568 \\ 1296 \end{pmatrix}$	$\begin{pmatrix} 1618 \\ 798 \\ 71 \\ 5 \\ 76 \end{pmatrix}$	$\begin{pmatrix} 1582 \\ 786 \\ 71 \\ 7.27764 \\ 4 \end{pmatrix}$	$\begin{pmatrix} 1185 \\ 795 \\ 30 \\ 7.04223 \\ 0 \end{pmatrix}$