Université de Montréal

Une métaheuristique pour le problème d'affectation de longueurs d'onde, de groupage et de routage du trafic dans les réseaux optiques WDM

par

Yannick Solari

Département d'informatique et de recherche opérationnelle,
Université de Montréal
Faculté des arts et des sciences

Mémoire présenté à la Faculté des études supérieures
en vue de l'obtention du grade de Maître ès sciences (M. Sc.)
en informatique

Août, 2005

Université de Montréal
Faculté des études supérieures

Ce mémoire intitulé :

Une métaheuristique pour le problème d'affectation de longueurs d'onde, de groupage et de routage du trafic dans les réseaux optiques WDM

présenté par :
Yannick Solari

a été évaluée par un jury composé des personnes suivantes :

Jean-Yves Potvin, président-rapporteur
Brigitte Jaumard, directrice de recherche
Alain Houle, co-directeur
Abdelhakim Hafid, membre du jury

Mémoire accepté le 4 novembre 2005

Résumé

Le développement récent de la technologie de multiplexage de division de longueurs d'onde a conduit à un accroissement significatif des capacités de transmission dans les réseaux à large diffusion. Ces réseaux ne sont plus limités dans leur capacité de transmission, mais plutôt par leur capacité de traitement dans les commutateurs électroniques, les routeurs et les multiplexeurs. Un domaine de recherche clé correspond au problème GRWA, c'est-à-dire au problème d'affectation de longueurs d'onde couplée avec le groupage et le routage du trafic avec pour objectif de minimiser le nombre (et donc le coût) des équipements électroniques nécessaires pour supporter le trafic dans le réseau tout en intégrant une limite sur le nombre de sauts optiques. Nous étudions un élément important de la couche physique, soit la problématique de la compensation en fonction de la distance parcourue par les longueurs d'onde, dépendant de leur granularité. Ce travail présente une étude sur la conception et l'implantation efficace d'une heuristique de type Recherche Tabou pour minimiser le coût des plateformes MSPP (MultiService Provisioning Platform) dans un réseau optique WDM ($2^{ème}$ génération SONET) tout en optimisant le groupage du trafic sous des conditions de routage avec des flots bifurqués. Le coût d'une plateforme MSPP est estimé par son nombre de cartes de transport. Nous intégrons également le coût de la compensation que nous minimisons conjointement avec le coût des cartes MSPP. L'heuristique développée pour ce problème particulier permet de définir le dimensionnement GRWA d'un réseau WDM à coût minimum, avec le meilleur compromis possible entre le coût des cartes de transport et celui de la compensation.

Mots-clés : réseaux optiques WDM, heuristique, groupage, routage, compensation, saut optique, plateforme MSPP, granularité

Abstract

The recent development of wavelength division multiplexing technology has lead to a significant increase of transmission capacity in broadband networks. These networks are no more limited by their transmission capacity, but by the handling capacity of electronic switches, routers and multiplexers. A key research domain is the GRWA problem, i.e. the problem of wavelengths assignment combined with the grooming and the routing of the traffic, with the aim of minimizing the number (and therefore the cost) of optical and electrical components needed to support the traffic of a network where furthermore the number of optical hops is limited. We are also studying an important element of the optical layer that is the compensation issue, where the compensation depends on the distance covered by the wavelengths and also of their granularity. This study is about the design and the efficient implementation of a Tabu Search heuristic aiming to minimize the cost of MSPP (MultiService Provisioning Platform) platforms in a WDM optical network (SONET 2^{nd} generation), while also optimizing the grooming of traffic under routing conditions allowing diverse routing. The cost of an MSPP platform will be estimated by its number of transport cards. The compensation cost will also be studied and minimized along with the cost of MSPP cards. For this particular problem, the developed heuristic allows the definition of the GRWA network provisioning for a WDM network at minimum cost, with the best compromise between the cost of transport cards and the compensation one.

Keywords : WDM optical networks, heuristic, grouping, routing, compensation, optical hops, MSPP platform, granularity

Table des matières

Liste des tableaux

Liste des figures

Remerciements

Mes remerciements s'adressent en premier lieu à ma directrice Mme Brigitte Jaumard qui m'a accueilli au sein de son équipe et qui a dirigé ce travail. Sa disponibilité, l'aide constante qu'elle m'a apportée et ses nombreux conseils m'ont permis de bien mener à terme ce projet.

Je tiens à remercier aussi mon co-directeur Alain Houle pour le support qu'il m'a apporté tout au long de ce travail et dont les remarques et suggestions m'ont permis de bien guider ma recherche.

Enfin, je tiens à exprimer toute ma gratitude à ma famille et mes amis pour leur appui, ainsi qu'à ma copine Manon qui m'a offert son aide et son soutien chaleureux tout au long de cette période.

Chapitre 1 – Introduction

1.1 Motivation

Dans le domaine des télécommunications, les réseaux de fibre optique WDM (*Wavelength Division Multiplexing*) apparaissent être une infrastructure prometteuse pour pouvoir desservir la demande croissante en bande passante. Le grand débit que peut supporter une fibre optique, le fait de pouvoir utiliser simultanément plusieurs canaux (longueurs d'onde) et le faible coût des fibres optiques sont les principales caractéristiques qui sont à l'origine de cette popularité grandissante.

Aujourd'hui, les aspects de la couche physique relatifs à un réseau de fibre optique sont bien connus. Certains paramètres tel le nombre de longueurs d'onde pouvant partager une même fibre, la distance que peut parcourir un signal lumineux, ou encore le débit des informations qui transitent par les fibres optiques sont autant d'aspects que l'électronique actuelle permet de maximiser. Cependant, nous en sommes à un point où il devient de plus en plus difficile et coûteux d'améliorer ces performances au niveau de la couche physique.

Une autre façon de pouvoir améliorer les performances consiste à faire de l'optimisation au niveau de la couche réseau, c'est-à-dire d'améliorer la façon de gérer le trafic qui y transite. Cette approche a pour objectif général d'utiliser l'équipement le plus efficacement possible pour satisfaire la demande. Rappelons ici que la dérégulation qui a eu lieu dans le domaine des télécommunications, ainsi que la perte des monopoles, a forcé les entreprises à se préoccuper davantage des coûts. Ainsi, au niveau de la gestion du trafic, des gains substantiels peuvent être obtenus sans avoir à remettre en cause

l'équipement utilisé. Dans le domaine des réseaux optiques, il s'agit d'un problème complexe auquel il convient de s'intéresser pour que cette technologie arrive à maturité.

Nous proposons de nous intéresser au problème de groupage et de routage du trafic, en même temps qu'au problème d'affectation de longueurs d'onde soit le problème GRWA (*Grooming, Routing and Wavelength Assignment*). De plus, pour pouvoir répondre à des besoins concrets, notre modèle doit se rapprocher le plus possible de la réalité en tenant compte de différentes caractéristiques physiques inhérentes à ce type de réseau. Ainsi, par exemple, nous considérerons les réseaux maillés qui sont amenés à prendre une place de plus en plus importante, au détriment des réseaux en anneau classiques ; c'est ce qui nous amène d'ailleurs à considérer le problème de routage du trafic qui était pratiquement inexistant dans les anneaux.

De son côté, la littérature propose plusieurs formulations exactes à ce problème typique. Toutefois, les heuristiques semblent être la voie à suivre puisque leurs performances peuvent être comparables à celles des méthodes exactes alors qu'elles permettent de résoudre beaucoup plus efficacement des problèmes de plus grande taille (plus représentatifs de la réalité). C'est pour cette raison que dans le cadre de ce mémoire, nous avons défini une heuristique de type « recherche tabou » où nous considérons simultanément plusieurs aspects souvent simplifiés dans les travaux antérieurs.

1.2 Contribution

- Dans ce mémoire, nous développons une heuristique de type recherche tabou pour résoudre le problème du GRWA où il y a continuité de longueurs d'onde et où des cartes de transport de débit OC-48 et OC-192 sont considérées. Le fait d'utiliser

simultanément deux types de cartes est exploité pour pouvoir en tirer profit. Cet aspect du problème n'a d'ailleurs jamais encore fait l'objet d'aucune étude.

- Nous proposons aussi une version de l'heuristique où la continuité de longueurs d'onde pour tous les sauts optiques n'est plus une contrainte. Un algorithme supplémentaire qui effectue une optimisation de la solution est développé pour atteindre cet objectif.

- Nous intégrons également la possibilité de considérer le problème GRWA en même temps qu'un aspect de la couche physique : la compensation. L'originalité de ce travail tient au fait que nous considérons simultanément deux couches distinctes : la couche réseau et la couche physique. L'objectif est de trouver le meilleur compromis entre l'utilisation des cartes et la compensation.

- Nous proposons enfin un plan de simulation sur les réseaux NSF (le réseau de l'article [14]) et EON2004 (voir l'article [2]) où les paramètres de l'algorithme de recherche tabou sont testés puis validés. Les performances de l'heuristique sont ensuite évaluées pour dimensionner un réseau de type NSF ou EON2004.

1.3 Organisation du mémoire

Le chapitre 2 présente la description complète du problème. Les éléments de base y sont présentés, les termes particuliers utilisés tout au long de ce mémoire sont définis. Les hypothèses et les contraintes relatives au problème que nous voulons résoudre y sont détaillées. La revue de la littérature quant à elle discute des travaux antérieurs qui sont en lien avec ce mémoire.

Le troisième chapitre présente les concepts clés avec lesquels nous allons travailler. Nous expliquons en détail la façon de déterminer l'équipement nécessaire pour supporter le trafic une fois groupé, routé et affecté à des longueurs d'onde pour un réseau donné. Les caractéristiques de cette configuration d'équipement sont présentées, ainsi que les règles qu'il faut respecter et les éléments qui vont nous permettre d'améliorer la qualité des solutions trouvées.

Le quatrième chapitre discute de la compensation. Notre heuristique peut considérer, en plus du problème GRWA, la compensation. Ainsi, toutes les règles qui en régissent le fonctionnement et les stratégies mises en œuvre pour traiter correctement cette caractéristique physique y sont expliquées.

Ensuite, le cinquième chapitre explique comment nous résolvons le problème GRWA dans le cadre d'une heuristique de type recherche tabou. Tous les éléments expliqués précédemment sont mis en pratique et combinés pour aboutir avec une stratégie complète de recherche de solution.

Puis finalement, le sixième chapitre présente les expérimentations effectuées accompagnées des résultats. L'analyse de ces résultats permettra de tirer plusieurs conclusions quant aux performances de l'heuristique et à la qualité des solutions obtenues.

Chapitre 2 – Description du problème

Le présent chapitre constitue une introduction aux aspects plus techniques du problème qui fait l'objet de ce mémoire de maîtrise. Tout d'abord, les structures de base, ainsi que leur intérêt, sont présentées brièvement pour donner une meilleure définition ainsi qu'une vue d'ensemble des différents éléments avec lesquels nous allons travailler. À ce sujet, il est approprié de consulter la référence [16] qui présente beaucoup d'informations d'ordre plus général sur les réseaux optiques. Par la suite, nous définirons les termes particuliers utilisés tout au long de ce travail ainsi que les hypothèses considérées. La revue de la littérature conclura ce chapitre.

2.1 Le réseau

Commençons tout d'abord par décrire, de façon générale, les caractéristiques des réseaux avec lesquels nous allons travailler. Soit un réseau optique WDM (un réseau maillé) décrit comme étant un multigraphe défini par un ensemble de nœuds $V = \{v_1, v_2, \ldots, v_n\}$ et un ensemble d'arcs $E = \{e_1, e_2, \ldots, e_m\}$. Les nœuds du graphe correspondent aux nœuds du réseau et les arcs correspondent aux liens asymétriques qui relient les nœuds. De façon concrète, chaque lien asymétrique correspond à une fibre optique. Toutes les fibres optiques peuvent transporter un nombre maximal fixe de longueurs d'onde, nous appellerons ce nombre W.

Les longueurs d'onde voyagent ensemble à l'intérieur d'une fibre, mais elles sont toutes traitées séparément une fois rendues à l'intérieur d'un nœud. Les nœuds sont

équipés de multiplexeurs et de démultiplexeurs pour pouvoir effectuer cette opération. La Figure 1 illustre ce phénomène.

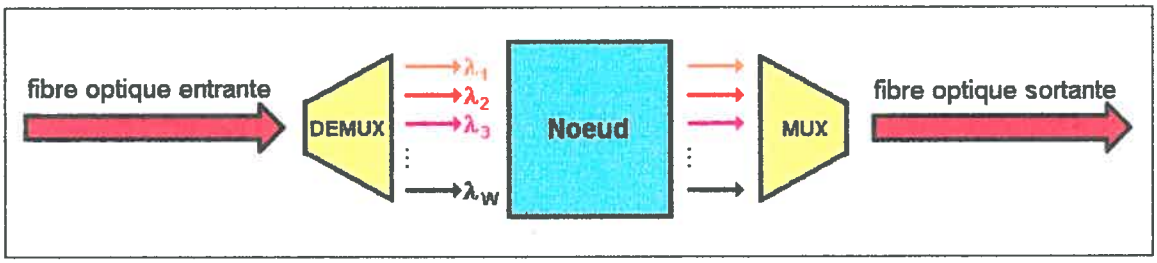

Figure 1 - Les longueurs d'onde passent par un nœud du réseau

De plus, chaque fibre optique est dirigée, elle relie un nœud source vers un autre nœud, le nœud destination. Du point de vue du nœud source, nous disons que cette fibre est une *fibre sortante* ; alors que du point de vue du nœud destination, la fibre en question est qualifiée de *fibre entrante*.

2.2 Le trafic

Pour ce qui est du trafic, il est décrit par un ensemble de matrices pour chaque granularité : R^t_{sd} pour $t \in T$ où $T = \{1, 3, 12, 24, 48\}$. Chaque élément R^t_{sd} de la matrice correspond au nombre de requêtes (unidirectionnelles) qui requièrent un débit OC-t de la source v_s à la destination v_d. Un débit OC-1 équivaut à 51.7 Mb/s et un débit OC-t équivaut à t fois le débit OC-1. La Table 1 ci-après indique la correspondance en bits par seconde des débits les plus couramment utilisés.

	Débit en bit/s
OC-1	51,7 Mb/s
OC-3	155 Mb/s
OC-12	622 Mb/s
OC-48	2,5 Gb/s
OC-192	10 Gb/s

Table 1 - Les différents débits

L'ensemble du trafic sera par la suite transféré dans une matrice unique : la matrice des demandes. La matrice des demandes K indique le débit total demandé entre une source et une destination. Cette matrice est définie ainsi : $K = \{k_{12}, k_{13}, \ldots k_{nn-1}\}$ où

$$k_{sd} = \sum_{t \in T} R_{sd}^{t} \times t \ .$$

Il est important de noter que les matrices de trafic sont asymétriques.

2.3 L'objectif

Aujourd'hui, dans les réseaux optiques, le problème n'est plus la bande passante disponible puisque les performances offertes par les infrastructures mises en place dans ce domaine sont bien suffisantes. Ainsi, il nous apparaît moins pertinent de s'intéresser au taux de blocage dans les réseaux optiques. En fait, c'est le problème de coût des équipements et des délais de propagation qui sont d'intérêt de nos jours. C'est pour cette raison que dans le cadre de ce projet, nous étudierons les façons de minimiser le coût des équipements des réseaux optiques en considérant les délais de propagation.

Ainsi, l'objectif du projet est, étant donné un réseau, une matrice de trafic et W (le nombre maximal de longueurs d'onde disponibles), de déterminer la meilleure façon

possible de grouper le trafic, router le trafic et l'affecter sur les différentes longueurs d'onde dans le but de minimiser le coût du matériel. Ce coût est évalué par le coût des différents équipements nécessaires pour répondre à la demande (il s'agit en fait des cartes de transport et des compensateurs).

2.4 Définition des termes

Voici une liste (en ordre alphabétique) de tous les termes particuliers utilisés dans ce travail accompagnés de leur définition respective. L'utilisation de ces termes vise à clarifier au maximum les explications en éliminant toute ambiguïté.

Capacité de transport

Nous parlons de capacité de transport (ou simplement de capacité) lorsqu'il est question de débits utilisés par l'équipement électronique. Dans notre cas, nous parlerons de la capacité OC-48 et de la capacité OC-192.

Chemin

Il s'agit d'un ensemble de liens consécutifs sans boucle (c'est-à-dire que les liens doivent former un chemin dans le sens classique du terme) qui relient un nœud source à un nœud destination (qui est différent du nœud source). Nous appelons C l'ensemble de tous les chemins qui existent dans le réseau. De cette façon, $C_{sd} = \{c_{sd}^1, c_{sd}^2, \ldots c_{sd}^{|C_{sd}|}\}$ correspond à l'ensemble de tous les chemins qui relient le nœud source v_s au nœud destination v_d.

Chemin physique

Il s'agit d'un chemin auquel est associé une longueur d'onde. Ce chemin physique aura pour objectif d'identifier exactement par où passe un flot. Un chemin physique peut être parsemé d'*interruptions* (voir définition plus loin).

Débit

Un débit est toujours exprimé comme un multiple entier du débit de base OC-1. Il s'agit, d'une certaine façon, de la quantité d'information par seconde qui doit être acheminée d'une source vers une destination.

Demande

La demande k_{sd} possède une source (v_s), une destination (v_d) ainsi qu'un débit demandé. Le débit demandé correspond à la somme de toutes les requêtes qui partagent ce même couple (source, destination). Ainsi, il existe seulement au maximum une demande qui relie une source à une destination.

Fibre

Une fibre est, physiquement parlant, la fibre optique elle-même. Elle relie deux nœuds et elle est dirigée. Sur le graphe dirigé du réseau, chaque fibre est représentée par un arc. Pour le présent projet, le terme *fibre* est un synonyme du terme *lien*. Cependant, nous préférons le terme *fibre* au terme *lien* lorsque nous discutons des aspects plus physiques du problème.

Flot

Il s'agit d'un chemin physique auquel est associé un débit. Chaque demande est séparée en un ensemble de flots qui partagent tous la même origine et la même destination et dont la somme des débits doit être égale au débit de la demande qui relie cette source à cette destination.

Interruption

Une interruption survient sur un flot, à un nœud particulier. Plus précisément, cela signifie que le signal optique d'un flot est interrompu à un nœud intermédiaire du chemin (un nœud qui se trouve entre la source et la destination) en passant par une carte de transport. Le signal passe alors du domaine optique vers le domaine électrique, pour ensuite revenir de l'électrique vers l'optique.

Lien

Le terme *lien* est synonyme du terme *fibre*. Cependant, nous préférons le terme *lien* au terme *fibre* lorsque nous discutons des aspects plus théoriques du problème.

Requête

Il s'agit en fait d'une requête élémentaire inscrite dans les matrices de trafic. Une requête doit avoir un nœud source (v_s) et un nœud destination (v_d) qui est toujours différent de la source. Le débit de cette requête est OC-t où $t \in T$ et où $T = \{1, 3, 12, 24, 48\}$. Il peut bien sûr y avoir plusieurs requêtes qui vont de la même source à la même destination.

Saut optique (hop)

Le concept de saut optique est relatif aux flots. En fait, il s'agit d'une partie d'un flot, c'est-à-dire une section du flot qui débute à un port de sortie et qui se termine à un port d'entrée sans subir d'interruptions. Un flot possède, au minimum, un saut optique si le signal du flot n'est jamais interrompu. Un flot possède toujours exactement un saut optique de plus que le nombre d'interruptions.

Segment

Il s'agit d'un chemin optique qui débute à un port de sortie d'une carte de transport MSPP et qui se termine à un port d'entrée plus loin sur une carte d'un autre nœud et ce, en restant toujours dans le domaine optique ; c'est-à-dire sans passer par d'autres cartes de transport (et donc être converti en signal électrique) au cours du chemin. Il s'agit en quelque sorte d'un saut optique sauf qu'il n'est pas associé à un flot particulier : plusieurs flots peuvent partager le même segment. Lorsque le concept de segment est introduit, cela fait référence à la capacité de transport, au chemin et la longueur d'onde qui y est associé.

Signal optique

Par signal optique, nous entendons l'onde lumineuse elle-même.

Solution

Une solution est entièrement définie par un groupage, un routage et une affectation de longueurs d'onde où toutes les requêtes sont satisfaites. Ainsi, de chaque solution nous pouvons en déduire le coût et la réalisabilité.

2.5 Les hypothèses

Pour encadrer la recherche, nous avons posé plusieurs hypothèses. Ces hypothèses ont pour but de définir exactement le contexte du problème que nous voulons résoudre dans le cadre de ce projet. Voici donc la liste des énoncés de toutes les hypothèses de travail, les hypothèses concernant les données du problème, ainsi que les hypothèses simplificatrices.

2.5.1 Les hypothèses de travail

Cas statique

Nous nous intéressons ici uniquement au cas statique, c'est-à-dire que la matrice de trafic est donnée comme entrée au problème et qu'elle ne changera jamais. De plus, nous prenons toujours pour acquis qu'il n'y a aucun équipement initial lorsque nous voulons trouver une solution.

Satisfaire une demande

Pour satisfaire une demande, il faut affecter un certain nombre de flots qui possèdent tous la même source et destination que la demande et dont la somme des débits équivaut au débit total de la demande. Il faut que toute la bande passante demandée soit affectée. Nous supposerons également qu'il y a assez d'espace dans le réseau pour supporter la totalité de la matrice des demandes.

Les capacités de transport

Dans le cadre de ce projet, nous considérons deux capacités de transport : OC-48 et OC-192. Notons qu'il existe aussi des équipements permettant de faire circuler une capacité de transport d'OC-768 mais cette technologie est aujourd'hui encore trop coûteuse pour permettre une implantation rentable à grande échelle.

Qualité des nœuds

Tous les nœuds du réseau sont définis par les liens qui leurs sont adjacents et l'équipement (les cartes MSPP) qui y est installé. Ainsi, il n'y a pas de façon d'indiquer, de manière explicite, qu'un ou plusieurs nœuds doivent agir d'une façon particulière (comme des nœuds de service par exemple).

Flots bifurqués

Nous travaillons avec des flots bifurqués, qui s'appliquent par exemple dans le contexte des protocoles LCAS ou VCAT (voir la référence [8] pour plus de détails sur le fonctionnement de ces nouveaux protocoles). Cela signifie qu'une requête peut être découpée en plusieurs flots qui voyagent indépendamment les uns des autres (sur des chemins physiques différents). Pour cette raison, lors de la résolution du problème, une fois toutes les demandes k_{sd} obtenues à partir des requêtes des matrices de trafic, nous ne tiendrons plus compte du détail des requêtes, nous travaillerons seulement avec les demandes. Nous pouvons nous permettre cette simplification puisque nous pouvons séparer une demande en autant de flots que nous le voulons sans avoir à se préoccuper de quelles requêtes appartiennent à quels flots car les requêtes peuvent être divisées sans que cela ne pose problème. Ainsi, nous pouvons toujours traiter une demande ayant un débit total d'OC-t comme étant t demandes OC-1.

Solution incomplète

Pour qu'une solution soit bonne, 100% des demandes doivent être acceptées et donc affectées dans la solution. Une solution où il existe au moins une demande dont la bande passante demandée n'a pas été totalement affectée n'est pas une solution, c'est une solution incomplète.

2.5.2 Les hypothèses concernant les données du problème

Les liens du réseau

S'il y a un lien qui relie le nœud v_s au nœud v_d, cela n'implique pas nécessairement qu'il existe un lien qui relie le nœud v_d au nœud v_s. De plus, il peut y avoir plus d'un lien qui relie le nœud v_s au nœud v_d.

La matrice de trafic

Une requête doit toujours avoir un nœud destination différent du nœud source. Ainsi, la matrice des demandes sera constituée des éléments k_{ij} tel que i et j sont distincts.

Le nombre de longueurs d'onde

Le nombre de longueurs d'onde disponible W est fixé au début, il s'agit d'un paramètre du problème et il est toujours le même pour toutes les fibres du réseau.

2.5.3 Les hypothèses simplificatrices

Les chemins

Comme cela a été indiqué dans la définition du terme *chemin*, un chemin utilisé par un flot ne peut pas boucler ; il ne peut pas non plus passer deux fois par le même nœud.

Continuité de longueurs d'onde

Il s'agit ici d'une hypothèse simplificatrice qui pourrait gêner la construction de bonnes solutions. Lorsqu'un flot est créé pour satisfaire une partie d'une demande, nous lui affectons une longueur d'onde sur laquelle il voyagera pendant tout le trajet de la source à la destination. En théorie, si ce flot possède une (ou plusieurs) interruption, le signal qui passe du domaine optique au domaine électrique pourrait ensuite être régénéré sur une longueur d'onde différente de la longueur d'onde initiale. Cependant, pour simplifier notre problème, nous nous limitons dans le sens où pour chaque interruption, nous imposons que le signal soit régénéré sur la même longueur d'onde. Toutefois, nous verrons plus loin (à la section 3.3) qu'une procédure spéciale de mise au point d'une solution permet, d'une certaine façon, d'éliminer cette contrainte.

Granularité des débits

Nous posons que l'unité de base indivisible d'un débit est d'OC-1 (soit de 51,7 Mb/s). Puisque les débits des matrices de trafic en entrée sont toujours exprimés comme des multiples entiers du débit OC-1 et puisque les débits supportés par les cartes de transport sont aussi des multiples entiers d'OC-1, cela ne pose aucun problème. En effet, pour une solution où les débits de certains flots sont fractionnaires, la solution où tous les

débits sont arrondis à l'entier supérieur est toujours équivalente en termes de coût et de choix de capacité des cartes de transport ; et dans ce cas, le trafic supporté par cette solution ne peut pas être inférieur. Ainsi, les débits sont toujours représentés comme étant des entiers qui correspondent au nombre d'OC-1.

2.6 Revue de littérature

Il n'y a pas d'articles qui traitent le problème GRWA tel que nous le considérons. Toutefois, nous trouvons plusieurs articles qui s'intéressent à une version simplifiée du problème. La majorité des articles qui se rapprochent de notre étude considèrent le problème de groupage, en plus du problème de routage et d'affectation de longueurs d'onde classique. Ainsi, les articles les plus pertinents par rapport au travail qui fait l'objet de ce mémoire seront expliqués et détaillés. Puis nous réserverons un court commentaire pour les autres articles qui sont en rapport avec notre problème.

2.6.1 Article [17]

Le problème de groupage et d'affectation de longueurs d'onde est étudié sur les réseaux en anneau. L'article met tout d'abord l'emphase sur le fait que le groupage est un problème très important puisque s'il est bien traité, cela peut réduire de beaucoup les coûts du réseau (le nombre de cartes de transport, comme dans notre problème). Le problème est considéré dans le cas dynamique : ceci signifie qu'une solution initiale est construite (issue de la matrice de trafic originale) et qu'il faut trouver une autre solution issue d'une nouvelle matrice de trafic en prenant pour acquis que les équipements de la première solution vont rester en place. Le problème est considéré sous deux aspects : le « *best fit case* » et le « *full*

fit case ». Pour le « *best fit case* », l'objectif est de satisfaire au maximum la nouvelle matrice de trafic selon la solution obtenue pour la matrice de trafic originale alors que pour le « *full fit case* », le but est de satisfaire entièrement la nouvelle matrice de trafic en ajoutant le moins d'équipements possible à la solution obtenue pour la matrice de trafic originale. Pour ce faire, une formulation en nombres entiers a été développée ainsi que deux heuristiques de type recherche tabou (une pour le « *best fit case* » et une autre pour le « *full fit case* »).

Pour les résultats, les heuristiques considèrent l'aspect dynamique et utilisent les solutions exactes fournies par la formulation en nombres entiers comme solution de départ. La taille des expérimentations varie et se rend jusqu'à environ 15 longueurs d'onde et une vingtaine de noeuds. Les résultats montrent que les deux heuristiques proposées donnent de bons résultats et, bien qu'un peu plus lentes en termes de temps de calcul, elles donnent de meilleurs résultats que les autres heuristiques issues de publications précédentes.

Le fait de traiter le cas dynamique et de s'intéresser à deux problèmes (le « *best fit case* » et le « *full fit case* ») importants fait ressortir tout l'intérêt de cet article. La formulation en nombres entiers est intéressante, car les réseaux en anneau sont représentés par des « cercles » qui permettent d'isoler les longueurs d'onde et de faciliter le groupage. Cependant, cette approche est limitée en ce sens que la plus petite granularité est définie dès le départ comme étant une fraction de la bande passante maximale permise ; et si cette granularité est trop petite (OC-1 sur OC-192 par exemple), le nombre de cercles à traiter devient beaucoup trop important et la complexité devient vite difficile à gérer. De plus, dans les anneaux, le problème de routage n'existe pas (ou presque si l'anneau est bidirectionnel), ce qui simplifie aussi beaucoup le problème par rapport à celui que nous considérons dans ce mémoire.

2.6.2 Article [13]

Ici les réseaux optiques maillés sont étudiés. Le problème de groupage est considéré en plus du problème de routage et d'affectation de longueurs d'onde traditionnel ce qui rend l'étude plus pratique. Ainsi, pour un réseau, une matrice de trafic statique et des équipements, l'objectif est de maximiser la bande passante affectée à un chemin dans le réseau. Les équipements, qui constituent une entrée du problème, sont des émetteurs et des récepteurs (l'équivalent des ports de sortie et des ports d'entrée). Une approche par formulation de problème en nombres entiers et deux heuristiques (prévues pour les problèmes de plus grande taille) sont proposées pour maximiser l'objectif. Les deux heuristiques sont en fait relativement simples puisque ce sont des heuristiques gloutonnes.

Le nombre de longueurs d'onde est fixé dès le début, comme c'est le cas pour notre étude. À la base, il n'y a pas de contrainte sur le nombre maximal de sauts optiques (*multi-hop*) ; mais l'auteur propose une modification à son modèle pour limiter le nombre de sauts optiques à un seul par connexion (*single-hop*). Il propose aussi une autre amélioration possible au modèle pour le rendre plus réaliste ; cette dernière permet d'attribuer une certaine priorité (représentée par un poids) à chaque connexion pour pouvoir tenir compte de la qualité de service ou encore de la distance à parcourir pour satisfaire la connexion. De plus, le modèle proposé dans l'article ne permet d'identifier qu'un seul débit valable pour toutes les longueurs d'onde (c'est la capacité de transport OC-48 qui a été considérée dans cet article).

L'expérimentation faite avec l'approche exacte est de très petite taille : un réseau de 6 noeuds avec 3 ou 4 longueurs d'onde. Quant aux deux heuristiques, elles ont été testées sur le même réseau avec pratiquement les mêmes contraintes. Les deux se sont révélées assez semblables entre elles et les résultats obtenus sont toujours très proches de ceux obtenus par l'approche exacte. Les résultats sont concluants certes, mais il aurait été

intéressant de voir le comportement de ces différentes approches sur des problèmes de plus grande taille, notamment concernant la taille du réseau utilisé, ceci de manière à pouvoir tirer plus de conclusions.

2.6.3 Article [15]

L'article propose un algorithme multi-objectifs pour résoudre le problème GRWA sur les réseaux optiques maillés. Au total, l'algorithme permet de résoudre trois objectifs distincts : maximiser le nombre de connexions acceptées, minimiser le coût du réseau (en termes de nombre d'émetteurs et de récepteurs) et minimiser le délai de propagation moyen (c'est-à-dire minimiser le nombre de sauts optiques). L'heuristique développée en est une de type algorithme génétique, ainsi, nous travaillons avec des populations de solutions (des populations de dominants versus des populations de dominés) où il devient relativement facile d'identifier l'ensemble des solutions optimales, c'est-à-dire toutes les solutions qui dominent les autres selon l'un ou l'autre des objectifs considérés. Cette approche est plus profitable que le fait d'associer simplement un poids à chacun des objectifs, car dans ce cas il n'y a qu'une seule solution trouvée à la fin du processus.

L'auteur utilise les mêmes instances que l'article [2] pour tester l'algorithme (un réseau comportant 6 noeuds avec 3 ou 4 longueurs d'onde). Il compare ses résultats avec les deux heuristiques présentées dans ce dernier article. Les résultats appuient le fait que l'algorithme présenté ici est plus évolué que les deux précédents, puisqu'ils sont toujours meilleurs ou équivalents aux résultats précédents. La comparaison est intéressante mais il reste toutefois que les expérimentations sont encore faites sur des problèmes de très petites tailles.

L'aspect multi-objectif présenté dans cet article est très intéressant puisqu'il peut être étendu à autant d'objectifs que nécessaire et qui peuvent être en contradiction. Nous nous

intéressons cependant à un problème légèrement différent de celui présenté ici, puisque nous avons un seul objectif, le nombre maximal de sauts optiques est limité dès le départ et nous exigeons que toutes les connexions soient acceptées (voir section 2.3). Tout de même, cet article met clairement en valeur l'intérêt d'utiliser plusieurs objectifs pour résoudre le problème GRWA puisqu'il n'y a pas que le coût qui doit être considéré lors de la conception d'un réseau optique.

2.6.4 Article [11]

Dans cet article, les auteurs considèrent le problème GRWA sur des réseaux optiques. L'article se distingue principalement par le fait que les trois problèmes (groupage, routage, et affectation de longueurs d'onde) sont considérés ; contrairement à beaucoup d'articles antérieurs où un sous-ensemble des trois problèmes était considéré. Ce sont des réseaux maillés qui sont considérés puisque que c'est ce type de réseaux qui sera en particulier utilisé dans la prochaine génération de réseaux optiques ; ceci a pour cause leur grande flexibilité et leur résistance aux bris.

Ainsi, pour une matrice de trafic statique donnée, l'objectif défini est de minimiser le nombre de transpondeurs (et donc de minimiser le coût). Notons qu'un transpondeur correspond en fait à un port dans notre travail. Une approche par formulation de problème en nombres entiers est proposée pour résoudre le problème. Cependant, lorsque la taille du réseau devient trop importante, il est impossible d'obtenir des résultats en temps raisonnable. C'est pourquoi une décomposition en deux sous-problèmes (le problème GR, puis le problème WA) permet d'obtenir de bonnes solutions (et parfois même, la solution optimale) en temps plus raisonnable comme le montrent les résultats.

Cet article est intéressant puisque le problème posé se rapproche relativement du problème qui fait l'objet de ce mémoire. Toutefois, plusieurs simplifications importantes

ont été effectuées. Il y a tout d'abord le fait qu'une seule capacité supportée par les transpondeurs (uniquement OC-48 ou OC-192 par exemple) est considérée. De plus, les chemins virtuels sont déjà définis et font partie des entrées du programme, alors que dans notre cas, tous les chemins virtuels sont considérés.

La section décrivant les expérimentations est un peu décevante, plus de détails auraient été appréciés. Remarquons toutefois que la taille des réseaux est comparable à ceux que nous étudions, et que la formulation exacte parvient à résoudre en temps raisonnable des réseaux de 12 nœuds. Toutefois, plusieurs paramètres importants (tels le nombre de longueurs d'onde utilisées) étaient absents de la description des expérimentations.

2.6.5 Autres articles

Il y a en fait plusieurs autres articles ([1], [3], [4] et [9]) qui présentent des travaux semblables. Toutefois, nous ne les survolerons que très brièvement puisque ce que nous devons retenir des autres articles reste dans le même esprit que ce qui a été mentionné précédemment concernant les articles détaillés aux sections 2.6.1 à 2.6.4. Notons aussi qu'en moyenne les instances présentées dans ces articles sont de petite taille, comparable aux instances des articles précédents.

Ainsi, l'article [4] propose un modèle exact et un modèle approximatif. Une analyse des performances est ensuite effectuée pour valider le modèle approximatif. L'article [9] propose une approche exacte au problème de groupage où ce dernier n'est considéré que sur certaines longueurs d'onde, ceci dans le but de simplifier les équations tout en considérant le groupage lorsque cela devient critique. Dans l'article [1], des réseaux en anneaux sont étudiés où certains nœuds offrent la possibilité de changer de longueur d'onde. Ainsi, les sauts optiques de chaque connexion peuvent être sur des longueurs

d'onde différentes, ce qui est contraire à l'hypothèse formulée à ce sujet dans la majorité des articles. Notons toutefois que bien que le problème qui fait l'objet de ce mémoire puisse aussi permettre aux sauts optiques d'être sur différentes longueurs d'onde, il nous sera difficile de s'inspirer des résultats de cet article dû à la grande différence entre les problèmes étudiés. Puis finalement, l'heuristique développée dans l'article [3] pour résoudre le problème du GRWA dans un réseau maillé en est une de type « algorithme génétique ». L'algorithme génétique est ensuite comparé à différentes heuristiques issues de publications précédentes.

2.6.6 Discussion sur la littérature

Les articles parcourus offrent un bon aperçu des travaux qui ont été réalisés dans le passé sur le problème de groupage, de routage et d'affectation de longueurs d'onde dans un réseau optique WDM. Quelques aspects (tels la considération de plusieurs objectifs, l'aspect dynamique ou encore la priorité des connexions dans un contexte où toutes les connexions ne peuvent pas être satisfaites) sont introduits dans certains articles alors que nous n'en tiendrons pas compte dans le présent mémoire.

D'un autre côté toutefois, notre travail se démarque par le fait qu'un bon nombre d'éléments ne sont pas considérés dans les articles, il y a entre autres les granularités possibles du trafic, l'utilisation de flots bifurqués ou encore le fait de pouvoir utiliser simultanément deux capacités de transport (OC-48 et OC-192) avec tout ce que cela implique. Notons à ce sujet qu'aucun article ne traite de la possibilité que plus d'une capacité de transport coexistent dans une solution. De plus, l'aspect *multi-hop* (plusieurs sauts optiques), lorsqu'il est introduit dans les articles, implique la plupart du temps que la même longueur d'onde est utilisée pour chaque segment ; alors que dans ce travail, nous considérons aussi la régénération sur des longueurs d'onde différentes. Finalement, nous

ne trouvons aucun article dans lequel un aspect important de la couche physique qu'est la compensation est introduit. Le fait que nous allons résoudre le problème du GRWA en tenant compte de la compensation constitue une voie de recherche encore inexplorée dans ce domaine.

Nous remarquons aussi qu'il existe plusieurs formulations en nombres entiers pour ce problème ; cependant, dû à la trop grande complexité du problème, la taille des instances résolues par ces méthodes n'ont pratiquement rien à voir avec les réseaux réels tellement elles sont simples et de petite envergure. Remarquons cependant qu'une formulation exacte peut s'avérer utile pour fournir une borne inférieure de bonne qualité. Citons à titre d'exemple une thèse de doctorat en cours [12]. Ainsi, les articles démontrent que si nous voulons trouver une solution au problème du GRWA sur un réseau le moindrement grand, en tenant compte de plusieurs facteurs, alors nous devons développer une heuristique. Les heuristiques présentées dans les articles permettent de résoudre des problèmes de taille assez grande. Toutefois, la taille des expérimentations que nous exécuterons dans le cadre de ce mémoire est d'un ordre de grandeur supérieur à la majorité des expérimentations présentées dans les articles.

Chapitre 3 – Solutionner le problème

Dans ce mémoire, nous expliquons la méthode que nous avons utilisée pour résoudre notre problème. L'heuristique de type « recherche tabou » que nous avons développée est présentée plus loin au chapitre 5. De façon globale, cette heuristique permet d'explorer un certain nombre de solutions et retiendra celle qui a le coût minimum. Rappelons ici qu'une solution est définie par un groupage, un routage et une affectation de longueurs d'onde pour un réseau et un trafic donné (voir la définition à la section 2.4). Ainsi, avant d'entrer dans les détails de l'heuristique, il convient d'expliquer en détail les éléments qui font partie intégrante des solutions et les caractéristiques importantes qu'il faut en déduire.

Ce chapitre présente donc tous ces éléments. Nous expliquons comment les solutions sont construites et ce qui les caractérise. Nous discutons aussi des éléments qui vont permettre de déduire le coût et la réalisabilité d'une solution. Une procédure d'optimisation des solutions est aussi proposée à la section 3.3.

3.1 Caractéristiques des solutions

Comme il en a été mention précédemment, une solution est entièrement définie par un groupage, un routage et une affectation de longueurs d'onde où toutes les requêtes sont satisfaites. Les sections 3.1.1, 3.1.2 et 3.1.3 présentent les détails de ce qui constitue les différents aspects d'une solution.

3.1.1 Les flots

La définition de tous les flots d'une solution est ce qui constitue le groupage, le routage et l'affectation de longueurs d'onde. De manière à illustrer ce concept, considérons l'exemple de solution qui satisfait un trafic donné sur le réseau présenté à la Figure 2 ci-dessous.

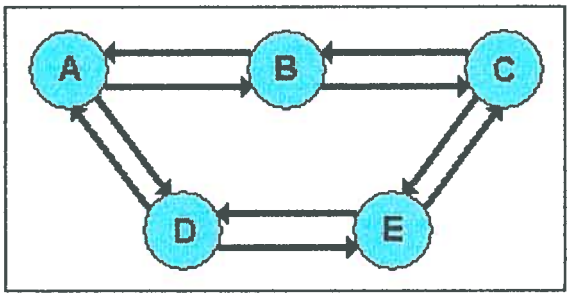

Figure 2 - Exemple de réseau

Supposons que nous ayons 18 requêtes OC-12 débutant toutes au nœud A et se dirigeant toutes vers le nœud C (ceci équivaut donc à 216 requêtes de débit OC-1 puisque nous utilisons des flots bifurqués). Nous poserons donc qu'il y a une seule demande (la demande k_{AC}) qui relie le nœud A au nœud C et qui demande de transporter l'équivalent d'un OC-216.

Nous constatons ici que nous avons 2 chemins qui relient le nœud A au nœud C :

$$C_{AC} = \{c_{AC}^1, c_{AC}^2\}$$

avec $c_{AC}^1 = \{e_{AB}, e_{BC}\}$ et $c_{AC}^2 = \{e_{AD}, e_{DE}, e_{EC}\}$.

Le groupage, le routage et l'affectation de longueurs d'onde se traduisent par l'intermédiaire des flots. Ainsi, une solution proposée pour satisfaire le trafic pourrait être une solution constituée des deux flots décrits ci-dessous.

Flot 1 Départ du nœud A vers le nœud C en passant par c_{AC}^{1}, OC-192 sur λ_1.

Flot 2 Départ du nœud A vers le nœud C en passant par c_{AC}^{2}, OC-24 sur λ_1.

3.1.2 Les capacités

Pour une solution donnée, nous trouvons sur chaque longueur d'onde de chaque lien un certain débit qui y transite. Ce débit représente la somme des débits des requêtes qui voyagent sur ce lien, et qui y empruntent cette longueur d'onde.

Pour chaque longueur d'onde de chaque lien, nous associerons toujours une des trois capacités suivantes : OC-0, OC-48 ou OC-192. La capacité OC-0 ne peut être allouée que si aucun débit ne transite par cette longueur d'onde sur ce lien (soit si la longueur d'onde sur ce lien est inutilisée). La capacité OC-48 peut être allouée seulement si le débit total ne dépasse pas OC-48. Sinon, la capacité affectée à la longueur d'onde du lien doit être la capacité maximale, c'est-à-dire OC-192. Ainsi, dans la solution de l'exemple précédent, la longueur d'onde λ_1 sur les liens $\{e_{AB}, e_{BC}\}$ a une capacité OC-192 alors que la longueur d'onde λ_1 sur les liens $\{e_{AD}, e_{DE}, e_{EC}\}$ peut avoir une capacité OC-48 ou OC-192. Notons aussi que sur une même fibre optique, plusieurs longueurs d'onde qui ont des capacités différentes peuvent voyager ensemble sans problème.

Dans une fibre optique, une longueur d'onde ne peut pas transporter un débit plus élevé que OC-192. Si une solution possède au moins une longueur d'onde sur un lien où le

débit transité est supérieur à OC-192, alors nous dirons de la solution qu'elle est irréalisable.

3.1.3 Les interruptions, les sauts optiques et les segments

Soit le réseau illustré à la Figure 3.

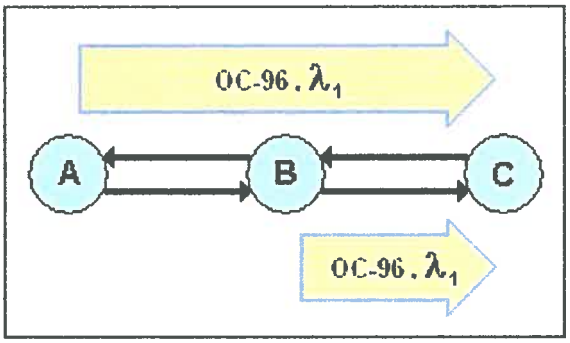

Figure 3 - Exemple de solution # 1

La solution qui y est présentée possède deux flots tels que décrits ci-dessous.

Flot 1 Départ du nœud A vers le nœud C, OC-96 sur la longueur d'onde λ_1.

Flot 2 Départ du nœud B vers le nœud C, OC-96 sur la longueur d'onde λ_1.

La solution est correcte, sauf que quelque chose de particulier est arrivé au flot 1 : il y a eu une interruption au nœud B. La raison est qu'avant d'entrer dans le nœud B, la longueur d'onde λ_1 transportait un flot ayant une capacité OC-96. Mais au nœud B, le signal optique de cette longueur doit être modifié pour pouvoir y ajouter l'information transportée par le flot 2. Or, pour réaliser cette opération, le signal optique de la longueur

d'onde λ_1 doit être converti en signal électrique pour pouvoir être traité à l'intérieur du nœud. En sortie, le nœud B produit un signal ayant un débit d'OC-192 (qui occupe toute la bande passante disponible) qui inclut à la fois l'information transmise par le flot 1 et l'information transmise par le flot 2.

Ainsi, chaque fois qu'un tel traitement est nécessaire (ajout, retrait ou modification de l'information transportée par une longueur d'onde), nous devons procéder de la même façon : il faut convertir le signal optique de cette longueur d'onde en un signal électrique qui sera ainsi traité à l'intérieur du nœud. Nous dirons que si, pour un flot qui voyage sur une longueur d'onde, le signal optique du flot doit être traité à l'intérieur d'un nœud qui n'est pas le nœud source ou le nœud destination, alors ce flot doit subir une *interruption* à ce nœud intermédiaire (comme pour le nœud B de l'exemple précédent).

Remarquons que si un quatrième nœud relié au nœud B venait s'ajouter au réseau et que le flot 2 en était originaire comme cela est illustré à la Figure 4, alors nous aurions eu le même phénomène. Le nœud B doit traiter deux signaux en entrée provenant de deux liens distincts pour les mixer ensemble et envoyer le signal résultant (qui est plus dense en information) sur une même longueur d'onde d'un lien de sortie.

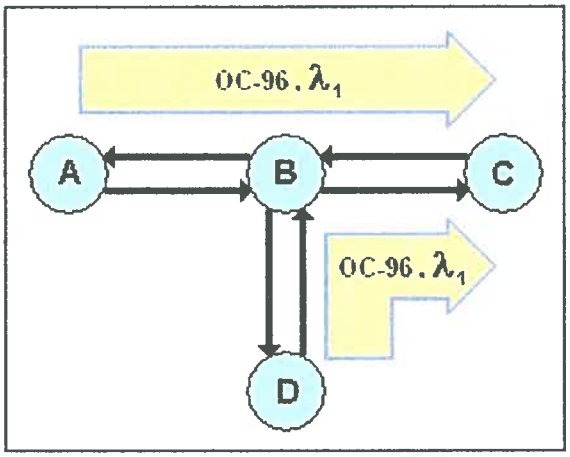

Figure 4 - Exemple de solution # 2

Ceci nous amène à introduire la notion de *segment* : nous appelons ainsi *segment* une partie du chemin physique du flot où le signal n'est jamais interrompu et où le signal est généré (ou régénéré) au premier nœud du segment et reçu par le dernier nœud du segment. Ce segment constitue un *saut optique*. Dans notre dernier exemple (celui de la Figure 4), il y a trois segments, ils sont décrits ci-dessous.

Segment 1 Départ du nœud A vers le nœud B,
OC-96 sur la longueur d'onde λ_1 pour le flot 1.

Segment 2 Départ du nœud D vers le nœud B,
OC-96 sur la longueur d'onde λ_1 pour le flot 2.

Segment 3 Départ du nœud B vers le nœud C,
OC-192 sur la longueur d'onde λ_1 pour les flots 1 et 2.

Une interruption a lieu seulement lorsque plusieurs flots partagent la même longueur d'onde. Ainsi, si nous avions eu 2 longueurs d'onde disponibles et qu'au flot 2 nous avions affecté la longueur d'onde λ_2, alors il n'aurait pas été question de subir une interruption car même si elles voyagent ensemble sur un même lien, les longueurs d'onde fonctionnent de façon totalement indépendante. Le signal de la longueur d'onde λ_1 n'aurait pas eu besoin d'être modifié pour y incorporer le deuxième flot.

Comme dernier exemple, regardons le cas où nous sommes sur un réseau comme celui présenté à la Figure 5. La solution illustrée possède deux flots décrits ci-après.

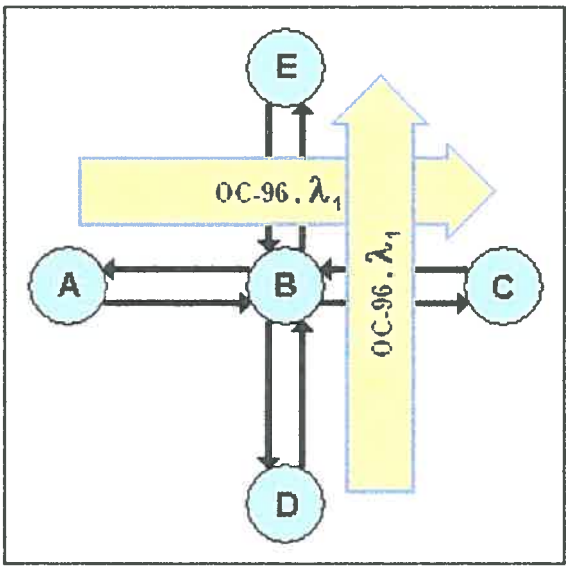

Figure 5 - Exemple de solution # 3

Flot 1 Départ du nœud A vers le nœud C, OC-96 sur la longueur d'onde λ_1.

Flot 2 Départ du nœud D vers le nœud E, OC-96 sur la longueur d'onde λ_1.

Ce que nous voulons faire ressortir dans cet exemple est que même si les deux flots se croisent sur la même longueur d'onde au nœud B, il n'y aura pas d'interruption engendrée puisque les signaux optiques n'ont pas à interagir entre eux entre la source et la destination. Nous dirons que chacun des deux flots a utilisé une « autoroute » (*bypass*) au nœud B, c'est-à-dire que chacun des deux flots a passé du lien entrant vers le lien sortant approprié du nœud B sans s'y être interrompu. La Figure 6 illustre cet exemple.

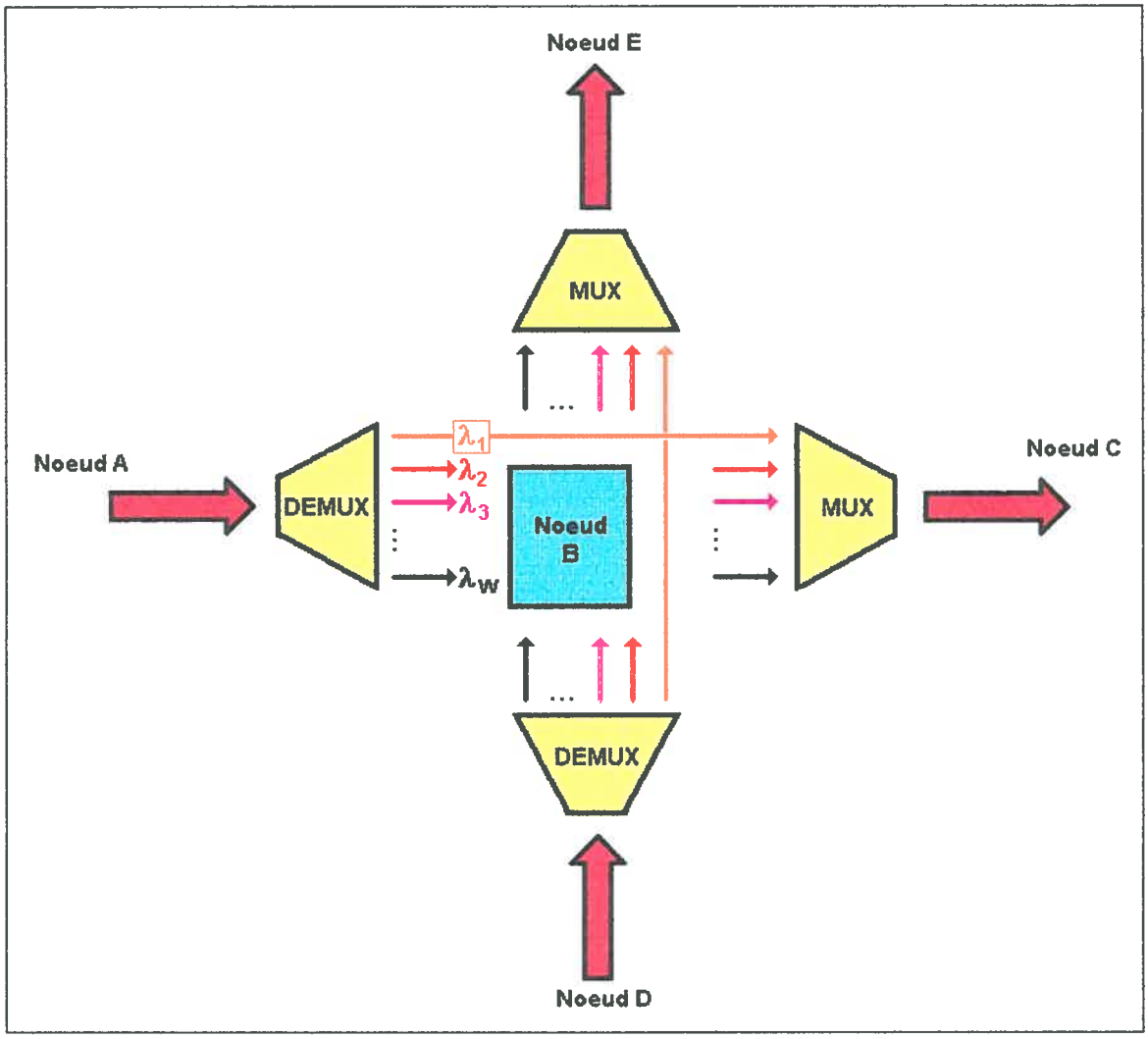

Figure 6 - Les signaux empruntent une autoroute au nœud B

De façon concrète, lorsqu'un flot subit une interruption, cela introduit un certain délai dans la transmission. Ainsi, pour ne pas produire des délais trop longs, une borne est fixée sur le nombre maximal d'interruptions que peut comporter un flot. Si au moins un flot d'une solution subit plus d'interruptions que le nombre maximal fixé, alors la solution est considérée comme irréalisable.

3.2 Les cartes MSPP

Pour pouvoir envoyer et recevoir des requêtes, nous utilisons de l'équipement électronique bien particulier : les cartes MSPP (*MultiService Provisioning Platform*). Lorsqu'un nœud qui traite de l'information doit recevoir ou envoyer un signal optique, c'est par l'intermédiaire des cartes MSPP que cela est possible. Ainsi, chaque nœud du réseau possède un certain nombre de cartes MSPP qui permettent de recevoir et d'envoyer des requêtes.

De cette façon, nous dirons que toutes les cartes MSPP appartiennent à un nœud. Chaque carte possède aussi un débit (OC-48 ou OC-192) ainsi qu'un port d'entrée et un port de sortie. À ces ports nous associons le débit de la carte à laquelle ils appartiennent. Nous associons à chaque port d'entrée une fibre entrante du nœud avec une longueur d'onde et à chaque port de sortie, une fibre sortante du nœud avec la même ou une autre longueur d'onde. Pour une carte donnée, la longueur d'onde à laquelle est associée chacun de ses deux ports peut être différente. La seule restriction est que le débit supporté par le port d'entrée soit le même que celui supporté par le port de sortie. Notons qu'il est aussi possible qu'un port d'une carte soit inutilisé. La Figure 7 illustre une situation où des cartes MSPP à un nœud donné traitent les signaux des différentes longueurs d'onde.

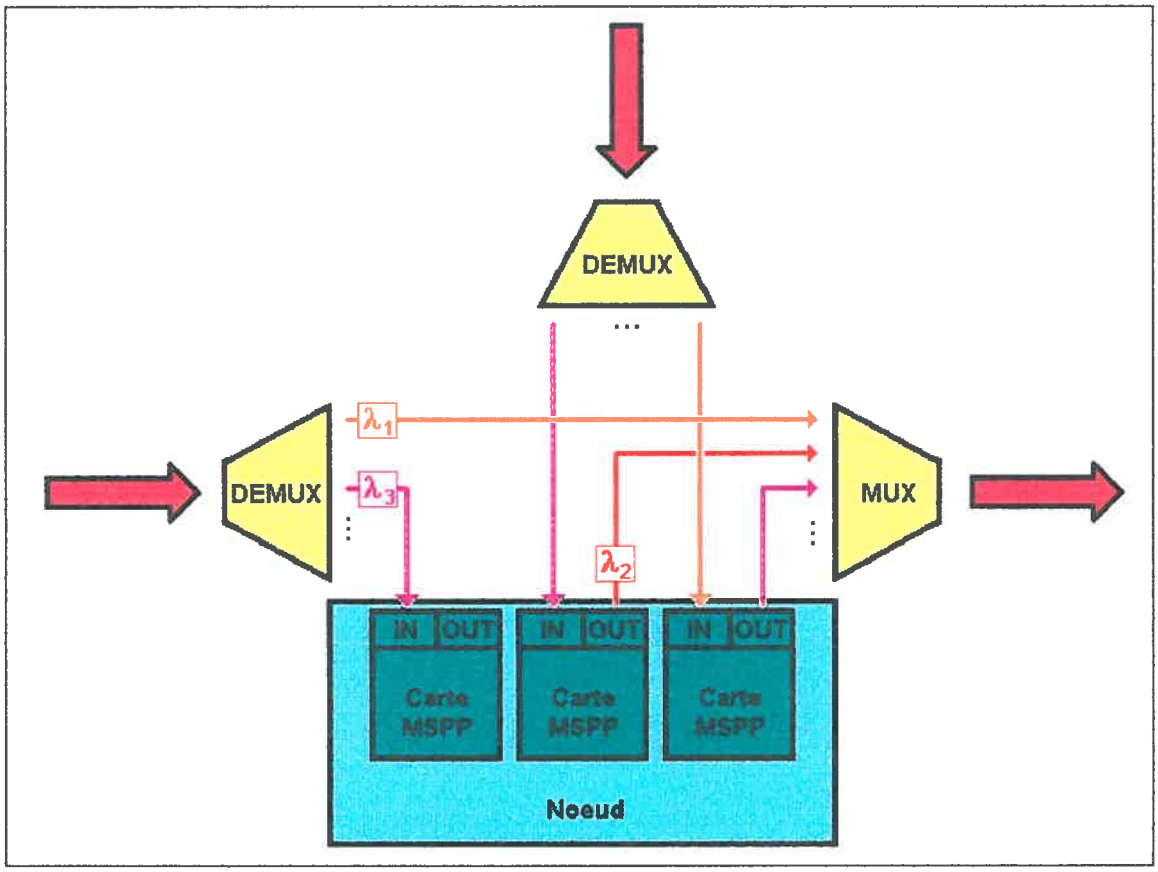

Figure 7 - Les cartes MSPP à un nœud donné

De cette manière, si, au nœud v, une requête doit être ajoutée sur la fibre sortante e et sur la longueur d'onde λ, alors il faut qu'il y ait une carte MSPP à ce nœud dont le port de sortie est connecté sur le lien e et sur la longueur d'onde λ pour pouvoir générer le signal. C'est ce qui se passe pour la longueur d'onde λ_2 à la Figure 7. Le même principe s'applique pour une requête qui doit être reçue au port d'entrée d'une carte MSPP. De plus, la capacité du port de sortie sur lequel le signal d'une requête est généré doit être la même que celle du port d'entrée de l'autre nœud qui recevra la requête.

Il est important de noter ici que les nœuds peuvent accueillir autant de cartes de transport que nécessaire et que chaque carte (selon s'il s'agit d'une carte OC-48 ou OC-192) coûte toujours le même prix peu importe le nœud où elle est installée dans le réseau.

Le présent projet vise à concevoir une métaheuristique qui a pour objectif de minimiser le nombre total de cartes MSPP utilisées dans le réseau pour satisfaire tout le trafic. Le coût d'une solution est en pratique dominé (sauf lorsque la compensation est considérée, voir le chapitre 5) par le coût des cartes MSPP qu'il faut placer à chaque nœud. Pour donner une idée de l'ordre de grandeur, la Table 2 indique les coûts approximatifs des deux types de cartes (notons que ces valeurs sont des paramètres modifiables du problème).

	Coût
Carte OC-48	4 000 $
Carte OC-192	10 000 $

Table 2 - Ordre de grandeur du coût des cartes de transport

Pour une solution donnée, il est important de bien compter le nombre de cartes requises pour supporter cette solution. Puisque sur une carte MSPP il y a deux ports (un port d'entrée et un port de sortie), nous commencerons par nous intéresser au nombre de ports requis. Pour ce faire, il y a plusieurs règles à respecter que nous décrivons ci-dessous.

- S'il y a au moins un flot qui débute au nœud v, sur la longueur d'onde λ et qui emprunte la fibre sortante e, alors il doit y avoir un port de sortie à cet endroit (c'est-à-dire au nœud v, sur la longueur d'onde λ et sur la fibre e). La capacité du port (OC-48 ou OC-192) dépendra du débit du flot ou de la somme des débits des flots s'il y en a plusieurs.

- Le même principe s'applique pour les ports d'entrée. La capacité du port d'entrée (OC-48 ou OC-192) dépendra aussi du débit du flot ou de la somme des débits des flots.

- La fibre entrante e peut transporter plus d'un flot sur la longueur d'onde λ. Il peut aussi arriver qu'au nœud v, les différents flots empruntent des liens sortants distincts. Lorsque ceci survient, il faut pouvoir séparer les informations transportées par la longueur d'onde λ, il faut absolument transformer ces signaux optiques en signaux électriques. Ainsi, il sera nécessaire d'avoir un port d'entrée qui accueillera le signal optique de cette longueur d'onde sur la fibre entrante e. Le signal électrique sera alors traité dans le nœud. Il faudra aussi plusieurs ports de sortie, un pour chacune des fibres sortantes utilisées par les flots ; et ceci, toujours pour la même longueur d'onde (voir l'hypothèse sur la continuité de longueurs d'onde à la section 2.5.3). Notons que le cas s'applique de la même façon si un des flots arrive à destination au nœud v ; dans ce cas toutefois, le flot arrivé à destination ne requière pas de port de sortie.

- En suivant la même logique, le même principe s'applique si des flots originaires de liens d'entrée différents doivent se fusionner pour poursuivre leur chemin sur un même lien de sortie. Ainsi, pour pouvoir multiplexer les informations provenant des différentes sources, il faut procéder de la même façon, c'est-à-dire qu'il faut transformer ces signaux optiques en signaux électriques. Il sera donc nécessaire d'avoir des ports aux liens d'entrée pour accueillir chacun des flots concernés. Il faudra aussi un port de sortie pour produire le signal une fois les flots combinés. Encore une fois, si le nœud source d'un des flots est le nœud v, alors aucun port d'entrée n'est nécessaire pour recevoir le signal électrique de ce flot puisqu'il est généré à même le nœud v.

Pour pouvoir faire le calcul du nombre de ports en respectant toutes les règles mentionnées précédemment, nous nous aiderons d'un tableau que nous appelons le tableau *track*. Au tableau *track* est associé un nœud, une longueur d'onde et une capacité de transport (OC-48 ou OC-192) ; la Table 3 présente ce tableau une fois initialisé à 0. Rappelons qu'à chaque nœud, il y a un certain nombre de liens entrants (noté *in* dans le tableau) et de liens sortants (noté *out* dans le tableau).

		liens sortants				
		lien *out* 1	lien *out* 2	lien *out* 3	...	MSPP
liens entrants	lien *in* 1	0	0	0	...	0
	lien *in* 2	0	0	0	...	0
	lien *in* 3	0	0	0	...	0
	...	...	...	...	...	...
	MSPP	0	0	0	...	×

Table 3 - Le tableau *track* initialisé à 0

Le tableau *track* indique, par des '0' et des '√', tout ce qui traverse le nœud sur la longueur d'onde et pour la capacité de transport qui nous intéresse. Toutes les entrées possibles sont identifiées sur la gauche, la case 'MSPP' signifiant que le flot vient du nœud lui-même (c'est-à-dire qu'il s'agit du nœud source). De la même façon, en haut se retrouvent toutes les sorties possibles, la case 'MSPP' s'appliquant aux flots qui ont pour destination ce nœud. La case qui lie les deux 'MSPP' n'est pas utilisée car une demande ne peut avoir le même nœud comme source et destination. Un '√' dans une case signifie qu'il y a un flot qui provient de l'entrée correspondante et qui se dirige vers la sortie indiquée.

Ensuite, avec ces informations, il faut en déduire les ports qui doivent être présents pour supporter la solution. Ceci sera mémorisé grâce à deux tableaux illustrés à la Table 4 : le premier note avec un '0' ou un '√' l'absence ou la présence de ports d'entrée sur les

liens correspondants et le second fonctionne de la même façon pour les ports de sortie. Les valeurs contenues dans ces deux tableaux sont obtenues à l'aide du tableau *track*.

		port de sortie
liens sortants	lien *out* 1	0
	lien *out* 2	0
	lien *out* 3	0
	...	...

		port d'entrée
liens entrants	lien *in* 1	0
	lien *in* 2	0
	lien *in* 3	0
	...	...

Table 4 - Les tableaux des ports initialisé à 0

3.2.1 Déterminer le nombre de ports

Rappelons que tout ceci s'applique pour un seul nœud, une seule longueur d'onde et une seule capacité de transport (OC-48 ou OC-192). Pour effectuer une mise à jour complète, il faut faire les mêmes étapes pour tous les nœuds, toutes les longueurs d'onde et toutes les capacités.

L'algorithme présenté ici s'exécute en deux temps. Il faut tout d'abord, mettre à jour le tableau *track*, puis ensuite, il faut utiliser ce tableau pour compter correctement le nombre de ports.

Mise à jour du tableau *track*

1. Identifier tous les flots qui débutent au nœud, sur la longueur d'onde sélectionnée et dont la capacité utilisée correspond à celle de ce tableau. Pour chacun, identifier par

un '√' qu'un flot provenant du nœud lui-même (la case MSPP) est envoyé sur le lien sortant correspondant.

2. Faire ensuite la même chose pour les flots dont le nœud étudié est la destination. Indiquer par des '√' dans la colonne intitulée « MSPP » le lien d'entrée sur lequel au moins un flot arrive à destination.

3. Puis identifier tous les flots qui passent, de façon intermédiaire, par ce nœud (qu'il y ait ou pas d'interruptions). Pour chacun de ces flots, cocher à l'aide d'un '√' la case du tableau *track* qui associe la fibre entrante à la fibre sortante correspondante. La Figure 8 présente un exemple où ces trois premières étapes ont été réalisées. Le nœud qui y est étudié possède quatre liens d'entrée et quatre liens de sortie.

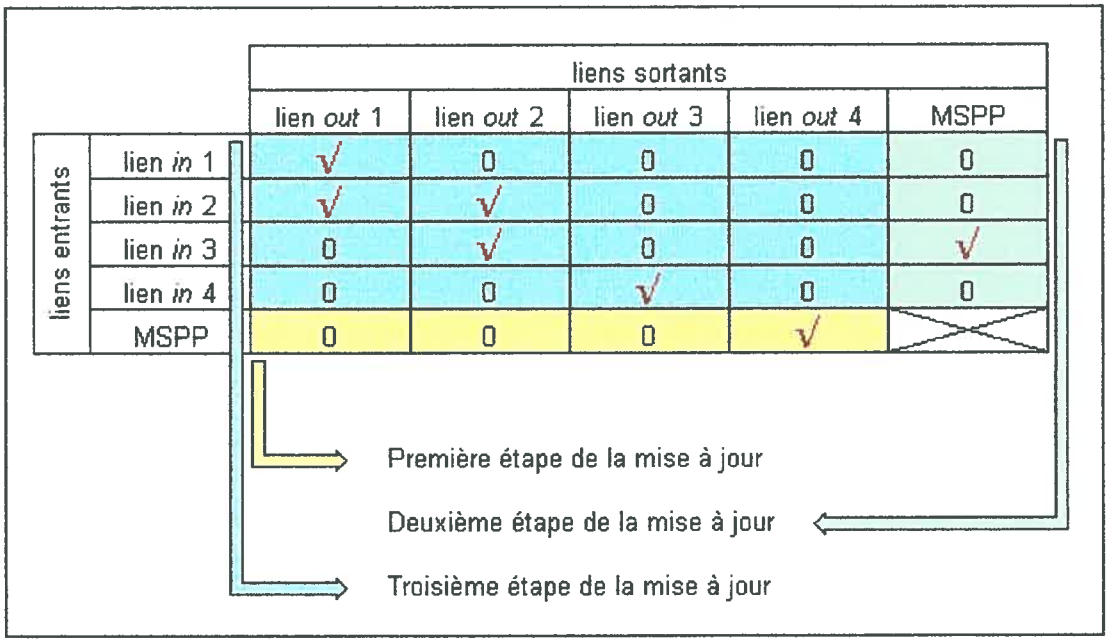

Figure 8 - Étapes de la mise à jour du tableau *track*

Calcul du nombre de ports

4. L'objectif est maintenant de mettre à jour les deux tableaux des ports, ce sont eux qui vont permettre d'identifier les ports requis. La première chose à faire maintenant est d'aller chercher, dans le tableau *track*, les ports d'entrée et les ports de sortie dont la présence est requise pour les flots dont le nœud courant est la source ou la destination. Ceci est identifié dans le tableau *track* par les associations entre les liens (entrant ou sortant) et les cartes MSPP. La Figure 9 illustre ce qu'il faut faire.

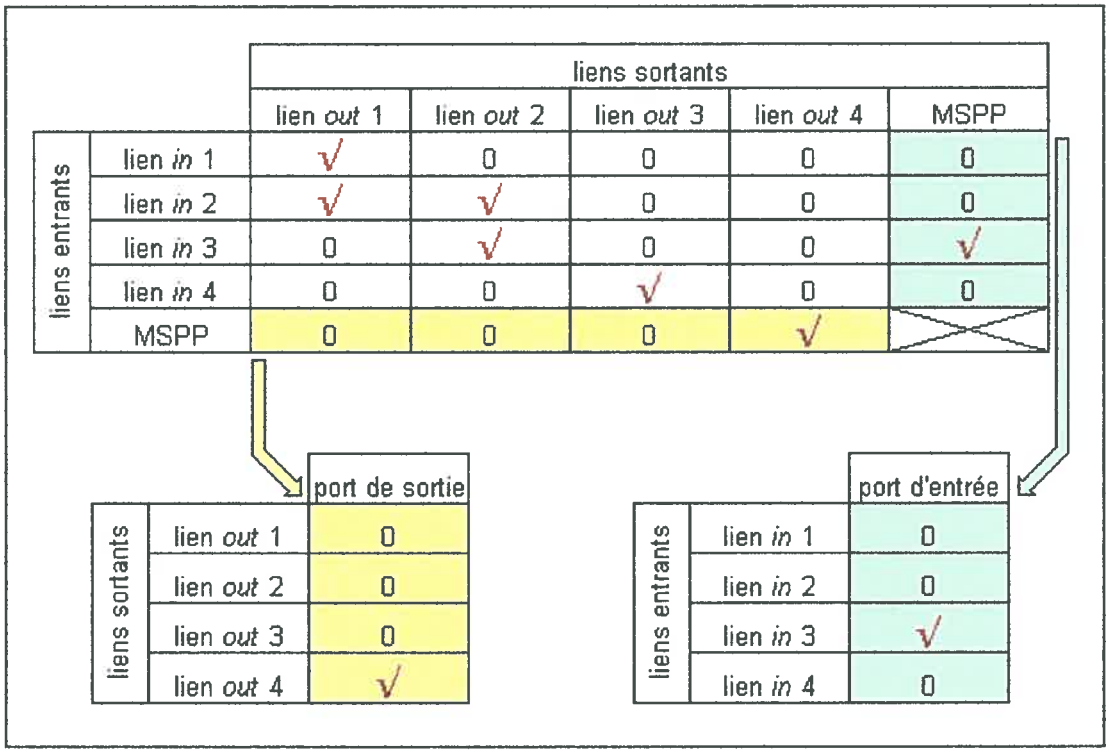

Figure 9 - Les structures de données pour identifier les ports

5. Mais cela n'est pas complet, il faut aussi vérifier le tableau *track* pour s'assurer de considérer toutes les interruptions qui ont lieu au nœud concerné. Par exemple, deux (ou plusieurs) flots peuvent voyager ensemble initialement, puis prendre chacun une direction différente une fois rendus au nœud. Ceci apparaît dans le tableau *track* si, sur une même ligne, il y a plus d'une case marquée d'un '√'. Dans ce cas, le lien entrant devra être équipé d'un port d'entrée pour que le nœud puisse traiter et séparer ces deux flots. De plus, chacun des liens sortants où un '√' apparaît dans cette ligne devra être à son tour équipé d'un port de sortie. Rappelons que la colonne « MSPP » ne correspond à aucun lien sortant puisqu'il s'agit du nœud lui-même. Pour l'exemple que nous suivons depuis le début de ces explications, c'est ce qui arrive aux liens entrants 2 et 3 comme nous le démontre la Figure 10.

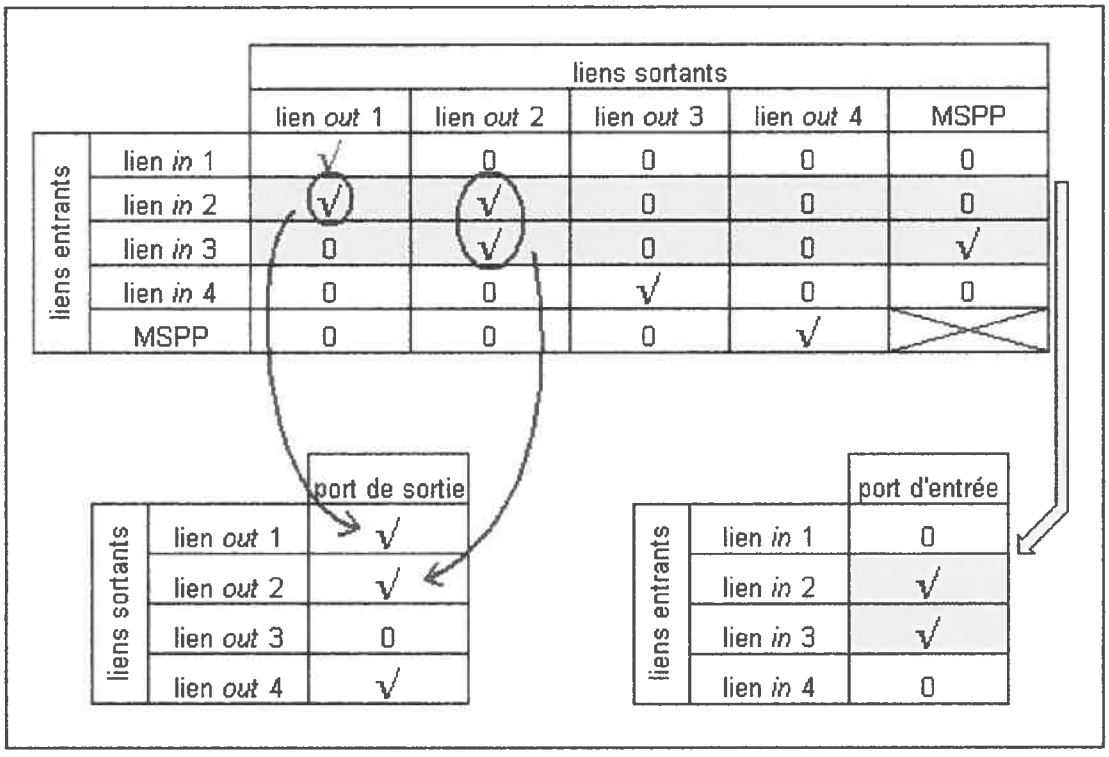

Figure 10 - Illustration de l'étape 5

6. Pour compléter l'inventaire des interruptions, il faut procéder à cette dernière étape : trouver les unions, c'est-à-dire deux flots distincts qui proviennent de deux liens d'entrée différents et qui se combinent pour poursuivre leur voyage ensemble, sur un même lien de sortie. Il suffit d'appliquer la même logique qu'à l'étape 5 lorsque nous cherchions des séparations. Ceci est représenté dans le tableau *track* par une colonne où il y a plus d'une case marquée d'un '√'. Dans un tel cas, le lien sortant correspondant à la colonne identifiée devra être équipé d'un port de sortie pour que le nœud puisse envoyer le nouveau signal optique qui transporte les différents flots. De la même façon, chacun des liens entrants où un '√' apparaît dans la colonne devra à son tour être équipé d'un port d'entrée. La Figure 11 reprend le même exemple où les unions de flots sont identifiées, la mise à jour dans les deux tableaux des ports y est illustrée.

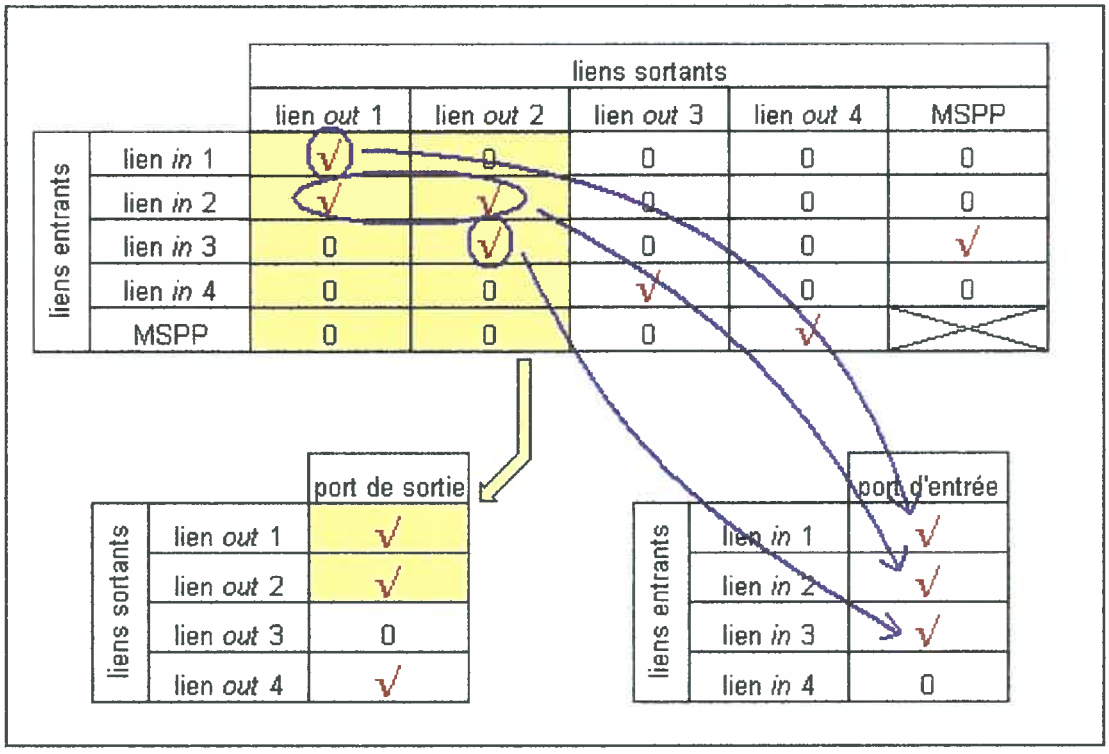

Figure 11 - Illustration de l'étape 6

7. Maintenant, les tableaux des ports d'entrées et des ports de sortie sont complets, chaque '√' correspond à un port. Il ne reste qu'à compiler les résultats pour chaque nœud, chaque longueur d'onde et chaque capacité. Nous obtiendrons de cette façon le nombre total de ports nécessaires.

Voici le pseudo code des étapes 4 à 7 (calcul du nombre de ports) ; nous supposons ici que le tableau *track* est déjà correctement initialisé.

```
nombre_de_ports_in_OC48[pour tous les noeuds] ← 0
nombre_de_ports_out_OC48[pour tous les noeuds] ← 0
nombre_de_ports_in_OC192[pour tous les noeuds] ← 0
nombre_de_ports_out_OC192[pour tous les noeuds] ← 0

Pour chaque noeud, chaque longueur d'onde et chaque capacité

      /* l'étape 4 */
      Nous initialisons les tableaux des ports d'entrée et des ports de sortie
      aux valeurs correspondantes dans le tableau track.

      /* l'étape 5 */
      Pour chaque lien entrant du noeud

            Si il y a plus d'un '√' dans le tableau track sur la ligne
            qui correspond au lien entrant

                  Dans le tableau des ports d'entrée, indiquer par un '√' qu'il
                  doit y avoir un port d'entrée pour le lien entrant
                  correspondant.

                  Marquer aussi, dans le tableau des ports de sortie, tous les
                  liens sortants où les '√' ont été trouvés dans le tableau
                  track.
```

```
/* l'étape 6 */
Pour chaque lien sortant du noeud

      Si il y a plus d'un '√' dans le tableau track sur la colonne
      qui correspond au lien sortant

            Dans le tableau des ports de sortie, indiquer par un '√'
            qu'il doit y avoir un port de sortie pour le lien sortant
            correspondant.

            Marque aussi, dans le tableau des ports d'entrée, tous les
            liens entrants où les '√' ont été trouvés dans le tableau
            track.

/* l'étape 7 */
Si la capacité étudiée est OC48
      nombre_de_ports_in_OC48[pour le noeud concerné] ←
            nombre_de_ports_in_OC48[pour le noeud concerné] +
            nombre de '√' dans le tableau des ports d'entrée
      nombre_de_ports_out_OC48[pour le noeud concerné] ←
            nombre_de_ports_out_OC48[pour le noeud concerné] +
            nombre de '√' dans le tableau des ports de sortie
Sinon
      nombre_de_ports_in_OC192[pour le noeud concerné] ←
            nombre_de_ports_in_OC192[pour le noeud concerné] +
            nombre de '√' dans le tableau des ports d'entrée
      nombre_de_ports_out_OC192[pour le noeud concerné] ←
            nombre_de_ports_out_OC192[pour le noeud concerné] +
            nombre de '√' dans le tableau des ports de sortie
```

3.2.2 L'arrangement des cartes

Une fois tous ces calculs effectués, nous connaissons le nombre de ports, mais ce qui nous intéresse maintenant c'est le nombre de cartes MSPP puisque c'est ce qui dicte le coût d'une solution. Or nous savons qu'une carte OC-48 comporte un port d'entrée OC-48 et un port de sortie OC-48 alors qu'une carte OC-192 comporte un port d'entrée OC-192 et un port de sortie OC-192. Rappelons-nous aussi que sur une carte, le port d'entrée et le port de sortie ne travaillent pas nécessairement sur la même longueur d'onde (mais les deux ports doivent bien évidemment être sur le même nœud). Ainsi, à partir du nombre de ports obtenu à l'aide de l'algorithme décrit à la section précédente, nous proposons trois façons d'en déduire le nombre de cartes de transport.

- **La façon sans arrangement** : la première façon est de différencier les ports OC-48 et les ports OC-192 et d'en déduire directement le nombre de cartes.
- **Forcer l'utilisation de cartes OC-192** : il est possible de forcer l'utilisation de cartes OC-192 dans tous les cas sans se poser de question.
- **Optimiser l'arrangement des cartes** : nous proposons de faire un peu d'optimisation, nous utilisons des cartes OC-192 à la place de cartes OC-48 lorsque cela permet de réduire le coût.

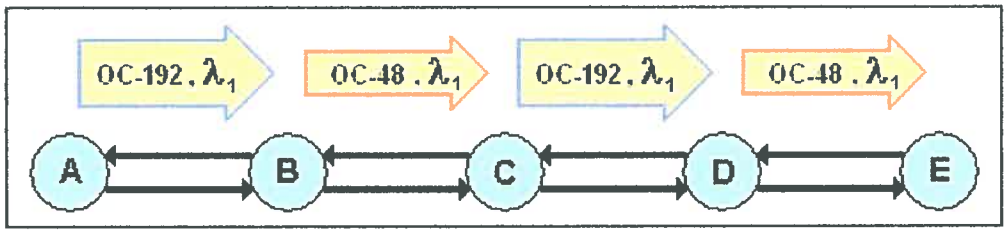

Figure 12 - Exemple de réseau

Dans le but de montrer les différences entre les trois méthodes proposées, regardons l'exemple du réseau de la Figure 12 où transitent quatre flots définis tel que décrits ci-dessous.

Flot 1 Départ du nœud A vers le nœud B, OC-192 sur la longueur d'onde λ_1.
Flot 2 Départ du nœud B vers le nœud C, OC-48 sur la longueur d'onde λ_1.
Flot 3 Départ du nœud C vers le nœud D, OC-192 sur la longueur d'onde λ_1.
Flot 4 Départ du nœud D vers le nœud E, OC-48 sur la longueur d'onde λ_1.

La façon sans arrangement

C'est la façon la plus évidente de déterminer le nombre de cartes : pour un nœud donné et une capacité donnée, il s'agit tout simplement de prendre le maximum entre le nombre de ports en entrée (la somme sur toutes les longueurs d'onde) et le nombre de ports en sortie (la somme sur toutes les longueurs d'onde). Nous pouvons ainsi déterminer le nombre de cartes OC-48 et OC-192 nécessaires pour supporter le trafic dans une solution donnée. Si nous appliquons ce procédé à notre exemple, nous obtenons la solution illustrée à la Table 5.

		les ports en entrée	les ports en sortie	les cartes	coût
noeud A	OC-48	-	-	-	
	OC-192	1 port *IN* inutilisé	1 port *OUT* OC-192	1 carte OC-192	10 000 $
noeud B	OC-48	1 port *IN* inutilisé	1 port *OUT* OC-48	1 carte OC-48	4 000 $
	OC-192	1 port *IN* OC-192	1 port *OUT* inutilisé	1 carte OC-192	10 000 $
noeud C	OC-48	1 port *IN* OC-48	1 port *OUT* inutilisé	1 carte OC-48	4 000 $
	OC-192	1 port *IN* inutilisé	1 port *OUT* OC-192	1 carte OC-192	10 000 $
noeud D	OC-48	1 port *IN* inutilisé	1 port *OUT* OC-48	1 carte OC-48	4 000 $
	OC-192	1 port *IN* OC-192	1 port *OUT* inutilisé	1 carte OC-192	10 000 $
noeud E	OC-48	1 port *IN* OC-48	1 port *OUT* inutilisé	1 carte OC-48	4 000 $
	OC-192	-	-	-	
coût					56 000 $

Table 5 - La solution sans arrangement

Forcer l'utilisation de carte OC-192

Dans ce cas, nous utilisons uniquement des cartes de transport OC-192, autrement dit, nous considérons tous les ports identifiés OC-48 comme étant des ports OC-192. Nous obtenons ainsi la solution décrite dans la Table 6.

		les ports en entrée	les ports en sortie	les cartes	coût
noeud A	OC-48	-	-	-	
	OC-192	1 port *IN* inutilisé	1 port *OUT* OC-192	1 carte OC-192	10 000 $
noeud B	OC-48	-	-	-	
	OC-192	1 port *IN* OC-192	1 port *OUT* OC-192	1 carte OC-192	10 000 $
noeud C	OC-48	-	-	-	
	OC-192	1 port *IN* OC-192	1 port *OUT* OC-192	1 carte OC-192	10 000 $
noeud D	OC-48	-	-	-	
	OC-192	1 port *IN* OC-192	1 port *OUT* OC-192	1 carte OC-192	10 000 $
noeud E	OC-48	-	-	-	
	OC-192	1 port *IN* OC-192	1 port *OUT* inutilisé	1 carte OC-192	10 000 $
coût					50 000 $

Table 6 - La solution où nous forçons l'utilisation de cartes OC-192

Notons ici que nous trouvons un coût inférieur à celui que nous avons trouvé pour la première méthode. Cela peut effectivement arriver mais c'est rarement vrai, il s'agit ici en fait d'un cas particulier. Il est facile de construire un exemple où la première méthode est plus avantageuse que cette méthode-ci. C'est surtout à titre de comparaison des résultats que ce moyen de compter les cartes est utilisé.

Optimiser l'arrangement des cartes

Cette fois-ci, nous reprendrons la première solution (celle obtenue avec la façon sans arrangement) et nous tenterons de minimiser son coût en changeant des ports OC-48 en ports OC-192 uniquement lorsque cela est avantageux.

Suivant cette philosophie, il serait intéressant pour notre exemple de faire voyager le flot 2 sur un signal OC-192 au lieu d'un signal OC-48. Nous utilisons ainsi le port de sortie inutilisé de la carte OC-192 au nœud B ainsi que le port d'entrée inutilisé de la carte OC-192 au nœud C. De cette façon, nous n'avons plus besoin des cartes de transport OC-48 qui se trouvent aux nœuds B et C ; nous économisons donc 8 000 $.

Par contre, il n'est pas avantageux de faire de même pour le flot 4. Il est vrai que nous pourrions ainsi rendre utile le port de sortie inutilisé de la carte OC-192 au nœud D (nous pourrions ainsi économiser 4 000 $ pour la carte OC-48 au nœud D). Mais il faut échanger la carte OC-48 au nœud E pour une carte OC-192, or, cela revient à un coût de : 10 000 $ – 4 000 $ = 6 000 $. Donc d'une part, nous réalisons une économie de 4 000 $ et de l'autre, il faut débourser 6 000 $, ce n'est donc pas avantageux. Le détail de la solution est illustré dans la Table 7.

		les ports en entrée	les ports en sortie	les cartes	coût
noeud A	OC-48	-	-	-	
	OC-192	1 port *IN* inutilisé	1 port *OUT* OC-192	1 carte OC-192	10 000 $
noeud B	OC-48	-	-	-	
	OC-192	1 port *IN* OC-192	1 port *OUT* OC-192	1 carte OC-192	10 000 $
noeud C	OC-48	-	-	-	
	OC-192	1 port *IN* OC-192	1 port *OUT* OC-192	1 carte OC-192	10 000 $
noeud D	OC-48	1 port *IN* inutilisé	1 port *OUT* OC-48	1 carte OC-48	4 000 $
	OC-192	1 port *IN* OC-192	1 port *OUT* inutilisé	1 carte OC-192	10 000 $
noeud E	OC-48	1 port *IN* OC-48	1 port *OUT* inutilisé	1 carte OC-48	4 000 $
	OC-192	-	-	-	
coût					48 000 $

Table 7 - La solution où nous optimisons l'arrangement

Il était facile de trouver un tel arrangement optimal pour cet exemple simple. Mais un algorithme un plus complexe a été mis au point pour trouver un arrangement optimal (ou presque) dans une solution à plus grande échelle.

L'algorithme se déroule en deux passes. Dans un premier temps, nous cherchons tous les segments OC-48 tels que s'ils étaient des segments OC-192, nous pourrions économiser à la fois au nœud source ainsi qu'au nœud destination. Quand un tel segment est identifié, il faut effectuer le changement. Dans l'exemple précédent, le segment qui correspond au flot 2 satisfait ce critère de sélection puisque nous avons économisé au nœud B ainsi qu'au nœud C.

Dans un deuxième temps, nous cherchons tous les segments OC-48 tels que s'ils étaient des segments OC-192, nous pourrions économiser au nœud source sans changer le coût des cartes au nœud destination ; ou encore l'inverse : économiser au nœud destination sans changer le coût des cartes au nœud source. Dans l'exemple précédent, cette situation aurait pu arriver pour le segment qui correspond au flot 4. Supposons en effet que nous avions déjà, au nœud E, un port de sortie OC-48 ainsi qu'un port de sortie OC-192. Alors nous aurions bien sûr pu rendre utile le port de sortie inutilisé de la carte OC-192 au nœud

D (une économie de 4 000 $ au nœud D). Et au nœud E, cela ne changerait rien, car au lieu d'utiliser le port d'entrée OC-48, c'est le port d'entrée OC-192, disponible lui aussi, qui aurait été utilisé.

Le pseudo code de l'algorithme qui permet cette optimisation est présenté ci-dessous. Notons que dans l'algorithme, les gains sont évalués en comparant le nombre de ports d'entrée avec le nombre de ports de sortie pour chacune des deux capacités de transport. Par exemple, si nous savons qu'il y a moins de ports d'entrée que de ports de sortie (pour un certain débit à un nœud donné), alors nous savons que le fait d'ajouter un port d'entrée au nœud ne modifiera pas le coût (aucune carte supplémentaire ne sera requise), nous savons aussi que si un port de sortie est retiré, il y aura une carte économisée.

```
/* La première passe */

Pour chaque segment OC-48

      Si (nombre de ports OUT OC-48 au noeud source >
                  nombre de ports IN OC-48 au noeud source) ET
         (nombre de ports IN OC-192 au noeud source >
                  nombre de ports OUT OC-192 au noeud source) ET
         (nombre de ports OUT OC-48 au noeud destination <
                  nombre de ports IN OC-48 au noeud destination) ET
         (nombre de ports IN OC-192 au noeud destination <
                  nombre de ports OUT OC-192 au noeud destination)

            Alors nous changeons ce segment pour un segment OC-192

            Nous enlevons le port OUT OC-48 du segment au noeud source
            Nous ajoutons un port OUT OC-192 au segment au noeud source
            Nous enlevons le port IN OC-48 du segment au noeud destination
            Nous ajoutons un port IN OC-192 au segment au noeud destination
```

```
/* La deuxième passe */

Pour chaque segment OC-48

      Si ((nombre de ports OUT OC-48 au noeud source >
                  nombre de ports IN OC-48 au noeud source) ET
         (nombre de ports IN OC-192 au noeud source >
                  nombre de ports OUT OC-192 au noeud source) ET
         (nombre de ports IN OC-192 au noeud destination <
                  nombre de ports OUT OC-192 au noeud destination))

         /* si nous trouvons une diminution du coût au noeud source et aucun
            changement du coût au noeud destination */

         OU

         ((nombre de ports OUT OC-48 au noeud destination <
                  nombre de ports IN OC-48 au noeud destination) ET
         (nombre de ports IN OC-192 au noeud destination <
                  nombre de ports OUT OC-192 au noeud destination) ET
         (nombre de ports IN OC-192 au noeud source >
                  nombre de ports OUT OC-192 au noeud source))

         /* ou si nous trouvons une diminution du coût au noeud destination et
            aucun changement du coût au noeud source */

            Alors nous changeons ce segment pour un segment OC-192

            Nous enlevons le port OUT OC-48 du segment au noeud source
            Nous ajoutons un port OUT OC-192 au segment au noeud source
            Nous enlevons le port IN OC-48 du segment au noeud destination
            Nous ajoutons un port IN OC-192 au segment au noeud destination
```

Notons finalement que l'arrangement obtenu par cet algorithme ne sera pas toujours optimal, cela dépend de l'ordre dans lequel les segments sont considérés. L'exemple ci-

dessous illustre un cas particulier où l'ordre des segments influence sur la qualité de l'optimisation effectuée.

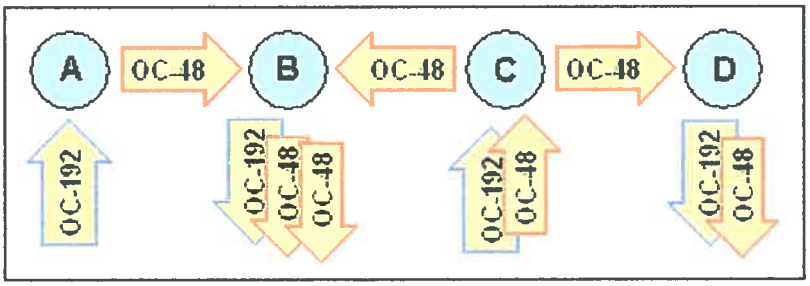

Figure 13 - Exemple de cas problématique

La Figure 13 présente quatre nœuds d'un réseau. Les flèches représentent des flots qui correspondent chacun à un segment. En analysant cette partie de solution, nous trouvons que pour le segment OC-48 qui voyage de A vers B, nous économiserions une carte de transport OC-48 au nœud A si ce segment devenait un segment OC-192. De la même façon, si le segment OC-48 qui voyage de C vers B était un segment OC-192, nous pourrions économiser une carte de transport OC-48 au nœud C. Nous arrivons à la même conclusion si le segment qui voyage de C vers D était un segment OC-192. De cette façon, lors de la deuxième passe de l'algorithme, nous trouvons trois segments dont il serait avantageux de changer la capacité de transport pour économiser une carte OC-48. Cependant, il n'est pas possible d'effectuer ces trois changements simultanément puisqu'il y a des interactions entre eux.

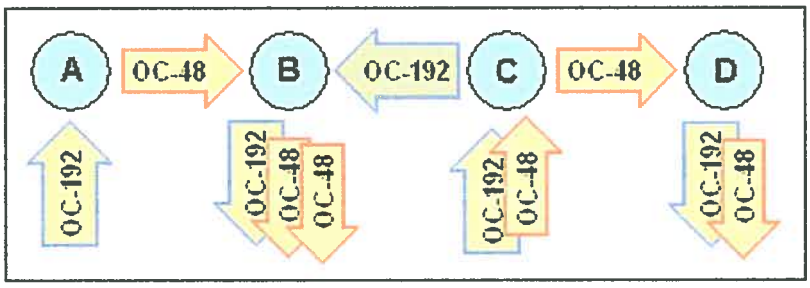

Figure 14 - Un seul changement est effectué

La Figure 14 présente la même solution où nous avons choisi de commencer par changer la capacité de transport du segment qui voyage de C vers B. Nous avons effectivement économisé une carte OC-48 au nœud C ; mais il est à présent impossible de changer la capacité de transport des deux autres segments sans avoir recours à une carte de transport OC-192 supplémentaire. En procédant de cette façon, nous avons économisé seulement une carte OC-48.

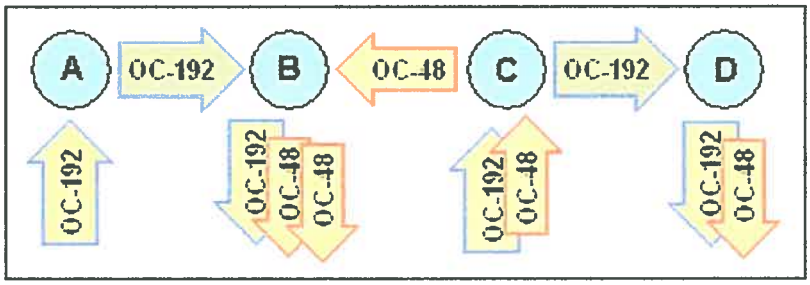

Figure 15 - Deux changements sont effectués

L'alternative présentée à la Figure 15 est plus avantageuse. En commençant par effectuer le changement pour le segment qui voyage de A vers B, nous économisons une carte OC-48. En plus, après cette modification, il est encore possible d'économiser une carte OC-48 en changeant la capacité du segment qui voyage de C vers D pour une capacité OC-192.

3.2.3 Borne inférieure

Pour avoir un indicateur sur la qualité d'une solution, il est intéressant d'avoir une borne inférieure. Cette borne sur le coût nous indique que la solution optimale aura nécessairement un coût supérieur ou égal au coût de la borne inférieure. Avec ce qui été expliqué aux sections précédentes, nous sommes en mesure de calculer une borne inférieure.

Pour ce faire, nous regardons, sur la matrice de trafic, pour chaque nœud le débit total qui doit arriver à destination au nœud ainsi que le débit total dont la source est le nœud. Nous ne faisons aucun routage ou affectation de longueurs d'onde, nous regardons seulement le nombre minimal de ports d'entrée et de sortie qu'il faudrait à chaque nœud pour que le trafic au complet puisse minimalement entrer et sortir de chaque nœud.

En fait, nous remarquons que plus la taille de l'instance (la taille du réseau et la taille du trafic) est importante, plus la borne que nous proposons ici est de mauvaise qualité. La première raison est que cette borne ne tient pas compte du trafic en transit, c'est-à-dire lorsqu'il y a des interruptions. En général, plus l'instance est de grande taille, plus les solutions à ces instances ont un grand nombre d'interruptions. Or, il faut évidemment des cartes de transport pour supporter ces interruptions et la borne présentée ici ne tient pas compte de ce phénomène puisque aucun routage ou groupage n'est effectué. De plus, le groupage peut varier beaucoup selon les différentes directions que peuvent prendre les flots, et dans le calcul de la borne, cet aspect est aussi simplifié. Bref, lorsque nous interprétons la valeur de la borne, il faut garder à l'esprit que la qualité de celle-ci dépend beaucoup de l'instance étudiée.

Voyons maintenant le pseudo code de l'algorithme qui permet d'obtenir cette borne inférieure. Il faut cependant faire attention à une chose, le calcul de la borne repose en

partie sur cette constatation : un port OC-192 coûte environ 2,5 fois plus cher qu'un port OC-48. Ainsi, pour faire passer un débit OC-96 par exemple, il est plus avantageux d'utiliser deux ports OC-48 plutôt que d'utiliser un port OC-192.

```
nombre_de_cartes_OC_48 ← 0
nombre_de_cartes_OC_192 ← 0

Pour chaque noeud du réseau

      total_IN = débit total entrant dans ce noeud selon la matrice de trafic
      total_OUT = débit total sortant de ce noeud selon la matrice de trafic

      nombre_de_ports_IN_OC_192 ← (total_IN + 95) / 192
      nombre_de_ports_OUT_OC_192 ← (total_OUT + 95) / 192

      restant_IN ← total_IN - (nombre_de_ports_IN_OC_192 * 192)
      restant_OUT ← total_OUT - (nombre_de_ports_OUT_OC_192 * 192)

      nombre_de_ports_IN_OC_48 ← (restant_IN + 47) / 48
      nombre_de_ports_OUT_OC_48 ← (restant_OUT + 47) / 48

      /* il faut faire une dernière étape pour minimiser le coût des cartes */

      Si (nombre_de_ports_IN_OC_48 > nombre_de_ports_OUT_OC_48) ET
         (nombre_de_ports_OUT_OC_192 > nombre_de_ports_IN_OC_192)

            nombre_de_ports_IN_OC_192 ← nombre_de_ports_IN_OC_192 + 1
            nombre_de_ports_IN_OC_48 = nombre_de_ports_OUT_OC_48

      Sinon, si (nombre_de_ports_OUT_OC_48 > nombre_de_ports_IN_OC_48) ET
         (nombre_de_ports_IN_OC_192 > nombre_de_ports_OUT_OC_192)

            nombre_de_ports_OUT_OC_192 ← nombre_de_ports_OUT_OC_192 + 1
            nombre_de_ports_OUT_OC_48 ← nombre_de_ports_IN_OC_48
```

```
        Sinon
              /* nous ne pouvons rien faire pour améliorer la situation */

        nombre_de_cartes_OC_48 ← nombre_de_cartes_OC_48 +
              max (nombre_de_ports_IN_OC_48, nombre_de_ports_OUT_OC_48)

        nombre_de_cartes_OC_192 ← nombre_de_cartes_OC_192 +
              max (nombre_de_ports_IN_OC_192, nombre_de_ports_OUT_OC_192)

borne_inférieure ← (nombre_de_cartes_OC_48 × coût d'une carte OC_48) +
                   (nombre_de_cartes_OC_192 × coût d'une carte OC_192)
```

3.3 La non continuité de longueurs d'onde

Il a été mentionné plus tôt que nous avons fait l'hypothèse simplificatrice de la continuité de longueurs d'onde (voir section 2.5.3), c'est-à-dire qu'un flot ne peut voyager que sur une seule longueur d'onde de la source à la destination. Donc, même si ce flot est interrompu en cours de route à un nœud intermédiaire, il devra être régénéré sur la même longueur d'onde ; or, ceci n'est pas toujours la meilleure chose à faire sachant que selon nos hypothèses de départ, il n'est pas plus coûteux de changer de longueur d'onde après une interruption.

Dans le but d'améliorer l'algorithme et de permettre d'intégrer la non continuité de longueurs d'onde aux solutions trouvées par l'heuristique, une méthode a été implémentée sous forme de mise au point de la solution courante. Cela signifie que, pour une solution donnée (une solution normale où il y a toujours continuité de longueurs d'onde), la procédure proposée y modifie l'affectation des chemins et des longueurs d'onde de manière à relaxer cette contrainte tout en gardant comme objectif de minimiser autant que possible le coût de la nouvelle solution.

Du point de vue de l'heuristique, cette procédure peut être appliquée à tout moment à une solution pour en déduire le nouveau coût lorsque la non continuité de longueurs d'onde est considérée. Il est ainsi possible d'améliorer le coût d'une solution en une seule étape. Toutefois, il convient de mentionner que la définition initiale des contraintes de ce projet précisait qu'il y avait continuité de longueurs d'onde. Par la suite, nous avons voulu aussi considérer la possibilité de régénérer le signal en profitant des conversions optiques/électriques/optiques. Malheureusement il était impossible de changer toutes les structures de données du programme, qui avaient été optimisées pour le cas de la continuité des longueurs d'onde. C'est pour cette raison que nous avons conçu une procédure qui recherche une solution optimisée (sans continuité de longueurs d'onde) à partir d'une solution avec continuité de longueurs d'onde plutôt qu'une procédure qui permet de faire évoluer cette solution optimisée au fil des itérations. La procédure expliquée dans cette sous-section constitue ainsi une forme de « post-optimisation » des solutions.

Discutons maintenant de cet algorithme qui permet d'inclure la non continuité de longueurs d'onde à une solution. La première étape d'initialisation consiste à segmenter la solution courante. Ceci signifie que chaque flot est découpé en x flots (où x est le nombre de sauts optiques pour ce flot). Le découpage se fait aux nœuds où il y a interruption. Autrement dit, ce sont les segments de tous les flots qui deviennent les flots avec lesquels nous travaillons dans la nouvelle solution. L'idée ensuite est de reconsidérer la longueur d'onde et le chemin de chacun des nouveaux flots ainsi obtenus, si nous trouvons qu'un changement à ce niveau entraîne une diminution du coût de la nouvelle solution, alors il faut effectuer ce changement. Ce processus est réalisé en quatre phases par l'algorithme que nous proposons. Pour chacune des phases, tous les flots sont considérés et lorsqu'un changement est effectué, il faut revenir à la première phase puisque la nouvelle configuration de la solution peut permettre de nouvelles opportunités. Une brève description des quatre phases est présentée ci-dessous.

1. Pour la première phase, chaque flot est autorisé à changer la longueur d'onde si cela n'engendre aucune interruption supplémentaire.

2. Lors de la deuxième phase, en plus d'être autorisé à changer de longueur d'onde, chaque flot peut changer de chemin. Toutefois, il faut toujours s'assurer qu'aucune interruption supplémentaire ne soit engendrée.

3. À la troisième phase, comme lors de la première, chaque flot est autorisé à changer la longueur d'onde sauf qu'il est permis que des interruptions supplémentaires soient introduites dans la mesure où la contrainte sur le nombre maximal d'interruption est toujours respectée.

4. Finalement, pour la dernière phase, chaque flot peut changer de longueur d'onde et de chemin. Des interruptions peuvent survenir mais la contrainte sur le nombre maximal d'interruptions doit toujours être respectée.

Le détail de cet algorithme est présenté sous forme de pseudo code.

```
/* Initialisation de la solution post-optimisée */

solution post-optimisée ← solution courante segmentée
passe ← 1
flot courant ← premier flot
flot limite ← dernier flot

Tant que passe ≤ 4

      Retirer le flot courant de la solution post-optimisée.

      meilleur_coût ← coût de la solution courante
```

```
Si passe = 1

    Pour chaque longueur d'onde différente de la longueur d'onde
    originale

        Essayer de replacer ce flot sur son chemin initial avec
        cette longueur d'onde.

        Évaluer le nouveau coût.

        Si ce changement a engendré au moins une interruption de
        plus qu'avant ou la nouvelle solution est irréalisable ou
        nouveau coût ≥ meilleur_coût

            Laisser tomber ce changement.
        Sinon,
            Retenir ce changement et mettre à jour meilleur_coût.

Sinon, si passe = 2

    Pour chaque chemin différent du chemin original
        Pour chaque longueur d'onde

            Essayer de replacer ce flot sur ce chemin et cette
            longueur d'onde.

            Évaluer le nouveau coût.

            Si ce changement a engendré au moins une interruption
            de plus qu'avant ou la nouvelle solution est
            irréalisable ou nouveau coût ≥ meilleur_coût

                Laisser tomber ce changement.
            Sinon,
                Retenir ce changement et mettre à jour
                meilleur_coût.
```

```
Sinon, si passe = 3

        Pour chaque longueur d'onde différente de la longueur d'onde
        originale

                Essayer de replacer ce flot sur son chemin initial avec
                cette longueur d'onde.

                Évaluer le nouveau coût.

                Si la nouvelle solution est irréalisable ou nouveau coût ≥
                meilleur_coût

                        Laisser tomber ce changement.
                Sinon,
                        Retenir ce changement et mettre à jour meilleur_coût.

Sinon /* c'est que passe = 4 */

        Pour chaque chemin différent du chemin original
                Pour chaque longueur d'onde

                        Essayer de replacer ce flot sur ce chemin et cette
                        longueur d'onde.

                        Évaluer le nouveau coût.

                        Si la nouvelle solution est irréalisable ou nouveau
                        coût ≥ meilleur_coût

                                Laisser tomber ce changement.
                        Sinon,
                                Retenir ce changement et mettre à jour
                                meilleur_coût.
```

```
Si un mouvement permettant d'améliorer le coût a été retenu

        Effectuer ce changement.  Autrement dit, replacer le flot courant
        avec son nouveau chemin et/ou longueur d'onde selon ce qui a été
        trouvé.

        passe ← 1

        flot limite ← flot courant

Sinon,

        Replacer le flot courant dans la solution post-optimisée.

        Si flot courant = flot limite

                passe ← passe + 1

Le flot courant devient le flot suivant (si c'est le dernier flot, il faut
revenir au premier).

Retourner au début de la boucle principale (tant que passe ≤ 4)
```

Cette stratégie de traiter la non continuité de longueurs d'onde sous forme de mise au point possède l'avantage d'être efficace : en une étape, nous passons d'une solution soumise à la contrainte de continuité de longueurs d'onde à une solution semblable où cette contrainte n'est plus présente. En plus, du point de vue de la fonction objective, cette étape effectue une amélioration de la solution (dans le pire des cas, la nouvelle solution trouvée possède un coût équivalent). Le principal désavantage de cette procédure concerne le temps de calcul. Nous remarquons que pour rester dans des temps raisonnables (quelques heures), il convient de ne pas effectuer cette optimisation trop fréquemment dans l'heuristique.

Chapitre 4 – La compensation

Quand un signal optique traverse une fibre optique, il subit l'effet de la dispersion chromatique. Si la distance parcourue dans la fibre est assez courte, alors l'effet de la dispersion chromatique subie par le signal sera négligeable et la carte de transport qui recevra ce signal pourra discerner les bits '0' et '1' avec un bon taux de succès. Cependant, si la distance parcourue dépasse une certaine longueur critique, la dispersion chromatique devient trop importante et un compensateur doit être installé sur la fibre, à son extrémité destination (il s'agit en fait de post-compensation, pour plus de détails, consulter la référence [7]). Un compensateur compense de x kilomètres le signal transporté par la fibre optique ; le nombre de kilomètres que doit compenser un compensateur peut varier et cette variable influence le coût du compensateur lui-même. Notons toutefois qu'un compensateur installé sur une fibre compense simultanément toutes les longueurs d'onde passant par cette fibre.

De plus, les différents débits ne sont pas tous aussi sensibles à la dispersion chromatique. Un débit d'OC-48 peut tolérer une dispersion chromatique jusqu'à 16 fois supérieure à celle que peut tolérer un débit d'OC-192. De cette façon, nous associons à un compensateur deux caractéristiques : le lien e qui doit être compensé et le nombre de kilomètres de fibre de transport pour lesquels le compensateur doit compenser la dispersion chromatique.

S'il n'est pas interrompu, un signal optique peut traverser plusieurs fibres optiques du réseau avant d'arriver à destination : ceci constitue un segment. Si la distance totale parcourue par le signal optique (bref, la longueur en kilomètres du segment) est sous la limite fixée, alors aucune mesure ne doit être prise, car nous considérons les pertes engendrées par la dispersion chromatique comme étant négligeables. Cependant, si la

limite est dépassée, nous appliquons la règle suivante : une post-compensation doit être ajoutée à l'extrémité destination de chacune des fibres optiques utilisée par le segment (puisqu'il s'agit de post-compensation, la compensation elle-même se fait toujours à la fin d'un lien, tout juste avant d'entrer dans le nœud destination du lien). Notons que la quantité de dispersion chromatique à compenser (cela revient au nombre de kilomètres) est, dans tous les cas, toujours égale à la quantité de dispersion chromatique (donc la longueur en kilomètres) de la fibre sur laquelle est installé le compensateur. Remarquons ici que dans un objectif d'optimisation pur et dur, cela ne serait pas nécessaire. Nous pourrions nous permettre de compenser beaucoup moins, à l'extrémité destination de certaines fibres intermédiaires seulement, mais notre choix est motivé par le fait que cette façon de procéder est relativement répandue.

4.1 Les règles

4.1.1 Règles pour savoir quand et où mettre de la compensation

Tout d'abord, la première chose à faire est d'identifier tous les segments de la solution courante. Ensuite, il faut calculer la longueur (en kilomètres) de chacun de ces segments et nous retiendrons seulement ceux dont la longueur est supérieure ou égale à la longueur limite pour la capacité du signal. Rappelons que la longueur limite diffère selon qu'il s'agit d'un segment OC-48 ou d'un segment OC-192. Il faut ajouter de la post-compensation de x kilomètres à l'extrémité destination de chaque lien faisant partie de chaque segment retenu. Rappelons que le nombre de kilomètres à compenser correspond toujours à la longueur physique du lien.

4.1.2 Règles pour connaître le coût de la compensation

Pour connaître le coût de la compensation dans le réseau au complet, il s'agit dans un premier temps d'identifier les liens nécessitant de la compensation suivant les règles énoncées à la section précédente. Notons que le coût de la compensation possède deux attributs : le coût fixe de base pour compenser un lien, ainsi que le coût de chaque kilomètre à compenser. Donc, pour chaque lien à compenser, le coût se calcule ainsi :

$$coût = (coût\ de\ base) + (longueur\ du\ lien\ en\ km) \times (coût\ par\ km).$$

Ensuite, il s'agit simplement de faire la somme des coûts pour chacun des liens pour obtenir le coût total de la compensation.

4.1.3 Quelques exemples qui illustrent l'application de ces règles

Dans les exemples qui suivent, les longueurs limites sont celles inscrites dans la Table 8.

	Longueur limite
Signal OC-48	500 km
Signal OC-192	80 km

Table 8 - Longueur limite sans compensation selon le débit

Exemple 1

Soit le réseau de la Figure 16.

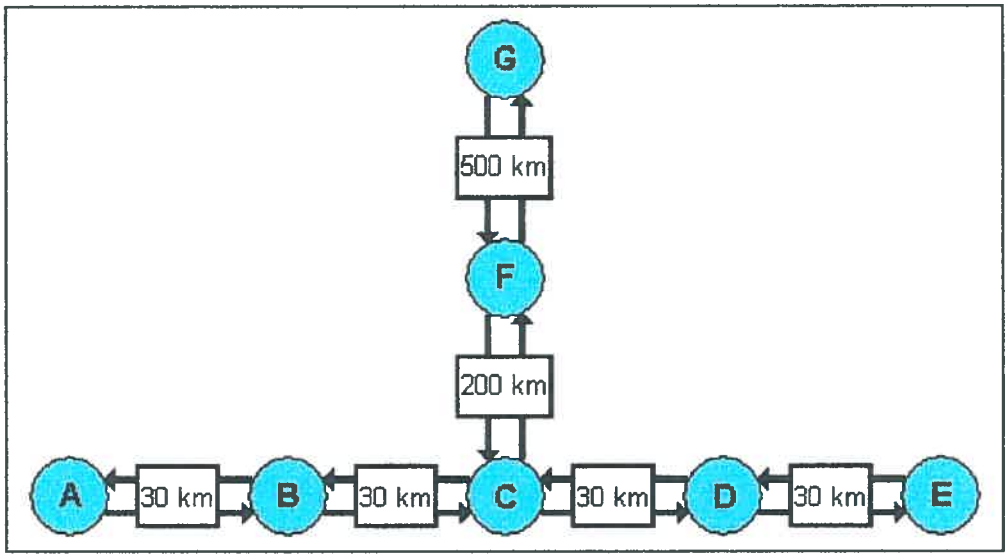

Figure 16 - Exemple de réseau avec compensation

Nous avons deux flots définis tels que décrits ci-dessous.

Flot 1 Départ du nœud A vers le nœud E, OC-96 sur la longueur d'onde λ_1.

Flot 2 Départ du nœud F vers le nœud E, OC-48 sur la longueur d'onde λ_1.

Pour pouvoir transporter ces flots, il faut trois segments.

Segment 1 Départ du nœud A vers le nœud C (60 km),
OC-96 (donc un signal OC-192) sur la longueur d'onde λ_1.

Segment 2 Départ du nœud F vers le nœud C (200 km),
OC-48 (donc un signal OC-48) sur la longueur d'onde λ_1.

Segment 3 Départ du nœud C vers le nœud E (60 km),
OC-144 (donc un signal OC-192) sur la longueur d'onde λ_1.

Dans cet exemple, aucun équipement de compensation n'est nécessaire, car tous les segments sont en dessous de la longueur limite qui leur est permise. Le coût de la compensation est donc nul.

Exemple 2

Soit le réseau de la Figure 16.

Nous avons quatre flots définis tels que décrits ci-dessous.

Flot 1 Départ du nœud A vers le nœud C, OC-192 sur la longueur d'onde λ_1.
Flot 2 Départ du nœud B vers le nœud E, OC-192 sur la longueur d'onde λ_2.
Flot 3 Départ du nœud G vers le nœud D, OC-2 sur la longueur d'onde λ_3.
Flot 4 Départ du nœud F vers le nœud D, OC-48 sur la longueur d'onde λ_3.

Pour pouvoir transporter ces flots, il faut quatre segments.

Segment 1 Départ du nœud A vers le nœud C (60 km),
OC-192 (donc un signal OC-192) sur la longueur d'onde λ_1.

Segment 2 Départ du nœud B vers le nœud E (90 km),
OC-192 (donc un signal OC-192) sur la longueur d'onde λ_2.

Segment 3 Départ du nœud G vers le nœud F (500 km),
OC-2 (donc un signal OC-48) sur la longueur d'onde λ_3.

Segment 4 Départ du nœud F vers le nœud D (230 km),
OC-50 (donc un signal OC-192) sur la longueur d'onde λ_3.

Dans cet exemple, nous avons 3 segments problématiques : les segments 2, 3 et 4 (notons que si nous retirons le flot 3, alors le segment 3 n'existerait plus et le segment 4 ne serait plus problématique, car il pourrait transporter un signal OC-48). Pour résoudre ce problème, il faut placer de la compensation à l'extrémité destination de chaque lien appartenant à chaque segment problématique. Il faudra donc compenser ces liens :

le lien qui relie B vers C (30 km), car ce lien appartient au segment 2;
le lien qui relie C vers D (30 km), car ce lien appartient aux segments 2 et 4;
le lien qui relie D vers E (30 km), car ce lien appartient au segment 2;
le lien qui relie G vers F (500 km), car ce lien appartient au segment 3;
le lien qui relie F vers C (200 km), car ce lien appartient au segment 4.

Maintenant, pour déterminer le coût, nous utiliserons les valeurs de la Table 9.

	Coût
Coût de base	20 000 $
Coût par kilomètre	375 $ / km

Table 9 - Les coûts relatifs à la compensation

Voici donc les coûts relatifs à la compensation pour chaque lien qui doit être compensé :

- le lien qui relie B vers C (30 km), coût : 20 000 $ + 30 km × 375 $ / km = 31 250 $,
- le lien qui relie C vers D (30 km), coût : 20 000 $ + 30 km × 375 $ / km = 31 250 $,
- le lien qui relie D vers E (30 km), coût : 20 000 $ + 30 km × 375 $ / km = 31 250 $,
- le lien qui relie G vers F (500 km), coût : 20 000 $ + 500 km × 375 $ / km = 207 500 $,
- le lien qui relie F vers C (200 km), coût : 20 000 $ + 200 km × 375 $ / km = 95 000 $.

Coût total de la compensation : 396 250 $.

4.1.4 Compensation versus cartes de transport

Il subsiste encore un problème à régler. Pour l'illustrer, voyons tout d'abord un exemple. Nous utiliserons les limites définies dans la Table 8 (la limite pour un signal OC-192 est de 80 km). Le coût d'une carte OC-192 est 10 000 $, comme indiqué à la Table 2, et les coûts relatifs à la compensation sont ceux énoncés à la Table 9.

Soit le réseau de la Figure 17.

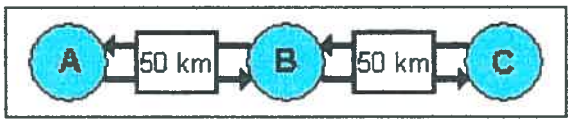

Figure 17 - Exemple de réseau

Sur ce réseau, nous avons un flot qui débute au nœud A et se dirige vers le nœud C, avec un débit OC-192 sur la longueur d'onde λ_1. Si le flot est transporté tel quel, en

ajoutant simplement une carte au nœud A et une autre au nœud C, nous voyons bien que nous avons besoin de compenser le lien qui relie A vers B, ainsi que le lien qui relie B vers C, car 50 km + 50 km > 80 km. Après calcul, nous trouvons que le coût total de la compensation est de 90 000 $. Cependant, il y a une façon plus économique de résoudre ce problème. En effet, il s'agit de faire passer le signal par une carte MSPP au nœud B, créant ainsi une interruption. Ceci aura pour effet de couper le segment qui relie A vers C sur la longueur d'onde λ_1 en deux plus petits segments : un segment qui relie A vers B et un autre qui relie B vers C, tous les deux sur la longueur d'onde λ_1. Des deux segments ainsi formés, aucun ne nécessite de la compensation, car ils sont trop courts (il y a donc une économie de 90 000 $). Toutefois, cela introduit un autre coût : le coût de la carte MSPP (10 000 $) qui fait le relais entre les deux nouveaux segments. Donc au total, en choisissant cette configuration, nous économisons 80 000 $.

Par contre, si nous avions 10 flots qui débutaient tous au nœud A et qui se dirigeaient tous vers le nœud C, avec un débit OC-192 sur les longueurs d'onde λ_1 à λ_{10}, il serait plus économique de laisser la compensation. Le coût avec la compensation serait toujours de 90 000 $, mais si nous choisissions de placer des cartes, alors nous aurions besoin de 10 cartes de transport au nœud B, ce qui coûterait plus cher (100 000 $). Notons qu'en cas d'égalité (par exemple si nous avions 9 des flots décrits précédemment, les deux coûts auraient alors été de 90 000 $), notre choix s'arrête sur la solution avec la compensation pour ne pas introduire d'interruption sans raison.

Le problème à résoudre est donc de déterminer où laisser la compensation et où placer des cartes de transport pour enlever les besoins de compensation, car nous avons toujours comme objectif de minimiser le coût total.

4.2 La stratégie

La stratégie appliquée ici est en fait une forme de mise au point de la solution courante. C'est-à-dire que lorsque nous tenons compte de la compensation, l'algorithme principal de l'heuristique fonctionne de la même façon, sauf qu'à chaque fois qu'une solution est évaluée, une fonction se charge d'incorporer la compensation à la solution courante en y ajoutant soit des compensateurs, soit des cartes de transport, tout ceci toujours dans l'objectif de minimiser le coût. Nous procédons ainsi car deux solutions très semblables peuvent avoir une configuration très différente une fois cette opération effectuée pour tenir compte de la compensation.

4.2.1 L'algorithme principal

La stratégie, qui s'exécute en trois étapes, est décrite brièvement ci-dessous.

1. Tout d'abord, nous prenons la solution telle qu'elle est (sans compensation). Nous plaçons la compensation partout où cela est nécessaire. Par le fait même, nous identifions, pour chaque lien qui nécessite un compensateur, les longueurs d'onde qui transportent des segments trop longs (c'est-à-dire ceux qui sont à l'origine de la nécessité de tenir compte de la compensation pour ce lien).

2. Pour chacun des liens du réseau, nous évaluons, pour chacune de ces longueurs d'onde, combien cela coûterait en termes de cartes de transport pour éliminer la nécessité d'utiliser un compensateur. Il est important de noter ici qu'il est possible qu'un lien ne puisse pas échapper à la nécessité d'avoir un compensateur. C'est le

cas si le lien lui-même est trop long, ou encore si un segment ne peut pas être coupé sans quoi la contrainte de réalisabilité concernant le nombre maximal d'interruptions permises ne sera plus respecté pour au moins un flot.

3. Ainsi, pour tous les liens où il faut compenser mais où cela peut être évité en utilisant des cartes, nous comparons le coût du compensateur au coût alternatif si des cartes de transport étaient utilisées. Il est ensuite possible de trouver le meilleur lien : celui où nous économisons le plus en utilisant des cartes de transport plutôt qu'un compensateur. S'il s'agit réellement d'une économie (si le coût en cartes est inférieur au coût du compensateur), alors nous procédons à ce changement en plaçant les cartes de transport additionnelles aux bons endroits de manière à introduire artificiellement des interruptions aux segments problématiques. Ainsi, pour chaque segment qui doit être interrompu, il s'agit simplement d'ajouter une carte de transport de même capacité que le segment au nœud où doit se produire l'interruption. Les deux ports de cette carte seront utilisés pour que le signal optique du segment y transite.

Le principe ensuite est de répéter cette manœuvre au complet (les trois étapes) tant qu'il est possible trouver au moins un lien où il est avantageux de remplacer un compensateur par des cartes de transport.

Voici donc le pseudo-code de la fonction qui organise la compensation pour une solution donnée.

```
Faire

    /* Nous prenons la solution telle qu'elle est et nous plaçons la
    compensation sur les liens qui en ont besoin selon les règles énoncées à
    la section 4.1 . */
```

Pour chaque lien du réseau

/ pour chaque lien, la variable* ***coût_des_cartes*** *indique le coût si nous voulions placer des cartes de transport pour ne pas avoir à utiliser de compensateur */*

coût_des_cartes ← 0

/ pour chaque lien, la variable* ***coût_de_la_compensation*** *indique le coût si nous voulions installer un compensateur sur ce lien */*

coût_de_la_compensation ← coût de base +
(longueur du lien courant × coût par kilomètres)

Pour chaque longueur d'onde empruntée par au moins un segment sur le lien courant

/ Nous vérifions si le lien seul est trop long */*

Si la longueur du lien ≥ longueur limite correspondant au débit du segment

Nous marquons le lien comme étant un lien où il faut ajouter de la compensation.

Nous marquons le lien comme étant un lien où il est impossible d'enlever de la compensation.

Arrêter ; il faut passer au prochain lien du réseau.

/ Nous vérifions si le lien a besoin d'être compensé du point de vue de la longueur d'onde courante */*

Si longueur du segment ≥ longueur limite correspondant au débit du segment

Nous marquons le lien comme étant un lien où il faut ajouter de la compensation.

Nous vérifions si nous avons besoin d'une ou de deux cartes de transport pour sectionner le segment de façon à éliminer la compensation. */* *Voir la section 4.2.2 sur l'ajout de carte dans un segment* */*

Soit **nb_cartes** le nombre de cartes nécessaires.

Notons que nous induirons **nb_cartes** interruptions supplémentaires aux flots qui utilisent le segment. Il faudra donc s'assurer que pour tous ces flots, la contrainte du nombre maximal d'interruptions soit toujours respectée si nous introduisons **nb_cartes** interruptions à chacun de ces flots.

Si nous trouvons que la solution deviendrait irréalisable (par rapport à la contrainte du nombre maximal d'interruptions permises)

Marquer le lien comme étant un lien où il est impossible d'enlever de la compensation.

Arrêter ; il faut passer au prochain lien.

Sinon,

coût_des_cartes ←
coût_des_cartes +
nb_cartes × coût d'une carte de transport pour le débit correspondant au segment.

Puis nous sélectionnons, parmi les liens où il est possible d'enlever la compensation et tels que **coût_de_la_compensation** > **coût_des_cartes**, le lien pour lequel **(coût_de_la_compensation** - **coût_des_cartes)** est le plus élevé.

```
        ajouter des cartes de manière à enlever la compensation sur le lien
        sélectionné. */

        Si un lien a été sélectionné

                Pour chaque longueur d'onde empruntée par au moins un segment sur
                le lien courant

                        Si longueur du segment ≥ longueur limite correspondant au
                        débit du segment

                                Nous vérifions si nous avons besoin d'une ou de deux
                                cartes de transport pour sectionner le segment de façon
                                à éliminer la compensation.  /* Voir la section 4.2.2
                                sur l'ajout de cartes dans un segment */

                                Nous plaçons nb_cartes cartes aux meilleurs endroits
                                possibles à l'intérieur du segment de manière à le
                                sectionner en segments plus petits.  /* Voir la
                                section 4.2.2 sur l'ajout de carte dans un segment
                                pour la description plus détaillé de cet algorithme */

Tant qu'il y a au moins un lien sur lequel nous pouvons gagner quelque chose
```

4.2.2 Ajout de cartes de transport dans un segment

L'ajout de cartes de transport pour sectionner un segment n'est pas une tâche très simple. Un segment peut être sectionné de multiples façons et il faut toujours choisir la meilleure façon selon le lien courant pour lequel nous voulons enlever la compensation.

Déterminer le nombre de cartes

Si nous voulons enlever la compensation sur un lien en sectionnant un segment, la première question à laquelle nous devons répondre est celle-ci : combien de cartes de transport devons-nous ajouter au minimum pour ne plus avoir à considérer la compensation sur ce lien ?

Regardons tout d'abord l'exemple illustré à la Figure 18.

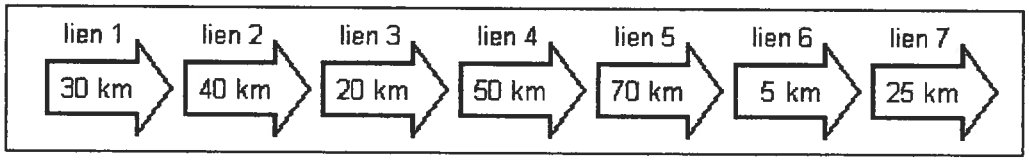

Figure 18 - Un segment de 240 km qui passe par 7 liens

Pour les explications qui suivent, nous supposerons que le segment transporte un débit d'OC-192 et que la longueur limite est de 80 km.

Si nous nous intéressons au lien 2 et que nous voulons enlever la compensation engendrée par le segment sur ce lien, alors une seule carte est nécessaire. Nous pouvons la placer au nœud qui se trouve entre les liens 2 et 3. Par contre, si nous nous intéressons au lien 3, alors il n'existe pas de solution où une seule carte est suffisante : il faut absolument ajouter deux cartes de transport. Il y a en outre plusieurs possibilités de placer ces deux cartes.

Revenons maintenant au cas général. Si la somme de la longueur du lien courant et de tous les liens qui le précèdent dans le segment est inférieure à la longueur limite pour le débit du segment, alors une seule carte de transport est nécessaire (par exemple, pour le lien 2, nous trouvons 40 km + 30 km < 80 km). Sinon, si la somme de la longueur du lien courant et de tous les liens qui le suivent dans le segment est inférieure à la longueur limite

relative au débit du segment, alors, dans ce cas aussi, une seule carte de transport est nécessaire (par exemple, pour le lien 6, nous trouvons 10 km + 15 km < 80 km). Sinon, alors c'est que deux cartes de transport sont nécessaires.

Où placer les cartes de transport si une carte est suffisante

Quand une seule carte de transport est suffisante pour sectionner le segment, il est assez simple de la placer convenablement. L'algorithme présenté ci-dessous a pour objectif de placer la carte au meilleur endroit possible, c'est-à-dire l'endroit où le plus de liens pourront en bénéficier. Par exemple, si, dans l'exemple précédent, nous voulions enlever la nécessité de compenser le lien 1, nous aurions pu engendrer une interruption au nœud qui se trouve entre les liens 1 et 2. Cependant, il serait beaucoup plus avantageux de le faire au nœud qui se trouve entre les liens 2 et 3. En procédant de cette façon, nous évitons que ce segment soit une des causes de la nécessité de compenser le lien 2.

L'idée derrière cet algorithme est de commencer par décider s'il faut partir du début ou de la fin du segment. Puis il faut progresser le long de ce segment (dans la direction appropriée) et s'arrêter à un nœud. L'objectif est de parcourir la plus longue distance possible sans pour autant dépasser ou atteindre la longueur limite. Puis, nous plaçons l'interruption à cet endroit, ce qui en fait bénéficier le maximum de liens possible.

```
/* Nous commençons tout d'abord par déterminer si il faut partir de la fin ou du
début du segment. */

distance_debut = distance qui sépare le début du segment au début du lien
distance_fin = distance entre la fin du lien et la fin du segment



Si distance_debut < distance_fin
```

```
        Nous commençons notre parcours à partir du début du segment.
        Nous indiquons que le lien courant est le premier lien du segment.

Sinon,

        Nous commençons notre parcours à partir de la fin du segment et nous nous
        dirigerons en sens inverse.
        Nous indiquons que le lien courant est le dernier lien du segment.

longueur_accumulée = longueur du lien courant

Tant que longueur_accumulée + longueur du lien suivant < longueur limite

        Le lien suivant (selon le sens dans lequel nous nous dirigeons) devient
        le lien courant.

        longueur_accumulée ← longueur_accumulée + longueur du lien courant

Si nous avions commencé notre parcours à partir du début du segment


        Placer une carte de transport au noeud qui est la destination du lien
        courant.

Sinon,

        Placer une carte de transport au noeud qui est la source du lien courant.
```

Ainsi, dans notre exemple, si nous nous intéressons au lien 1 ou au lien 2, notre algorithme choisira de placer une carte de transport au nœud qui se trouve entre les liens 2 et 3. De la même façon, si nous nous intéressons au lien 6 ou 7, l'algorithme placera une carte de transport au nœud qui se trouve entre les liens 5 et 6. Ainsi, nous faisons toujours bénéficier un nombre maximum de liens en posant une carte de transport.

Où placer les cartes de transport s'il faut deux cartes

Dans ce cas, bien choisir les deux nœuds où il faut placer une carte de transport est plus difficile. Encore une fois, l'algorithme proposé a pour objectif de placer les cartes aux meilleurs endroits possibles, de façon à en faire bénéficier les autres liens. L'idée ici est de parcourir le segment à partir du début et de le découper en autant de sections aussi longues que possible mais qui ne dépassent pas la limite en longueur. La Figure 19 illustre comment l'exemple de la Figure 18 doit être séparé en sections.

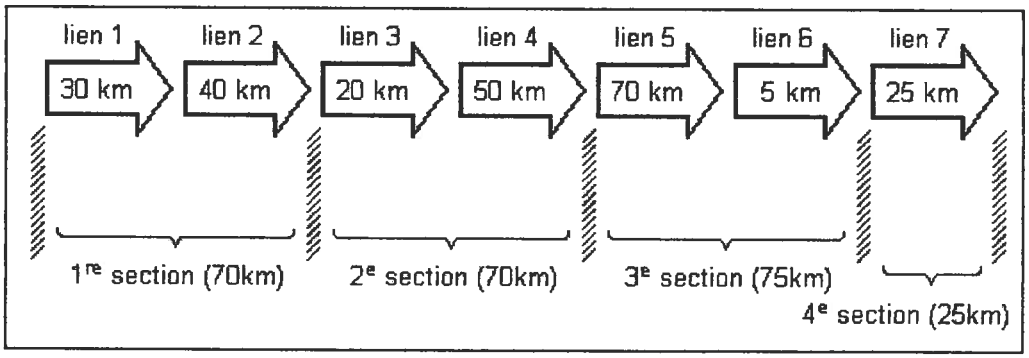

Figure 19 - Le segment de 240 km séparé en sections

Le lien sélectionné se trouve quelque part à l'intérieur d'une de ces sections. Il s'agit donc de placer une carte de transport au début de la section sélectionnée et une autre à la fin de manière à créer trois sous-segments à partir du premier segment et telle qu'au moins le sous-segment du milieu est assez court pour ne pas être une cause de la nécessité de compenser les liens faisant partie de ce sous-segment. Par exemple, si nous nous intéressons au lien 3, il faudrait placer une carte de transport au nœud entre les liens 2 et 3 et une autre entre les liens 4 et 5 comme cela est illustré à la Figure 20.

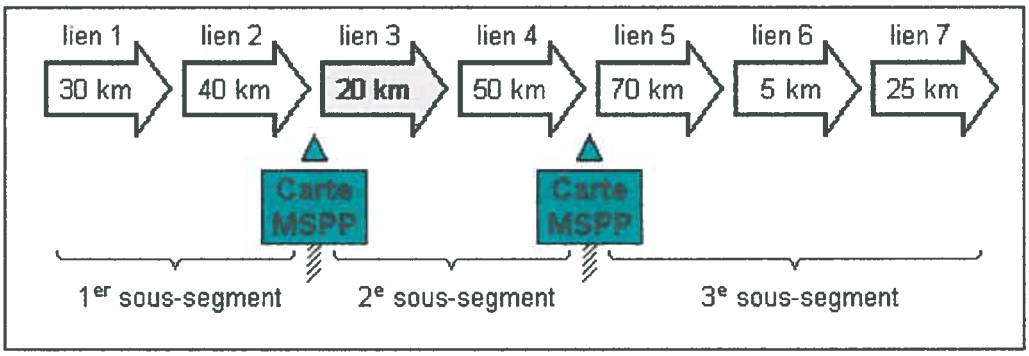

Figure 20 - Où placer les cartes de transport pour le lien 3 ?

Le détail de cet algorithme est présenté sous forme de pseudo code.

```
Le lien courant = le premier lien du segment

Tant que nous n'avons pas rencontré le lien qui nous intéresse (celui où il faut
enlever la nécessité de considérer la compensation)

        noeud_initial = le noeud qui correspond au noeud source du lien courant

        longueur_accumulée = longueur du lien courant

        Tant que longueur_accumulée + longueur du lien suivant < longueur limite

                Le lien suivant devient le lien courant
                longueur_accumulée ← longueur_accumulée + longueur du lien courant

        Si nous avons rencontré le lien qui nous intéresse

                Placer une carte de transport au noeud qui correspond à la
                destination du lien courant.

                Placer une seconde carte de transport au noeud noeud_initial.
```

Notons que dans cet algorithme, le lien qui nous intéresse ne devrait jamais être rencontré à la première itération de la boucle principale. Si tel était le cas, il aurait été possible d'éviter la nécessité de compenser ce lien à l'aide de seulement une carte de transport pour ce segment.

4.2.3 Discussion sur la stratégie

La stratégie décrite ici possède l'avantage d'être efficace et de toujours tenir à jour le meilleur (ou presque) compromis entre les cartes et la compensation. En fait, l'algorithme trouvera l'arrangement optimal la très grande majorité du temps. Cependant ceci n'est pas garanti à 100%, car il est possible d'imaginer un cas très particulier où l'algorithme ne choisira pas la meilleure solution. Il ne vaut pas la peine d'entrer trop dans les détails de ce cas particulier, mais remarquons que cela peut survenir lorsqu'il y a égalité entre deux liens pour savoir lequel doit être considéré en premier pour y remplacer la compensation par des cartes de transport. L'ordre est alors important et un exemple de situation où la solution optimale risquerait d'être manquée serait du même style que le cas problématique présenté à la fin de la section 3.2.2.

Notons finalement que la façon que nous proposons pour gérer la compensation possède un inconvénient : le fait d'effectuer ce traitement complexe à chaque fois qu'une solution est évaluée devient vite très long. Les expérimentations ont démontré que le temps de calcul de l'heuristique est beaucoup plus important lorsque la compensation est considérée. Des améliorations sont envisageables avec par exemple une réduction des aspects énumératifs à l'aide de critères de sélection pour mieux localiser les endroits où modifier la compensation.

Chapitre 5 – L'heuristique GRWABOU

Le problème posé dans le cadre de ce projet est résolu avec une heuristique de type tabou. Nous appelons GRWABOU l'heuristique développée dans le cadre de ce mémoire. Une version plus rudimentaire de GRWABOU est présentée dans [10] alors que le présent travail était en cours de réalisation.

Une heuristique tabou consiste principalement à trouver une solution initiale et à effectuer des mouvements, chacun permettant de passer d'une solution à une autre. Généralement, l'objectif d'un mouvement est d'améliorer la solution courante (c'est-à-dire en diminuer le coût). À la fin de l'exécution de l'heuristique, la meilleure solution réalisable rencontrée au fil des itérations est présentée. La particularité d'une « heuristique tabou » tient au fait que certains mouvements, qui impliqueraient de revenir à des solutions déjà rencontrées, sont interdits. Ceci permet d'éviter que l'algorithme ne finisse par cycler et tourner en rond (voir [5] et [6] pour plus de détails sur les heuristiques tabou).

Les différents éléments qui permettent d'effectuer cette recherche sont présentés et expliqués dans les sections qui suivent. À la première section, nous commençons par décrire comment les solutions sont évaluées. C'est une étape importante car l'évaluation permet de mesurer la qualité des solutions rencontrées. Ensuite, nous savons que pour une heuristique tabou, il est nécessaire de fournir une solution initiale. Ainsi, la section 5.2 présente la méthode utilisée pour construire cette solution initiale. Puis il faut des mouvements pour pouvoir passer d'une solution à une autre. La section 5.3 présente ces différents mouvements. Ensuite, la section 5.4 regroupe tous ces éléments pour enfin

présenter le détail de l'heuristique GRWABOU et les stratégies utilisées. Finalement, le chapitre se termine par une description de tous les paramètres du problème (section 5.5).

5.1 Évaluation d'une solution

Comme cela a été expliqué précédemment, le but de l'heuristique est d'explorer un grand nombre de solutions possibles pour ensuite choisir la meilleure parmi celle rencontrée. Ainsi, il faut pouvoir comparer quantitativement la qualité de ces solutions pour identifier les meilleures dans le but de bien orienter la recherche. C'est pour cette raison qu'avant d'entrer dans les détails du fonctionnement de l'heuristique, il faut expliquer comment les solutions sont évaluées. Ceci constitue une partie très importante du travail car c'est cette évaluation qui va permettre à l'heuristique de trouver une solution à coût minimum.

Tout d'abord, remarquons qu'il y a une fonction objective et une fonction d'évaluation. La fonction objective est ce qui détermine exactement le coût d'une solution, donc ce que nous voulons minimiser ; l'objectif de l'heuristique est de trouver une solution dont la fonction objective possède la plus petite valeur possible. La fonction d'évaluation quant à elle s'apparente à la fonction objective mais nous permet de manœuvrer plus efficacement à travers l'univers des solutions possibles. Par exemple elle permet de différencier deux solutions qui ont la même valeur objective. De plus, si une solution est irréalisable, la fonction objective ne peut l'exprimer car il ne s'agit pas d'une solution valide et cela n'aurait aucun sens. Cependant, la fonction d'évaluation permet d'affecter un certain 'coût' à la non réalisabilité, ceci nous permet de pouvoir être temporairement irréalisable pour explorer des solutions qui auraient été plus difficiles d'accès sinon.

Avant d'aller plus loin, il est important de noter un certain détail du calcul d'une évaluation : lorsque nous faisons face à une solution non réalisable due à au moins un débordement de capacité, alors pour toutes les longueurs d'onde de chaque lien qui doivent supporter un débit plus élevé qu'OC-192, c'est uniquement un port OC-192 qui sera utilisé pour satisfaire le trafic. Le débordement sera bel et bien considéré sauf que pour cette solution irréalisable nous posons qu'un port OC-192 peut émettre à un débit plus élevé.

5.1.1 La fonction objective

Comme mentionné précédemment, la fonction objective correspond exactement au coût de la solution étudiée. Elle est définie ainsi :

$$f_{OBJ} = \textit{coût des cartes} + \textit{coût de la compensation.}$$

Bien évidemment, lorsque nous ne considérons pas la compensation, le *coût de la compensation* vaut automatiquement 0. Rappelons aussi que pour qu'elle soit prise en considération, la solution dont nous calculons la valeur de la fonction objective doit être réalisable.

Le coût des cartes

Pour calculer le coût des cartes, il faut tout d'abord les dénombrer. Nous comptons le nombre de cartes à travers tout le réseau à l'aide de la procédure expliquée à la section 3.2. Ensuite, le coût est calculé ainsi :

coût des cartes = (*le nombre de cartes OC-48* × *coût d'une carte OC-48*)
+ (*le nombre de cartes OC-192* × *coût d'une carte OC-192*).

Les coûts des cartes OC-48 et OC-192 sont des paramètres du problème. Toutefois, il est important de mentionner ici que tout au long de ce travail, nous utilisons toujours les coûts définis à la Table 2, mais pour simplifier les calculs, ces coûts sont réduits d'un facteur de 1000. Ainsi, les coûts tels qu'utilisés dans l'heuristique sont définis dans la Table 10.

	Coût
Carte OC-48	4
Carte OC-192	10

Table 10 - Les coûts des cartes utilisés dans l'heuristique

Le coût de la compensation

Le chapitre 4 explique la méthode pour calculer la compensation pour une solution donnée. Les coûts utilisés sont ceux définis à la Table 9, sauf qu'encore une fois, ils sont réduits d'un facteur de 1000 pour simplifier les calculs et pour uniformiser les coûts des cartes et de la compensation. Ainsi, les coûts relatifs à la compensation utilisés dans l'heuristique sont ceux définis dans la Table 11.

	Coût
Coût de base	20
Coût par kilomètre	0.375 par km

Table 11 - Les coûts relatifs à la compensation utilisés dans l'heuristique

5.1.2 La fonction d'évaluation

La fonction d'évaluation tient compte des cartes, de la compensation et de la non réalisabilité. La fonction d'évaluation F_{EVAL} est définie ainsi :

$$
\begin{aligned}
F_{EVAL} = {} & \textit{coût des cartes} \times \textit{ratio d'évaluation pour les cartes} && 1\\
& + \textit{mesure corrective pour les cartes} && 2\\
& + \textit{coût de la compensation} \times \textit{ratio d'évaluation pour la compensation} && 3\\
& + \textit{mesure corrective pour la compensation} && 4\\
& + \textit{mesure corrective pour la non réalisabilité.} && 5
\end{aligned}
$$

Remarquons que si la compensation n'est pas considérée, les termes des lignes 3 et 4 valent automatiquement 0.

Le coût des cartes

Le coût des cartes est celui calculé pour la fonction objective. Ce coût est ensuite multiplié par le paramètre `evaluation_ratio_for_cards` (*ratio d'évaluation pour les cartes*). Il s'agit en fait d'un paramètre qui vaut généralement 1. Ce ratio indique, d'une certaine façon, l'importance à accorder au coût des cartes lorsque la solution est évaluée avec la fonction d'évaluation.

Mesure corrective pour les cartes

Pour deux solutions différentes dont le coût des cartes est identique, il est intéressant de pouvoir préférer une solution par rapport à l'autre selon certains critères. C'est la raison d'être de la *mesure corrective pour les cartes*.

Le critère choisi tient compte des ports qui sont inutilisés dans une solution. En effet, pour une solution donnée, nous calculons toujours le nombre de ports nécessaires, puis nous en déduisons le nombre de cartes en prenant le maximum entre le nombre de ports d'entrée et le nombre de ports de sortie. Ainsi, puisqu'il y a la plupart du temps un déséquilibre entre les ports d'entrée et de sortie, il y aura très souvent un certain nombre de ports qui seront inutilisés sur certaines cartes. La mesure corrective que nous appliquons ici est donc définie comme suit :

mesure corrective pour les cartes =

$$\left(\left(\frac{N_{OC-48} - U_{OC-48}}{N_{OC-48}} \times C_{OC-48} \right) + \left(\frac{N_{OC-192} - U_{OC-192}}{N_{OC-192}} \times C_{OC-192} \right) \right) \times \texttt{evaluation\_ratio\_for\_cards\_correction}$$

où

N_{OC-48} et N_{OC-192} = le nombre de cartes OC-48 et OC-192 respectivement,
U_{OC-48} et U_{OC-192} = le nombre de ports inutilisés OC-48 et OC-192 respectivement,
C_{OC-48} et C_{OC-192} = le coût d'une carte OC-48 et OC-192 respectivement.

Ainsi, pour deux solutions dont le coût des cartes est le même, c'est la solution où il y a le moins de ports inutilisés qui sera privilégiée. Le paramètre `evaluation_ratio_for_cards_correction` (qui correspond au ratio d'évaluation pour la mesure corrective des cartes) vaut 1 la plupart du temps, mais il est possible de le modifier pour donner une plus ou moins grande importance à la mesure corrective.

Le coût de la compensation

Le coût de la compensation est le même que celui calculé pour la fonction objective. Ce coût est ensuite multiplié par le paramètre `evaluation_ratio_for_compensation` (ratio d'évaluation pour la compensation). Comme pour le coût des cartes, il s'agit d'un paramètre qui vaut généralement 1 et il indique l'importance à accorder au coût de la compensation lorsque la solution est évaluée avec la fonction d'évaluation.

Mesure corrective pour la compensation

Lorsqu'il y a de la compensation sur un lien, ce peut être dû à un seul flot comme cela peut être dû à une grande quantité de flot ; mais la fonction objective ne fait pas la différence : il y a de la compensation ou il n'y en a pas. Pour cette raison, nous pouvons apporter une petite correction au coût de la compensation pour pouvoir faire la différence entre deux solutions différentes où le coût de la compensation est identique. Cette discrimination tient compte du nombre de longueurs d'onde qui sont à l'origine de la nécessité de compenser un lien. Par exemple, posons que nous avons deux liens à compenser et regardons les deux scénarios qui suivent.

Scénario A : sur le lien 1, il y a 4 longueurs d'onde qui doivent être compensées ;
sur le lien 2, il y a 4 longueurs d'onde qui doivent être compensées.

Scénario B : sur le lien 1, il y a 7 longueurs d'onde qui doivent être compensées ;
sur le lien 2, il y a 1 longueur d'onde qui doit être compensée.

Dans ces deux scénarios, 8 longueurs d'onde nécessitent de la compensation mais il faut privilégier le scénario B car sur le lien 1, la compensation est utile pour beaucoup de longueurs d'onde alors que sur le lien 2, il sera relativement facile d'éliminer la

compensation. Ainsi, la mesure corrective utilisée lors de l'évaluation va dans ce sens. Voici donc la formule qui permet de calculer cette mesure corrective :

$$\textit{mesure corrective pour la compensation} = \left(\sum_{liens} \sqrt{\frac{N_{lien}}{W}} \times \min(L_{lien}, L_{limite}) \times C_{km} \times C_{base} \right) \times \texttt{evaluation\_ratio\_for\_compensation\_correction}$$

où

N_{lien} = nombre de longueurs d'onde à compenser pour le lien courant,
W = nombre total de longueurs d'onde,
L_{lien} = longueur du lien,
L_{limite} = longueur limite OC-192 à partir de laquelle il faut utiliser de la compensation,
C_{km} = coût de la compensation par kilomètre,
C_{base} = coût de base pour compenser un lien.

Les expérimentations ont montré que la meilleure valeur empirique à affecter au paramètre `evaluation_ratio_for_compensation_correction` (ratio d'évaluation pour la mesure corrective de la compensation) doit être de l'ordre de 0,1. Il est possible de modifier cette valeur pour donner une plus ou moins grande importance à cette mesure corrective.

Mesure corrective pour la non réalisabilité

La mesure corrective pour gérer la non réalisabilité se sépare en deux parties :

mesure corrective pour la non réalisabilité =

(

mesure corrective pour la violation de la contrainte de capacité

+ *mesure corrective pour la violation de la contrainte des sauts optiques*

) × *ratio de validité*

Si pour une solution, l'une ou l'autre des deux contraintes pour être réalisable n'est pas respectée, alors sa valeur sera supérieure à 0 ce qui aura pour effet d'augmenter la valeur de la fonction d'évaluation et donc, par le fait même, rendre la solution étudiée moins intéressante. Ainsi, si la contrainte des capacités est violée, la mesure corrective à appliquer est celle-ci :

mesure corrective pour la violation de la contrainte de capacité =

(nombre de débordements de capacité répertoriés à travers la solution

× `evaluation_ratio_for_nb_capacity_overflow`)

+ (nombre d'OC-1 qui déborde dans la solution

× `evaluation_ratio_for_capacity_overflow`) .

Le paramètre `evaluation_ratio_for_nb_capacity_overflow` (ratio pour le nombre de débordements) et le paramètre `evaluation_ratio_for_capacity_overflow` (ratio pour le nombre d'OC-1 qui déborde) sont des paramètres du problème. Généralement, nous leurs affectons les valeurs 20 et 0,2 respectivement. L'intérêt de ce calcul est qu'il nous permet d'effectuer une discrimination correcte entre les différents scénarios possibles. L'exemple qui suit l'illustre bien.

Scénario A : À deux endroits nous trouvons un débordement.
Dans les deux cas nous trouvons l'équivalent d'un OC-193.

Scénario B : À un seul endroit il y a débordement de capacité.
Nous trouvons un OC-194 sur une longueur d'onde d'un lien.

Ainsi, ce sera le scénario B qui sera privilégié car il vaut mieux que l'équivalent d'un OC-2 qui déborde soit le plus concentré possible pour qu'il soit facile à éliminer.

La *mesure corrective pour la violation de la contrainte des sauts optiques* se calcule de cette façon :

mesure corrective pour la violation de la contrainte des sauts optiques =
(nombre minimal d'interruptions qu'il faut éliminer pour rétablir la réalisabilité × `evaluation_ratio_for_interruption_overflow`).

Comme pour la mesure corrective précédente, le paramètre `evaluation_ratio_for_interruption_overflow` (ratio pour le dépassement du nombre d'interruptions) est un paramètre du problème qui permet de définir l'importance à accorder à ce type de violation de contraintes. En général, cette valeur est fixée à 40.

Bien évidemment, si la solution courante est réalisable, aucune mesure corrective concernant la non réalisabilité ne doit être apportée ; le total des mesures correctives pour la non réalisabilité sera nul. Dans le cas contraire, cette valeur sera multipliée par le *ratio de validité*.

Le ratio de validité

Le *ratio de validité* permet de contrôler dynamiquement l'importance à accorder à la non réalisabilité. Ainsi, initialement, le ratio de validité sera relativement petit (par exemple : 1). Puis pour un groupe d'itérations consécutives passées dans le domaine non réalisable, ce ratio sera augmenté (jusqu'à un certain plafond) de telle sorte que pour l'itération suivante, choisir une solution non réalisable soit de plus en plus un mauvais choix. Puis au fur et à mesure que la solution courante redevient réalisable, il convient de faire diminuer graduellement ce ratio jusqu'à ce qu'il revienne à sa valeur initiale.

Les paramètres concernant la gestion du ratio de validité sont des paramètres du problème. En général, leurs valeurs sont les suivantes :

```
validity_ratio_minimum = 1 ,
validity_ratio_maximum = 5 ,
validity_ratio_increase_speed = 1,1 ,
validity_ratio_decrease_speed = 0,85 .
```

Lorsque l'heuristique rencontre des solutions non réalisables, le *ratio de validité* sera multiplié par `validity_ratio_increase_speed` ; et de la même façon, lorsque des solutions réalisables sont explorées par l'heuristique, le *ratio de validité* sera multiplié par `validity_ratio_decrease_speed`.

Ainsi, dans sa globalité, la fonction d'évaluation tient compte d'un maximum d'informations pour pouvoir calculer le plus correctement possible la qualité d'une solution. Chaque fois qu'un mouvement est effectué ou comparé à un autre mouvement, c'est toujours la fonction d'évaluation qui est utilisée comme facteur de décision.

5.2 Construction de la solution initiale

Pour pouvoir passer à l'exploration des solutions, l'heuristique GRWABOU doit avoir un point de départ : la solution initiale. C'est à partir de cette solution que la première itération de l'heuristique sera effectuée. Ainsi, dans le but de partir du bon pied, une attention particulière a été apportée pour bien concevoir la solution initiale de façon à s'approcher plus rapidement des bonnes solutions. L'idée ici consiste à commencer par regrouper ensemble les requêtes de façon à remplir le plus possible des capacités OC-192. Ensuite, le reste des requêtes est groupé sur des flots qui seront routés sur les plus courts chemins.

Voici une description des étapes à suivre pour obtenir la solution initiale.

1. Tout d'abord, chaque demande est séparée en x flots de débit OC-192 et au plus un flot (qui représente les restes) dont le débit est compris entre OC-1 et OC-191. Par exemple, si pour une source vers une destination nous avons une demande k_{sd} qui requiert une bande passante OC-390, elle sera séparée en deux flots de débit OC-192 et un flot ayant un débit d'OC-6 ($2 \times 192 + 6 = 390$).

2. Ensuite, les flots sont ordonnés selon leur débit de façon décroissante. Puis, chaque flot est routé et affecté à une longueur d'onde selon cet ordre. Quand il faut choisir le chemin physique qu'empruntera un flot, nous commençons par essayer le plus court chemin sur toutes les longueurs d'onde. La longueur d'onde choisie sera celle qui engendrera la solution de coût minimum (selon la fonction d'évaluation) tout en restant dans le domaine réalisable (nous évitons le plus possible de débuter avec une solution non réalisable). En cas d'ex æquo, c'est le hasard qui fera office de critère

secondaire. Cependant, il se peut qu'emprunter le plus court chemin sur n'importe laquelle des longueurs d'onde engendre toujours une solution irréalisable. Dans ce cas, nous répétons ce processus avec le deuxième plus court chemin et ainsi de suite jusqu'à avoir épuisé tous les chemins possibles. Si et seulement si aucune solution réalisable n'a pu être trouvée, nous admettons alors une solution irréalisable où le flot passe par le plus court chemin.

5.3 Description des mouvements

Nous avons mentionné plus tôt que lors de l'exécution de l'heuristique, il faut effectuer plusieurs mouvements consécutifs pour explorer plusieurs solutions possibles, ceci toujours dans l'espoir de trouver des solutions de moins en moins coûteuses. Pour atteindre cet objectif, plusieurs types de mouvements ont été élaborés ; une plus grande diversité de mouvements donne en quelque sorte plus de souplesse à l'heuristique. Les mouvements qui perturbent beaucoup la solution courante permettent par exemple d'aller explorer d'autres régions du domaine des solutions, alors que les mouvements qui perturbent peu la solution courante permettent d'explorer plus en profondeur un certain voisinage de la solution courante. Voici une liste et une description de tous les types de mouvement, mais avant, jetons un coup d'œil sur la façon dont la liste des éléments tabous est gérée par l'heuristique.

5.3.1 Gestion de la liste tabou

Dans une heuristique tabou, lorsqu'un mouvement est effectué, il est mémorisé pendant un certain nombre d'itérations de façon à ce que le mouvement inverse ne puisse

pas être exécuté. Ceci a pour objectif d'éviter qu'il n'apparaisse des cycles de solutions lors de la recherche. Le nombre de mouvements à conserver dans la liste des éléments tabous est un paramètre du problème, nous appellerons ce paramètre `number_of_element_in_tabu_list`. La règle générale stipule que le nombre de mouvements tabous doit être approximativement entre 10 % et 15 % des mouvements possibles.

Tous les types de mouvements que nous avons développés permettent de déplacer un ou plusieurs flots. Ainsi, le flot devient l'unité de base des mouvements en quelque sorte. La stratégie que nous proposons consiste à considérer les flots comme étant tabous. C'est-à-dire que lorsqu'un flot est déplacé, il est rendu tabou, il ne peut donc pas être déplacé de nouveau pendant un certain nombre d'itérations (ce qui implique qu'il ne peut pas retrouver son chemin physique initial pendant ce nombre d'itérations).

5.3.2 Mouvement A – déplacer un flot

Il s'agit ici du mouvement de base le plus élémentaire : nous sélectionnons un flot et nous lui affectons un autre chemin et/ou une autre longueur d'onde. Il s'agit d'un mouvement paramétrable de petite envergure qui permet d'explorer le voisinage proche d'une solution.

Pour ce faire, nous nous intéressons à un nœud, puis nous énumérons tous les flots non tabous qui utilisent au moins un port de ce nœud, que ce soit parce qu'ils y débutent, qu'ils s'y terminent ou qu'ils y passent dû à une interruption. Nous ne retiendrons en fait qu'un certain pourcentage de tous ces flots : nous appellerons ce paramètre le *pourcentage de détection*. Ensuite, pour chaque flot retenu, nous appliquons le procédé suivant.

1. Le flot est retiré de la solution courante.

2. Le flot est ensuite affecté à un certain pourcentage (paramètre que nous appellerons *pourcentage de choix*) de tous les autres chemins physiques. Notons que parmi l'ensemble des chemins physiques potentiels, nous considérons seulement les x plus courts chemins sur toutes les longueurs d'onde (x dépend directement du paramètre `number_of_extra_length_for_considering_path` qui est expliqué à la section 5.5). Puis, chaque fois que le flot est affecté à un autre chemin physique, nous évaluons cette nouvelle solution.

3. Ensuite, nous conservons, parmi tous les chemins physiques essayés, celui dont l'impact sur le coût est le plus avantageux (rappelons que nous utilisons la fonction d'évaluation pour juger cet impact). Comme toujours, le hasard est utilisé pour séparer les ex æquo. Notons aussi qu'il est possible qu'un déplacement sur le nouveau chemin physique détériore le coût de la nouvelle solution.

4. Le flot testé qui avait été préalablement retiré de la solution est ensuite rétabli à son chemin physique initial.

5. Nous répétons ensuite les étapes 1 à 4 pour tous les flots retenus initialement.

6. Ainsi, pour chaque flot qui passe par le nœud en question, nous connaissons le meilleur chemin physique alternatif. Il s'agit maintenant de sélectionner le flot dont le meilleur déplacement améliore le plus la solution courante.

7. Il faut ensuite retirer ce flot et de l'affecter au chemin physique qui a été choisi. Puis il faut mettre à jour la liste des éléments tabous en y ajoutant ce flot.

Il est possible de paramétrer le mouvement A pour que ce mouvement effectue une recherche dans un voisinage différent dans le but de modifier certains attributs de la

solution courante. Ainsi, nous pouvons découper des flots, ou encore, si la réalisabilité devient un problème, tenter de la rétablir en forçant un peu les choses.

Découper des flots

Minimiser le nombre total de flots (autrement dit, regrouper le plus possible les requêtes) n'est pas toujours la meilleure tactique. Par exemple, il peut être avantageux qu'un flot ayant un débit OC-96 soit séparé en deux flots OC-48. Ainsi, il est possible d'effectuer le mouvement A avec cette philosophie. Le seul changement dans le procédé présenté précédemment est qu'une fois que tous les flots potentiels pouvant être déplacés ont été choisis, au lieu de considérer le flot en entier, nous ne considérons qu'une certaine fraction du flot (ce qui résulte en deux flots plus petits). Tout d'abord, il faut mentionner que seulement des flots ayant au moins un débit d'OC-49 sont éligibles à un tel mouvement. Nous tenterons de déplacer le flot dont un débit d'OC-48 a été soustrait au débit total du flot. Ceci afin que le flot original reste toujours avec un débit d'OC-48. Ainsi, par exemple, si nous voulons déplacer un flot ayant un débit d'OC-80, nous nous intéresserons uniquement à déplacer un OC-32 de ce flot.

Rétablir la réalisabilité

Il est possible de paramétrer le mouvement de telle façon que la solution courante redevienne réalisable plus rapidement. En fait, après avoir fait une série de mouvements, il est possible d'arriver temporairement à des solutions non réalisables. Plus le nombre d'itérations consécutives où la solution est irréalisable augmente, plus l'accent est mis pour tenter de redevenir réalisable (voir la fonction d'évaluation à la section 5.1.2). Toutefois, quand le nombre d'itérations passées dans le domaine non réalisable devient trop important, cet effort n'est pas suffisant ; il faut alors procéder de façon plus drastique pour revenir à une solution réalisable.

Pour ce faire, nous commençons par sélectionner tous les flots (tabous, ou non, et ce peu importe les nœuds par lesquels ils transitent). Ainsi, que ce soit un débordement de capacité ou un dépassement du nombre maximal de sauts optiques (ou les deux) qui cause la non réalisabilité, le flot problématique pourra toujours être identifié et être routé ailleurs. Bien entendu, il faut refaire ce mouvement jusqu'à ce que tous les flots problématiques soit déplacés, autrement dit, jusqu'à ce que la solution redevienne réalisable.

5.3.3 Mouvement B – supprimer un port

Il s'agit ici d'un mouvement qui bouleverse la solution courante en redirigeant tous les flots qui passent par un port de façon à rendre inutilisé (et donc à supprimer) le port en question. Comme pour le mouvement A, nous nous intéressons ici à un nœud particulier. Nous commençons par regarder tous les ports OC-48 et OC-192 en se demandant s'il y a plus de ports d'entrée ou de ports de sortie de chaque type. Pour une capacité de transport donnée, s'il y a plus de ports d'entrée, nous privilégierons d'enlever ce type de port, alors que s'il y a plus de ports de sortie, nous donnerons plus de chance au retrait d'un de ces ports.

Ensuite, nous avons défini deux critères qui permettent d'identifier le port à enlever. Le choix du critère est un paramètre du mouvement. Les critères tels que définis ci-dessous ont été obtenus par essais et erreurs.

Critère 1

Le port choisi est celui dont cette somme est la plus petite :

$$3 \times b_{add/drop} + b_{int}$$

où

$b_{add/drop}$ = la bande passante qui transite par ce port,
b_{int} = la bande passante passant par ce port dû à une interruption.

Critère 2

Le port choisi est celui dont ce ratio est le plus petit :

$$n_{int} / n_{tot}$$

où

n_{int} = le nombre de flots qui passent par ce port dû à une interruption,
n_{tot} = le nombre de flots total qui transite par ce port.

Avant de choisir le port à retirer, il faut s'assurer que tous les flots devant être déplacés pour effectuer le mouvement ne soient pas tabous. Dans le cas où au moins un des flots est tabou, il ne faut pas considérer ce port. Toutefois, dans le cas particulier où par tous les ports transite au moins un flot tabou, alors il faut choisir parmi les ports accommodants un nombre minimum de flots tabous.

Maintenant que le port à retirer est identifié, il faut faire l'inventaire de tous les flots qui utilisent ce port ; c'est-à-dire tous les flots qui débutent ou terminent à ce port ainsi que les flots qui passent par ce port à cause d'une interruption. Ces flots sont retirés de la solution courante. Puis un à un, dans un ordre aléatoire, ils sont par la suite affectés à de nouveaux chemins physiques. Notons que les flots sont pris dans un ordre aléatoire car il y aurait beaucoup de choses à prendre en considération (comme la longueur du plus court chemin ou le débit de chaque flot) et de toute façon, l'objectif du mouvement B est de perturber la solution courante. Ensuite, le choix du nouveau chemin physique de chaque flot se fait exactement de la même façon que pour le mouvement A sauf que les paramètres *pourcentage de détection* et *pourcentage de choix* valent automatiquement 100% (ce ne sont donc pas des paramètres pour le mouvement B). Une autre différence par rapport au mouvement A est qu'une attention particulière est portée pour s'assurer que le port que nous venons tout juste d'enlever ne soit pas recréé immédiatement. La liste taboue est mise à jour avec les flots qui viennent d'être déplacés. Ainsi, le port ne peut pas être recréé par un de ces flots pour un certain nombre d'itérations ; mais il peut très bien réapparaître si un autre flot venait à requérir sa présence.

Notons finalement que lorsqu'un mouvement B est effectué, il est possible qu'un des flots devant être déplacé soit réaffecté à son chemin physique initial si et seulement si cela est inévitable car il n'y a pas d'autres chemins physiques disponibles; ou encore si ce flot utilisait le port supprimé pour effectuer une interruption au nœud concerné et que le bouleversement engendré par le mouvement fait en sorte que le flot n'a plus à être interrompu à ce nœud.

5.3.4 Mouvement C – supprimer la nécessité de compenser un lien

Ce mouvement s'apparente au mouvement B sauf qu'ici, pour une solution donnée, nous voulons modifier le coût relatif à la compensation. Encore une fois, il s'agit ici d'un mouvement qui bouleverse beaucoup la solution courante. L'objectif est de déplacer tous les flots qui passent par un lien où il y a de la compensation et qui sont à l'origine de la nécessité de compenser le lien. Ceci de façon à enlever pour un certain temps la compensation sur le lien concerné. Il est important de noter que ce type de mouvement ne peut être effectué que si la compensation est prise en considération lors de l'exécution de l'heuristique.

Nous commençons donc par faire une liste de tous les liens où il y a de la compensation. L'objectif est maintenant d'enlever la nécessité de compenser un de ces liens. Pour choisir le lien, nous utilisons un de ces critères (comme pour le mouvement B, le critère choisi constitue un paramètre du mouvement).

Critère 1

Le lien est choisi au hasard.

Critère 2

Le lien choisi est celui par lequel transite le plus petit nombre de flots étant à l'origine de la nécessité de recourir à la compensation.

Maintenant que le lien où il faut faire l'effort de retirer la compensation est identifié, il faut reprendre l'inventaire de tous les flots qui sont à l'origine de la nécessité de compenser ce lien. Qu'ils soient tabous ou non, tous ces flots sont retirés de la solution courante. Ensuite, un à un, ils sont chacun affectés à un nouveau chemin physique. Encore

une fois, le choix de leurs nouveaux chemins physiques se fait exactement de la même façon que pour le mouvement A sauf que les paramètres *pourcentage de détection* et *pourcentage de choix* valent automatiquement 100% et nous interdisons que le nouveau chemin emprunte le lien concerné, sauf si cela n'est pas possible (si tous les chemins qui relient une source à une destination passent tous par un certain lien par exemple). La liste tabou sera ensuite mise à jour avec tous les flots venant d'être déplacés.

Notons que ce mouvement bouleverse beaucoup plus la solution courante que le mouvement B car en général, il y a plus de flots qui sont à l'origine de la nécessité de compenser un lien que de flots qui utilisent un port spécifique. Aussi, pour le mouvement B, il est possible qu'un flot qui doit être déplacé passe de nouveau par le même chemin, voire même par la même longueur d'onde ; alors que pour le mouvement C, nous interdisons qu'il repasse par le même lien, ce qui est beaucoup plus contraignant. Le résultat étant que ce mouvement apportera beaucoup de changement à la solution courante.

5.4 Le détail de l'heuristique

L'heuristique consiste en gros en une recette qui est une séquence programmée des différents mouvements définis à la section 5.3. Les mouvements sont effectués selon un certain ordre et une certaine logique qui consiste principalement à deux choses : perturber une solution, puis tenter de la faire converger vers un optimum local.

Dans tous les scénarios considérés, il convient de commencer par ordonner les nœuds selon un certain critère. Les nœuds du début de la liste sont des nœuds auxquels il est particulièrement préférable d'effectuer certains changements, beaucoup plus que pour les derniers nœuds de la liste. Ensuite, nous effectuons une série de mouvements sur ces nœuds selon leur rang. Ceci a pour objectif d'orienter les efforts vers les nœuds où nous

jugeons qu'il y a le plus d'améliorations à apporter (réduire le nombre d'interruptions qui s'y produit, augmenter le pourcentage d'occupation des ports en termes de bande passante, etc.). De plus, lorsque nous voulons converger vers un optimum local, la stratégie mise en œuvre consiste à effectuer un mouvement A toujours sur le même nœud, ceci tant que les mouvements effectués ne détériorent pas la valeur objective de la solution pendant un certain nombre d'itérations consécutives et tant que nous trouvons au moins une amélioration de la valeur objective pour un certain nombre d'itérations consécutives. Ainsi, le nombre maximal de mouvements A consécutifs qui peuvent détériorer la valeur objective est : `maximum_number_of_deterioration`. De plus, le nombre maximal permis de mouvements A consécutifs qui peuvent ne pas améliorer la valeur objective est : `maximum_number_of_stagnation`. Lorsqu'une de ces deux limites est atteinte, il faut arrêter d'effectuer des mouvements A sur le nœud concerné et passer à l'étape suivante (des mouvements A sur un autre nœud ou un autre mouvement selon la façon dont l'heuristique est définie). Voyons maintenant plus en détail les critères d'ordonnancement des noeuds et les scénarios.

5.4.1 Les critères d'ordonnancement des noeuds

Avant de procéder à l'ordonnancement, chaque nœud est analysé, il faut faire le décompte, sur tous les ports du nœud, de la bande passante qui est inutilisée, la bande passante utilisée par des interruptions, le nombre de flots où il y a interruption, ainsi que la bande passante totale et le nombre de flots qui transitent par ce nœud au total. Après avoir effectué différentes expériences de calcul, nous avons sélectionné quatre critères d'ordonnancement. Les coefficients proposés résultent d'expériences réalisées avec différentes valeurs numériques.

Critère A

bande passante inutilisée + 2 × la bande passante utilisée pour des interruptions

Critère B

3 × bande passante inutilisée + la bande passante utilisée pour des interruptions

Critère C

20 × le nombre de flots interrompus – le nombre total de flots

Critère D

bande passante inutilisée + la bande passante utilisée pour des interruptions

5.4.2 Les scénarios

Plusieurs expérimentations ont permis de sélectionner quatre scénarios qui se sont révélés donner des résultats satisfaisants. Mentionnons tout de suite que, dans tous les scénarios, la boucle principale est séparée en trois petits scénarios qui seront exécutés en *round-robin*. C'est-à-dire qu'à la première itération de la grande boucle, le premier petit scénario sera exécuté, puis ce sera le deuxième, puis le troisième, puis nous reviendrons au premier, etc.

Les scénarios que nous proposons sont présentés sous forme de pseudo code. Dans le but d'alléger l'écriture des algorithmes, lorsqu'il sera inscrit :

```
MoveA (...)
```

il faudra toujours lire la procédure présentée ci-dessous.

```
nombre_de_détériorations ← 0
nombre_de_mouvements_stagnants ← 0
```

```
faire
        MoveA (...)
        Si la nouvelle solution est moins bonne que la précédente du point de vue
        de la fonction objective
                nombre_de_détériorations ← nombre_de_détériorations + 1
                nombre_de_mouvements_stagnants ← nombre_de_mouvements_stagnants + 1
        Sinon, si la nouvelle solution est identique à la précédente du point de
        vue de la fonction objective
                nombre_de_détériorations ← 0
                nombre_de_mouvements_stagnants ← nombre_de_mouvements_stagnants + 1
        Sinon
                nombre_de_détériorations ← 0
                nombre_de_mouvements_stagnants ← 0

tant que (nombres_de_détériorations < maximum_number_of_deterioration et
          nombres_de_mouvements_stagnants < maximum_number_of_stagnation)
```

Notons toutefois que cela est valable uniquement pour le mouvement A lorsqu'il n'est pas utilisé pour couper des connexions ou pour redevenir réalisable. Dans le cas où nous voulons utiliser le mouvement A pour couper des connexions, nous utiliserons le nom « `MoveA-cut` » alors que lorsque nous voudrons redevenir réalisable à l'aide du mouvement A, nous indiquerons « `MoveA-rescue` ». Avant de présenter les pseudos codes des scénarios, notons que le mouvement A prend trois paramètres :

`MoveA (numéro du noeud, *pourcentage de détection*, *pourcentage de choix*) .`

Le mouvement B quant à lui prend deux paramètres :

`MoveB (numéro du noeud, numéro du critère de sélection de port) .`

Concernant les quatre scénarios, nous remarquons que le scénario A est celui qui effectue le moins de mouvements perturbateurs (mouvements B). Le scénario B quant à lui effectue davantage de mouvements B. Le scénario C effectue lui aussi beaucoup de mouvements B et aussi beaucoup de mouvements « `MoveA-cut` ». Finalement, le scénario D est un mélange des trois scénarios précédents sauf que pour les mouvements A, le *pourcentage de détection* et le *pourcentage de choix* ont été fixés à des valeurs inférieures à 100% la plupart du temps. Voyons maintenant en détail les scénarios proposés.

Scénario A

```
Construire la solution initiale.

Tant que le critère d'arrêt n'est pas rencontré

      Si le ratio de validité est au maximum

            Faire au maximum 10 fois
                  MoveA-rescue ()
            Tant que nous sommes irréalisable

      Les noeuds sont ordonnés selon le critère.

      Si nous devons effectuer le premier petit scénario
            Pour i = 1 jusqu'à (nombre_de_noeud × 1/3)
                  MoveA (i^ème noeud de la liste, 100%, 100%)

      Sinon, si nous devons effectuer le deuxième petit scénario
            MoveB (1^er noeud de la liste, critère 1)
            MoveB (2^ème noeud de la liste, critère 2)
            Pour i = 1 jusqu'à (nombre_de_noeud × 1/4)
                  MoveA ((i+2)^ème noeud de la liste, 100%, 100%)
```

```
Sinon, c'est que nous devons effectuer le troisième petit scénario
        MoveA-cut (1er noeud de la liste, 100%, 100%)
        Pour i = 1 jusqu'à (nombre_de_noeud × 1/5)
                MoveA ((i+1)ème noeud de la liste, 100%, 100%)
```

Scénario B

```
Construire la solution initiale.

Tant que le critère d'arrêt n'est pas rencontré

    Si le ratio de validité est au maximum

        Faire au maximum 10 fois
            MoveA-rescue ()
        Tant que nous sommes irréalisable

    Les noeuds sont ordonnés selon le critère.

    Si nous devons effectuer le premier petit scénario
        MoveB (1er noeud de la liste, critère 1)
        Pour i = 1 jusqu'à (nombre_de_noeud × 2/5)
            MoveA (ième noeud de la liste, 100%, 100%)

    Sinon, si nous devons effectuer le deuxième petit scénario
        MoveB (1er noeud de la liste, critère 1)
        MoveB (2ème noeud de la liste, critère 2)
        Pour i = 1 jusqu'à (nombre_de_noeud × 1/3)
            MoveA ((i+2)ème noeud de la liste, 100%, 100%)

    Sinon, c'est que nous devons effectuer le troisième petit scénario
        MoveA-cut (1er noeud de la liste, 100%, 100%)
        Pour i = 1 jusqu'à (nombre_de_noeud × 1/5)
            MoveA ((i+1)ème noeud de la liste, 100%, 100%)
        MoveB (2ème noeud de la liste, critère 1)
```

Scénario C

```
Construire la solution initiale.

Tant que le critère d'arrêt n'est pas rencontré

      Si le ratio de validité est au maximum

            Faire au maximum 10 fois
                  MoveA-rescue ()
            Tant que nous sommes irréalisable

      Les noeuds sont ordonnés selon le critère.

      Si nous devons effectuer le premier petit scénario
            MoveB (1er noeud de la liste, critère 1)
            Pour i = 1 jusqu'à (nombre_de_noeud × 2/5)
                  MoveA (ième noeud de la liste, 100%, 100%)
            MoveA-cut (1er noeud de la liste, 100%, 100%)

      Sinon, si nous devons effectuer le deuxième petit scénario
            MoveB (1er noeud de la liste, critère 1)
            MoveA-cut (2er noeud de la liste, 100%, 100%)
            MoveB (3ème noeud de la liste, critère 2)
            Pour i = 1 jusqu'à (nombre_de_noeud × 1/2)
                  MoveA (iéme noeud de la liste, 100%, 100%)

      Sinon, c'est que nous devons effectuer le troisième petit scénario
            MoveA-cut (1er noeud de la liste, 100%, 100%)
            Pour i = 1 jusqu'à (nombre_de_noeud × 1/4)
                  MoveA ((i+1)éme noeud de la liste, 100%, 100%)
            MoveB (2éme noeud de la liste, critère 1)
```

Scénario D

```
Construire la solution initiale.

Tant que le critère d'arrêt n'est pas rencontré

      Si le ratio de validité est au maximum

            Faire au maximum 10 fois
                  MoveA-rescue ()
            Tant que nous sommes irréalisable

      Les noeuds sont ordonnés selon le critère.

      Si nous devons effectuer le premier petit scénario
            MoveB (1er noeud de la liste, critère 1)
            Pour i = 1 jusqu'à (nombre_de_noeud × 1/2)
                  MoveA (ième noeud de la liste, 60%, 60%)

      Sinon, si nous devons effectuer le deuxième petit scénario
            MoveB (1er noeud de la liste, critère 1)
            MoveA-cut (2er noeud de la liste, 100%, 100%)
            MoveB (3ème noeud de la liste, critère 2)
            Pour i = 1 jusqu'à (nombre_de_noeud × 1/2)
                  MoveA (ième noeud de la liste, 80%, 80%)

      Sinon, c'est que nous devons effectuer le troisième petit scénario
            MoveA-cut (1er noeud de la liste, 100%, 100%)
            Pour i = 1 jusqu'à (nombre_de_noeud × 1/2)
                  MoveA ((i+1)ème noeud de la liste, 100%, 100%)
```

5.5 Les paramètres

Nous avons survolé rapidement les paramètres qu'il est possible de fournir au problème. Nous présentons maintenant la liste exhaustive de tous ces paramètres. Plusieurs détails importants sont indiqués pour les paramètres de la recherche tabou (section 5.5.1).

5.5.1 Les paramètres de la recherche tabou

`number_of_element_in_tabu_list` : il s'agit du nombre de flots à conserver dans la liste tabou. Pour plus de détails, revoir la section 5.3.1.

`number_of_extra_length_for_considering_path` : ce paramètre permet de gérer le nombre de plus courts chemins à considérer. Soit x la longueur (en termes de nombre de liens) du plus court chemin, alors tous les chemins dont la longueur est égale ou inférieure à $(x + \lfloor$`number_of_extra_length_for_considering_path`$\rfloor)$ sont considérés. Notons que le paramètre `number_of_extra_length_for_considering_path` peut prendre une valeur fractionnaire. Dans un tel cas, un chemin dont la longueur est de $(x + \lceil$`number_of_extra_length_for_considering_path`$\rceil)$ possède (partie fractionnaire (`number_of_extra_length_for_considering_path`)) chance d'être considéré.

`maximum_number_of_deterioration` : ce paramètre est utilisé dans le détail de l'heuristique, il indique le nombre de détériorations consécutives tolérables avant de passer à un autre mouvement. Voir la section 5.4 pour plus de détails.

`maximum_number_of_stagnation` : ce paramètre est aussi utilisé dans le détail de l'heuristique, il indique le nombre de mouvements sans conséquence sur la fonction d'évaluation consécutifs tolérables avant de passer à un autre mouvement. Voir la section 5.4 pour plus de détails.

5.5.2 Les paramètres pour la fonction d'évaluation

Tous les paramètres relatifs à la fonction d'évaluation sont expliqués à la section 5.1.2. Voici la liste de ces paramètres :

```
evaluation_ratio_for_cards
evaluation_ratio_for_cards_correction

evaluation_ratio_for_compensation
evaluation_ratio_for_compensation_correction
validity_ratio_minimum
validity_ratio_maximum
validity_ratio_increase_speed
validity_ratio_decrease_speed

evaluation_ratio_for_nb_capacity_overflow
evaluation_ratio_for_capacity_overflow
evaluation_ratio_for_interruption_overflow
```

Chapitre 6 – Les résultats

Nous arrivons maintenant à la partie où nous allons exécuter le programme avec ses différentes fonctionnalités pour tenter de faire ressortir, à travers les résultats, les tendances ainsi que l'importance de chacun des paramètres. Concernant les paramètres de l'heuristique, ils ont tous été présentés à la section 5.5. Toutefois, pour le reste du mémoire, certains d'entre eux sont fixés, leur valeur ne changera jamais pour toutes les exécutions présentées dans ce chapitre. La liste de ces paramètres accompagnés de leur valeur est présentée ci-dessous (notons que ces valeurs ont été trouvées par essais et erreurs).

```
evaluation_ratio_for_cards = 1
evaluation_ratio_for_cards_correction = 1
evaluation_ratio_for_compensation = 1

validity_ratio_maximum = 5
validity_ratio_increase_speed = 1,1
validity_ratio_decrease_speed = 0,85

evaluation_ratio_for_nb_capacity_overflow = 20
evaluation_ratio_for_capacity_overflow = 0,2
evaluation_ratio_for_interruption_overflow = 40
```

Pour les différentes exécutions, les autres paramètres peuvent prendre différentes valeurs qui seront indiqués dans les sections subséquentes lorsque cela sera approprié.

6.1 Le réseau NSF

Pour tester le programme, le premier réseau que nous utilisons est le réseau NSF (voir par exemple [14]) inspiré d'un réseau aux États-Unis qui relie les principales villes américaines. La Figure 21 illustre ce réseau.

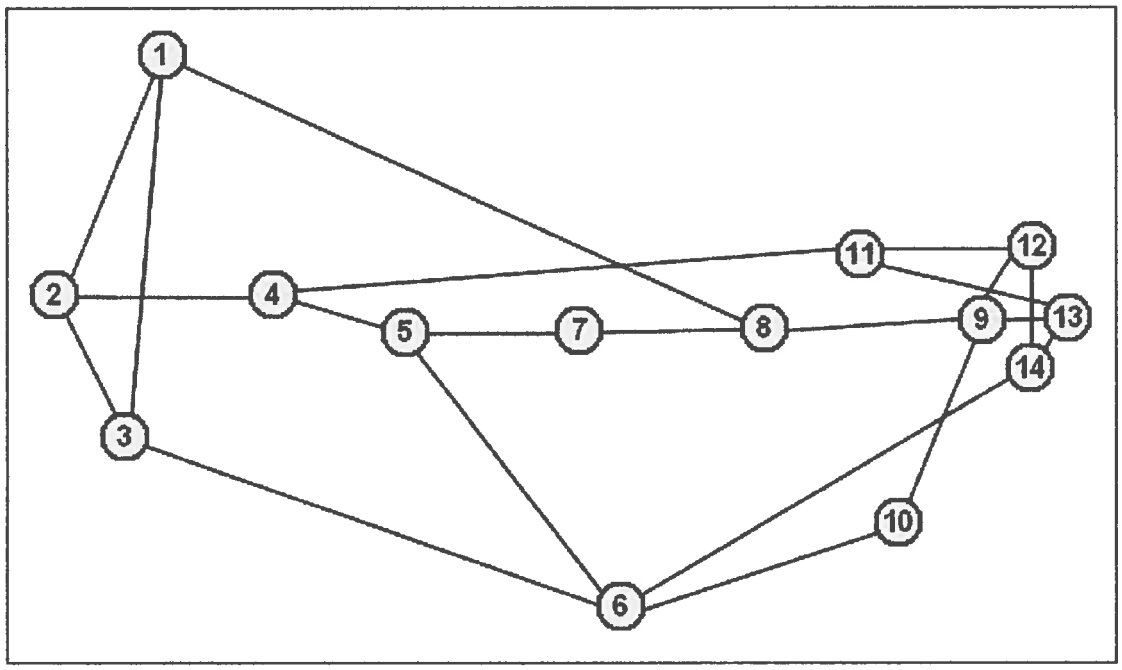

Figure 21 - Le réseau NSF

Pour cette instance, nous avons généré aléatoirement le trafic. Pour chaque paire de nœuds possible, nous avons tiré au hasard la bande passante demandée de cette façon :

nombre de requêtes OC-1 = une valeur aléatoire parmi {0, 1, 2, 3},
nombre de requêtes OC-3 = une valeur aléatoire parmi {0, 1, 2, 3},

nombre de requêtes OC-12 = une valeur aléatoire parmi {0, 1, 2, 3, 4, 5, 6},
nombre de requêtes OC-48 = une valeur aléatoire parmi {0, 1, 2, 3, 4, 5, 6}.

Ainsi, en moyenne, le trafic entre deux nœuds est l'équivalent d'un OC-186.9. De plus, pour cette instance, la borne inférieure a été calculée selon l'algorithme proposé à la section 3.2.2, sa valeur est de 1946. Nous calculons aussi que le nombre minimal de flots que peut avoir une solution valide est de 269 (c'est le nombre de flots que produit la fonction qui génère une solution initiale).

6.1.1 Étude de sensibilité des paramètres

Nous allons maintenant procéder à une étude de sensibilité de plusieurs paramètres de l'heuristique GRWABOU. Nous pouvons toujours exécuter le programme pendant un certain nombre d'itérations, puis aboutir avec une solution (la meilleure de toutes celles rencontrées au fil les itérations). Toutefois, les expérimentations ont démontré que si les paramètres de l'heuristique sont biens choisis, alors la meilleure solution pour un même nombre d'itérations est en moyenne de meilleure qualité. Ainsi, l'objectif visé par cette étude est de tenter de déterminer la meilleure combinaison de la valeur des paramètres pour que notre programme résolve l'instance du réseau NSF le plus efficacement possible. Pour ce faire, nous allons effectuer plusieurs expériences où nous fixons tous les paramètres sauf un que nous allons faire varier. Nous ferons ensuite la moyenne de la valeur objective de la meilleure solution sur plusieurs exécutions pour chaque valeur étudiée du paramètre considéré. Nous analyserons ensuite les résultats pour en déduire la valeur qui semble la plus appropriée pour ce paramètre. Rappelons que cette étude de sensibilité n'est valable que pour l'instance du réseau NSF ; le choix des valeurs à affecter aux paramètres de l'heuristique est propre à chaque instance. Ainsi, pour un autre réseau ou même une autre instance de trafic du réseau NSF, il convient de refaire cette recherche puisque les valeurs

qui donneront des résultats optimaux varient selon plusieurs éléments comme la connexité du réseau, la taille du trafic ou encore l'hétérogénéité de la matrice de trafic.

Pour ce qui est des exécutions à effectuer, nous prendrons la plupart du temps la moyenne sur un certain nombre d'exécutions où le nombre de longueurs d'onde disponibles varie entre 16 et 25. Nous choisissons de faire varier le nombre de longueurs d'onde puisque c'est l'instance étudiée qui influence le plus le choix des paramètres. Notre objectif est de résoudre l'instance NSF avec différents paramètres de résolution tel le nombre de longueurs d'onde et le nombre de sauts optiques maximal permis.

L'étude que nous présentons dans cette section a été effectuée dans le contexte où nous ne tenons pas compte ni de la compensation, ni de la possibilité de changer de longueur d'onde. De plus, l'arrangement des cartes est toujours effectué de façon optimale (voir section 3.2.2). Dans tous les cas, les exécutions sont arrêtées après 4000 itérations. Voici maintenant les valeurs par défaut des paramètres. L'étude que nous allons effectuer va tenter de raffiner le choix des valeurs pour tous ces paramètres.

```
number_of_element_in_tabu_list = 50
number_of_extra_length_for_considering_path = 3.2
maximum_number_of_deterioration = 2
maximum_number_of_stagnation = 7
validity_ratio_minimum = 0.5
```

Le nombre d'éléments dans la liste taboue

Il est reconnu qu'il faut généralement avoir une liste taboue représentant environ 10 à 15 % des mouvements possibles [6]. Or, nous savons que pour l'instance NSF, le nombre minimum de flots que peut avoir une solution est 269 flots ; et dans l'heuristique, ce sont les flots qui sont rendus tabous une fois qu'un mouvement a été effectué. Il faudrait donc une liste taboue pouvant contenir environ une quarantaine d'éléments ou plus si nous

considérons que le nombre de flots que possède une solution devient généralement légèrement supérieur au minimum au fil des itérations.

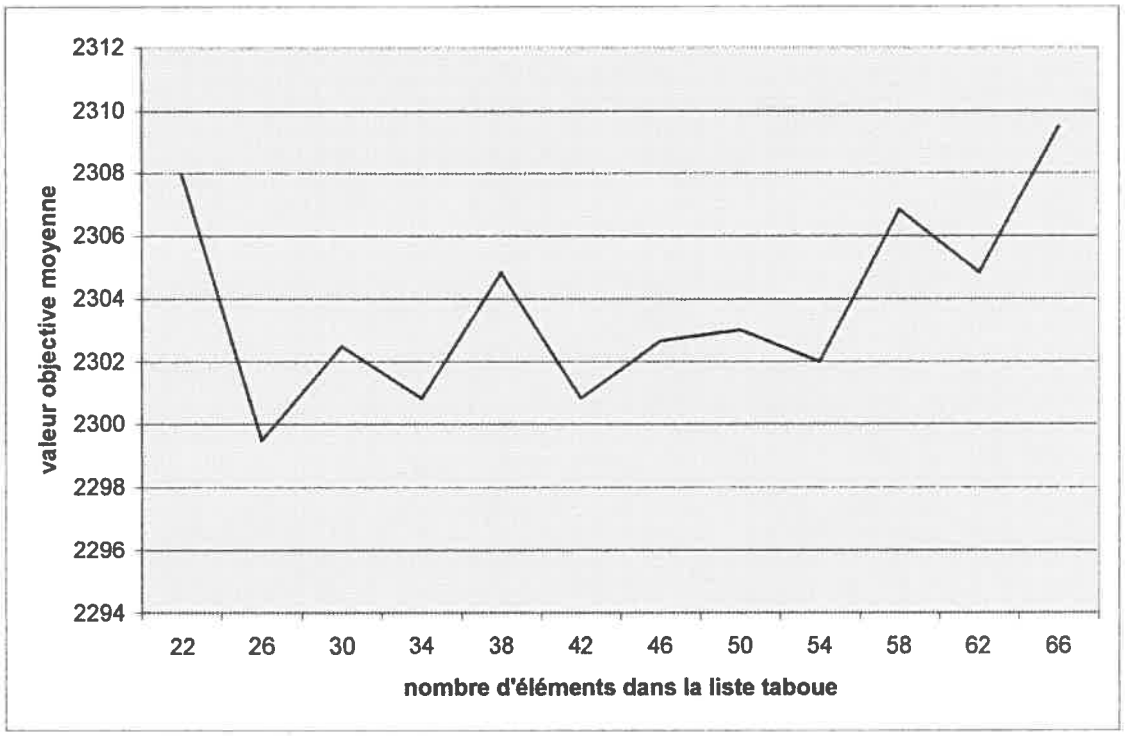

Figure 22 - L'impact du nombre d'éléments dans la liste tabou

Chaque point du graphique de la Figure 22 a été obtenu en calculant la moyenne du coût de la meilleure solution de 12 exécutions où seul le paramètre `random_seed` était différent. Le nombre de longueurs d'onde a été fixé à 18 et un maximum de 3 interruptions par flot était permis. Ce que nous constatons à première vue est que ce paramètre semble avoir peu d'influence sur la qualité de la meilleure solution trouvée. Le coût moyen varie entre 2300 et 2310, soit le coût d'une carte OC-192. De plus, nous avons testé des valeurs allant de 22 éléments (8%) à 66 éléments (25%) ; mais nous n'avons pas testé de valeurs extrêmes. Une liste taboue trop petite génère des cycles répétitifs dans la recherche alors

qu'une liste trop grande limite trop les choix possibles ce qui, dans les deux cas, limite généralement la qualité des meilleures solutions trouvées. À la lumière de ces résultats, nous concluons donc qu'une liste taboue pouvant contenir entre 10 et 20% des flots est un bon choix.

Nous trouvons toutefois cette plage de valeurs un peu trop grande malgré l'effort de calcul mis dans cette expérimentation. Remarquons cependant que nous avons effectué cette étude dans un contexte où l'heuristique taboue n'est pas limitée à un seul type de mouvements possibles. Ainsi, deux mouvements A élémentaires qui regardent deux nœuds différents étudient un sous-ensemble de flots complètement différents. Aussi certains mouvements tels le mouvement qui force une solution irréalisable à redevenir réalisable ne tient pas compte de la liste taboue dans son choix de mouvement (bien qu'elle la mette à jour) ; cet aspect devrait être plus approfondi pour pouvoir définir des expériences permettant de mieux analyser la meilleure valeur à donner à ce paramètre.

Le paramètre `validity_ratio_minimum`

La non réalisabilité est ce qui permet à l'heuristique de voyager d'un voisinage à un autre, et plus la séquence de solutions irréalisables est longue, plus les autres voisinages explorés sont éloignés. Le paramètre `validity_ratio_minimum` permet, de façon globale, de gérer le nombre d'itérations passées dans le domaine non réalisable. C'est par ce facteur que la pénalité associée à la non réalisabilité est multipliée la première fois qu'une solution irréalisable est rencontrée. Ainsi, plus celui-ci est petit, moins la non réalisabilité est pénalisée lorsqu'elle survient dans une solution. Rappelons qu'en plus du paramètre `validity_ratio_minimum`, il y a aussi les paramètres `validity_ratio_maximum`, `validity_ratio_increase_speed` et `validity_ratio_decrease_speed` qui permettent, tous ensemble, de gérer cet état ; mais pour simplifier notre étude, nous nous intéresserons uniquement au premier paramètre sans remettre en question les valeurs des trois autres paramètres.

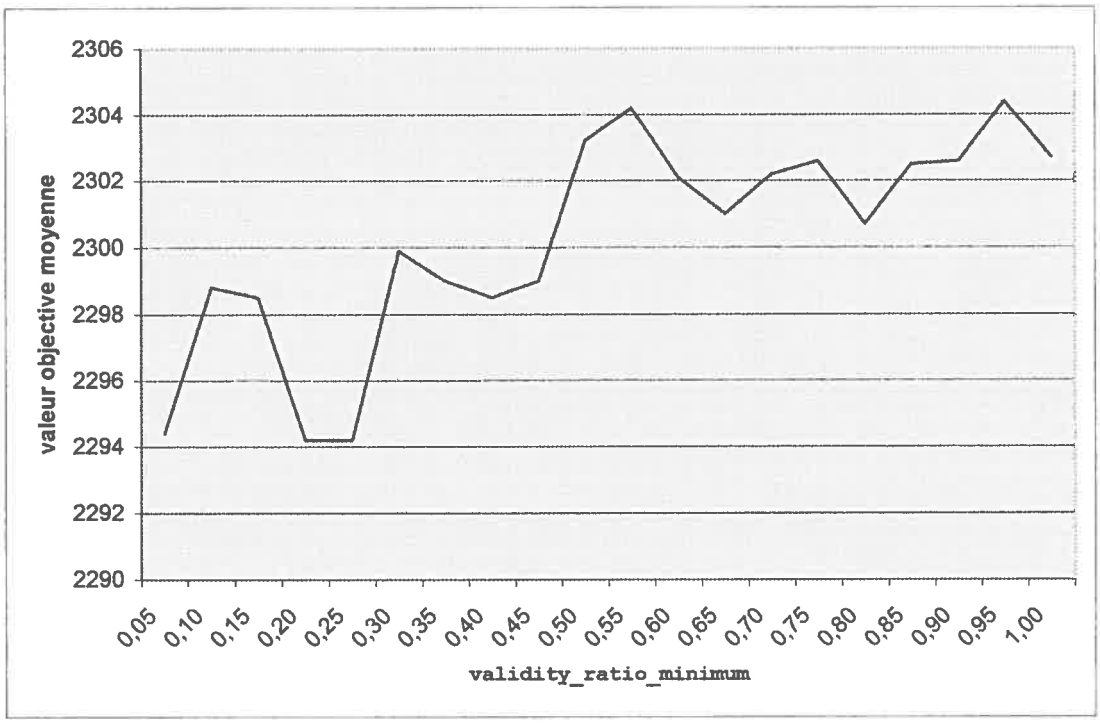

Figure 23 - L'impact du paramètre `validity_ratio_minimum`

Le graphique de la Figure 23 présente un graphique où chaque point correspond à la moyenne du coût de la meilleure solution de 20 exécutions. Pour chacune des 20 exécutions, nous avons fait varier le nombre de longueurs d'onde entre 16 et 25 ; et le nombre maximal d'interruptions a été fixé à 2, puis à 3 pour chaque nombre de longueurs d'onde. Encore une fois, il n'y a pas de tendance marquante qui ressort de ce graphique étant donné que la valeur objective moyenne oscille entre 2294 et 2304. Il convient toutefois de mentionner qu'une petite valeur (inférieure à 0,50) pour le paramètre `validity_ratio_minimum` semble préférable. Ainsi, nous pouvons conclure de ces résultats qu'il est légèrement avantageux de rester irréalisable plus longtemps pour explorer des voisinages plus éloignés les uns des autres.

L'étude de valeurs plus extrêmes en combinaison avec l'étude des effets des trois autres paramètres en rapport avec celui-ci pourrait certainement amener à des conclusions plus éclairées.

Le paramètre `number_of_extra_length_for_considering_path`

Analysons maintenant l'effet du paramètre qui gère le nombre de chemins à considérer lorsqu'un mouvement est effectué. Plus la valeur affectée à ce paramètre est petite, moins il y a de chemins considérés mais cela est compensé par une plus grande vitesse d'exécution.

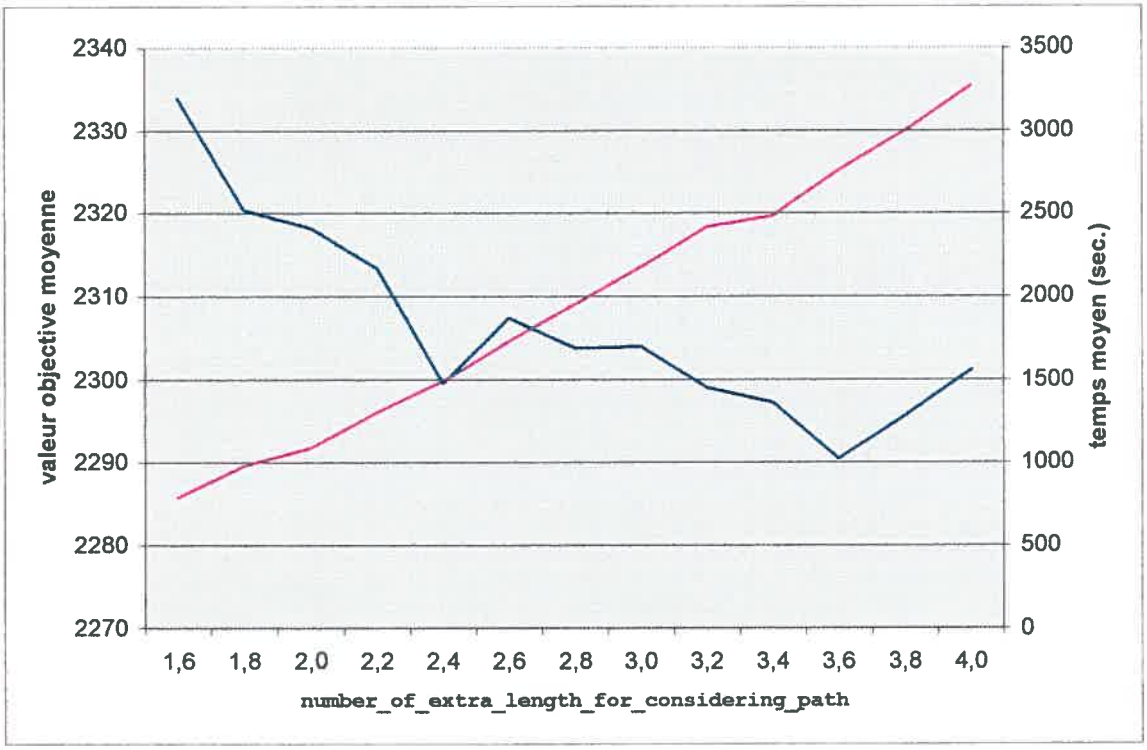

Figure 24 - L'impact du paramètre qui gère le nombre de chemins à considérer

Les points de la courbe bleue du graphique de la Figure 24 correspondent à la moyenne du coût de la meilleure solution de 10 exécutions. Pour chacune des 10 exécutions, le nombre de longueurs d'onde varie entre 16 et 25. Les points de la courbe rouge quant à eux indiquent la moyenne des temps nécessaires pour effectuer les 4000 itérations. Ce graphique démontre clairement ce à quoi nous devions nous attendre : puisque le critère d'arrêt concerne le nombre d'itérations, il est logique que plus nous considérons de chemins possibles pour un mouvement, meilleures sont les solutions obtenues mais plus le temps de calcul est important. Nous concluons de cette expérience qu'il convient d'affecter au paramètre `number_of_extra_length_for_considering_path` une valeur assez grande pour considérer assez de chemins mais pas trop grande non plus pour ne pas faire augmenter le temps de calcul inutilement.

Les paramètres qui gèrent l'effort passé sur un nœud

Ce test a pour objectif de déterminer les meilleures valeurs à affecter aux paramètres `maximum_number_of_deterioration` et `maximum_number_of_stagnation`. Ces paramètres, rappelons-le, représentent en quelque sorte l'effort à passer sur un nœud lorsqu'un mouvement A élémentaire est effectué. Plus leur valeur est grande, plus il y aura d'itérations consécutives concentrées sur ce nœud.

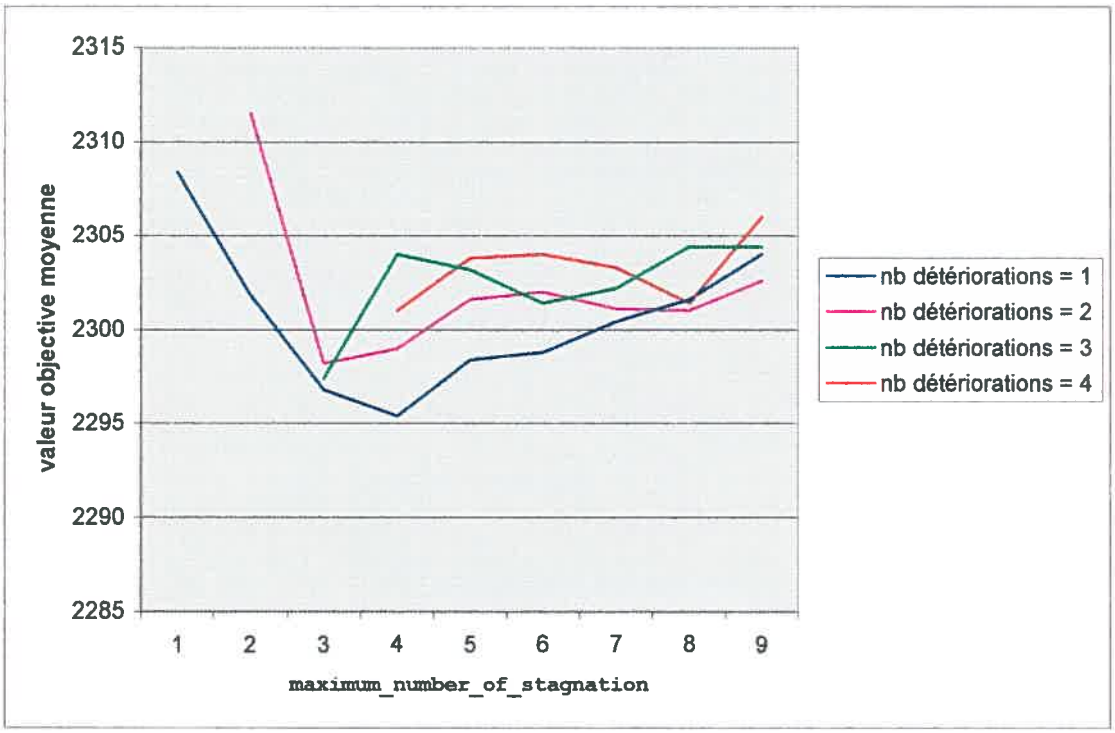

Figure 25 - L'impact des paramètres qui gèrent l'effort sur un nœud

Le graphique de la Figure 25 présente quatre courbes superposées. La première courbe (en bleu foncé) présente les points correspondants aux exécutions où la valeur du paramètre `maximum_number_of_deterioration` est fixée à 1 et la valeur du paramètre `maximum_number_of_stagnation` varie avec l'axe des abscisses ; le même principe s'applique pour les trois autres courbes avec le paramètre `maximum_number_of_deterioration` variant de 2 à 4. Tous les points correspondent à une moyenne sur 10 exécutions où le nombre de longueurs d'onde varie entre 16 et 25. Notons également que la valeur `maximum_number_of_stagnation` est toujours supérieure ou égale à celle du paramètre `maximum_number_of_deterioration`, ceci parce qu'une détérioration est aussi considérée comme un effet stagnant. Pour ce qui est des résultats de cette expérimentation, nous ne pouvons pas conclure beaucoup de choses si ce n'est que ces paramètres n'ont pas beaucoup d'influence sur la meilleure solution trouvée à long terme.

Le choix du scénario

Maintenant, l'objectif est de déterminer lesquels des quatre scénarios et des quatre critères d'ordonnancement des noeuds sont les plus efficaces. Jusqu'à maintenant, nous avions toujours utilisé le critère A et le scénario A. Pour cette expérience, nous avons testés les 16 cas possibles en 10 exécutions où nous avons fait varier le nombre de longueurs d'onde entre 16 et 25. Les valeurs dans la Table 12 sont une moyenne des meilleures valeurs objectives de ces 10 exécutions.

	scénario A	scénario B	scénario C	scénario D	
critère A	2318,4	2317,4	2306,8	2303,8	2311,6
critère B	2307,0	2304,2	2307,6	2298,8	2304,4
critère C	2329,2	2323,4	2311,6	2302,8	2316,8
critère D	2311,4	2309,0	2305,2	2302,8	2307,1
	2316,5	2313,5	2307,8	2302,1	

Table 12 - L'impact du choix du scénario et du critère d'ordonancement

Les valeurs inscrites sur la ligne du bas sont les moyennes pour chacun des scénarios et les valeurs dans la colonne de droite sont les moyennes pour chacun des critères. Dans tous les cas, ces valeurs représentent en fait la moyenne de 40 exécutions.

Encore une fois, l'écart observé pour toutes les possibilités est très faible. Ceci est dû au fait que les scénarios sont quand même relativement semblables entre eux, il n'y a pas de différences majeures entre eux. Les critères de sélection des nœuds quant à eux, bien qu'ils soient différents, partagent quand même la même philosophie : privilégier les nœuds où il y a beaucoup de bande passante inutilisée et beaucoup de bande passante utilisée pour des interruptions. Ainsi, nous conclurons que le choix du critère de sélection des nœuds et du scénario, parmi ceux proposés, a peu d'impact sur la performance de l'heuristique.

Remarquons toutefois qu'en observant strictement les valeurs de la Table 12, nous trouvons que la meilleure combinaison est : le scénario D et le critère B. Rappelons que les moyennes sur la ligne du bas et sur la colonne de droite sont une moyenne de 40 exécutions. Bien que les différences entre toutes ces valeurs soient petites, nous pouvons quand même les utiliser pour justifier notre choix puisqu'il s'agit quand même d'une moyenne sur un grand nombre d'exécutions. Nous trouverions probablement des différences plus marquées si des scénarios et des critères plus variés étaient utilisés.

Le critère pour décider entre deux valeurs identiques pour la fonction d'évaluation

Lorsque nous évaluons deux solutions distinctes avec la fonction d'évaluation et que les deux valeurs pour l'évaluation sont identiques, nous utilisons le hasard pour trancher entre les deux possibilités. Toutefois, nous avons expérimenté un autre critère pour choisir : le nombre d'interruptions contenues dans la solution au complet. Nous avons tenté de voir ce qui se produisait lorsque les interruptions sont pénalisées, puis lorsqu'elles sont au contraire encouragées.

une solution avec beaucoup d'interruptions est pénalisée	2300,0
utiliser le hasard pour discerner deux évaluations identiques	2298,8
une solution avec peu d'interruption est pénalisée	2300,2

Table 13 - L'impact du choix du critère secondaire

Les valeurs de la Table 13 sont une moyenne de 10 exécutions où le nombre de longueurs d'onde varie entre 16 et 25. Ce tableau nous montre que le choix du critère n'a pratiquement pas d'importance. La raison est qu'il existe beaucoup de solutions équivalentes ayant un nombre égal d'interruptions moyen. Ainsi, nous ne tiendrons plus

compte de cet aspect et nous prendrons pour acquis que le critère secondaire utilisé sera toujours le hasard ; mais nous pouvons noter qu'il est souvent possible de trouver des solutions équivalentes où le nombre d'interruptions peut être différent.

En conclusion générale sur les analyses de sensibilité des valeurs des paramètres, nous pouvons affirmer que pour les expérimentations présentées, les intervalles considérés (si nous excluons quelques valeurs extrêmes) fournissent tous des paramètres adéquats pour la performance de l'heuristique GRWABOU.

6.1.2 Analyse du dimensionnement GRWA de l'instance NSF

Nous allons maintenant utiliser les résultats obtenus de la section précédente pour examiner l'impact de certains paramètres de base (tel le nombre de longueurs d'onde et le nombre maximal d'interruptions) sur le dimensionnement obtenu par l'heuristique GRWABOU.

L'expérience que nous présentons dans les pages suivantes a pour objectif de montrer l'évolution de la fonction objective en fonction du nombre de longueurs d'onde et du nombre maximal d'interruptions autorisées. Les paramètres communs à toutes les exécutions sont définis ci-dessous. Notons aussi que le critère d'arrêt a été fixé à 40000 itérations.

```
number_of_element_in_tabu_list = 42
number_of_extra_length_for_considering_path = 3.2
maximum_number_of_deterioration = 1
maximum_number_of_stagnation = 4
validity_ratio_minimum = 0.25
```

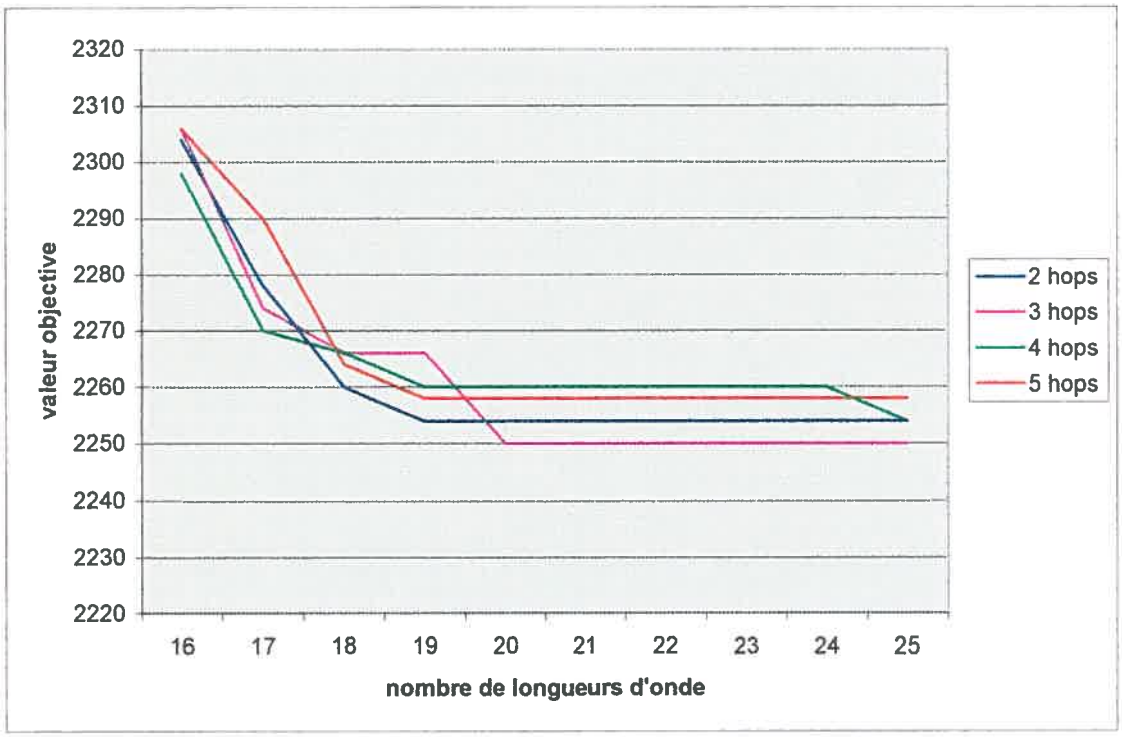

Figure 26 - Évolution de la fonction objective selon le nombre de longueurs d'onde

Pour obtenir les résultats présentés à la Figure 26, nous avons lancé 4 séries d'exécutions, une pour chaque nombre maximal de sauts optiques permis. Pour chaque série, une solution initiale a été construite pour 16 longueurs d'onde, puis pour 17 longueurs d'onde, la solution initiale utilisée a été la meilleure solution issue de l'exécution précédente (celle avec 16 longueurs d'onde). Nous avons procédé de la même façon jusqu'à 25 longueurs d'onde.

Les résultats confirment le fait que lorsque le nombre de longueurs d'onde augmente, le degré de liberté gagné entraîne généralement une amélioration de la meilleure solution trouvée. Toutefois, nous ne pouvons pas en dire autant sur le nombre maximal de sauts optiques. Les quatre courbes présentées sur le graphique sont entrecroisées et elles sont toutes assez proches les unes des autres. Ceci signifie que pour le réseau NSF, avec

son instance de trafic, il existe plusieurs solutions à coûts équivalents tels que le nombre moyen de sauts optiques est différent. Le degré de liberté engendré par l'augmentation du nombre d'interruptions maximal permet d'admettre un plus grand nombre de solutions dans le domaine réalisable mais cela n'implique pas pour autant que ces solutions soient particulièrement bonnes.

Notons aussi que même si le nombre de sauts optiques est indirectement minimisé à travers la minimisation du nombre de cartes, cela ne suffit pas pour favoriser l'obtention de solutions avec un nombre minimum de sauts optiques. À coût équivalent, de telles solutions seraient préférées par un concepteur de réseau. Il serait en conséquence opportun d'introduire une fonction auxiliaire comptant explicitement le nombre de sauts pour favoriser l'obtention des solutions à coût minimum avec un nombre minimum de sauts optiques.

Remarquons comme dernier point que la borne inférieure pour ce problème est de 1946. Or, la meilleure solution trouvée ici (pour un maximum de 3 sauts optiques et de 20 longueurs d'onde) possède un coût de 2250 soit seulement 14.6% de plus que la borne inférieure. Dans le but de raffiner l'analyse de la qualité de cette solution, regardons une métrique supplémentaire : le taux de remplissage moyen des ports. Or pour cette solution, nous trouvons que ce taux de remplissage est de 88,8 %. Une autre métrique consiste à calculer la longueur moyenne (en termes de nombre de liens) des chemins utilisés par chacun des flots de la solution par rapport au plus court chemin. Ainsi, pour cette solution, nous trouvons qu'en moyenne un flot parcourt un chemin qui possède 0,32 lien de plus que le plus court chemin. Notons cependant que cette dernière métrique ne peut pas être utilisée comme indicateur de qualité absolue puisque d'une part, elle est très dépendante de la connexité du réseau et, d'autre part, lorsqu'un seul saut optique est effectué, un nombre a priori arbitraire de liens peut être parcouru. Cette analyse nous permet ainsi de mieux juger de la qualité de cette solution. Donc puisque les valeurs trouvées sont assez satisfaisantes, nous pouvons affirmer que la solution analysée ici est de qualité très raisonnable.

Maintenant, pour comprendre un peu mieux le fonctionnement interne de l'heuristique, analysons plus en détail une exécution particulière : celle avec 17 longueurs d'onde et un maximum de 5 sauts optiques.

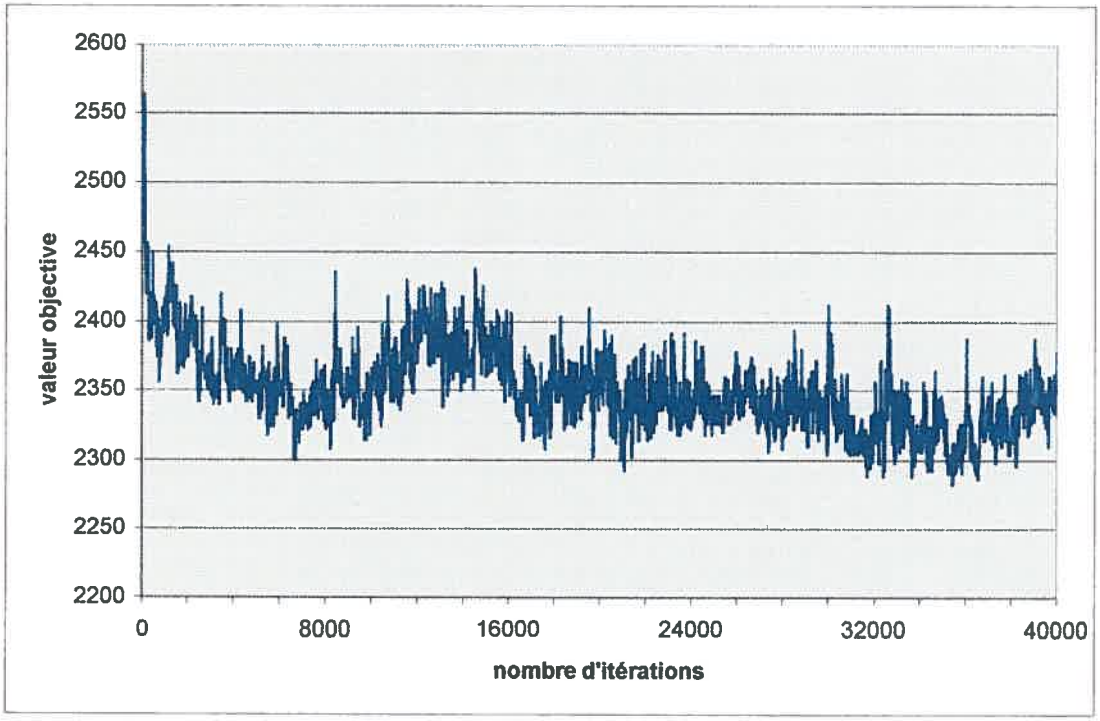

Figure 27 - Évolution de la fonction objective

La Figure 27 nous montre l'évolution de la valeur objective de la solution au fil des itérations. La solution retenue a été trouvée à l'itération 35420 et a un coût de 2290.

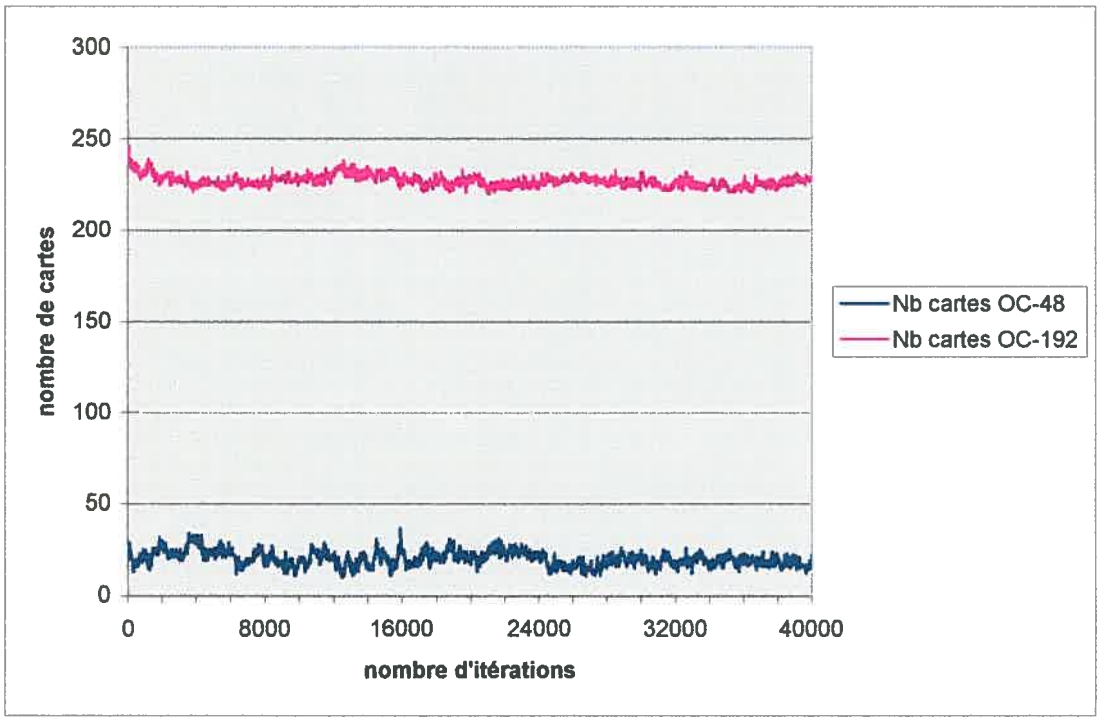

Figure 28 - Évolution du nombre de cartes OC-48 versus OC-192

Le graphique à la Figure 28 illustre l'évolution du nombre de cartes OC-48 versus OC-192. Nous voyons que le nombre de cartes OC-192, comme le nombre de cartes OC-48 dans les solutions reste relativement stable tout au long de la recherche.

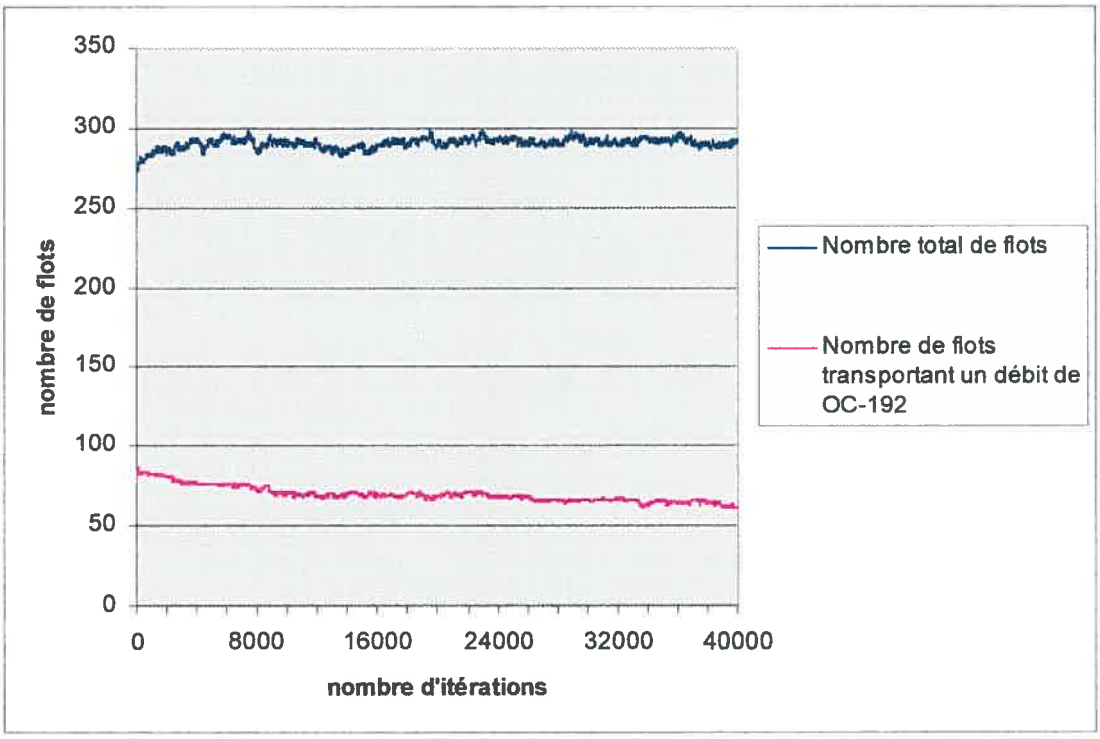

Figure 29 - Évolution du nombre de flots

Ici, à la Figure 29, nous voyons la variation du nombre de flots au fil des itérations. La solution initiale possède 269 flots mais nous voyons que ce nombre tend à augmenter au début pour ensuite se stabiliser autour de 290. Ceci est entre autre dû au fait que plusieurs flots sont coupés en flots de débit plus petit pour pouvoir entre autre bénéficier des avantages de l'utilisation de la capacité de transport OC-48 lorsque cela est possible. Ensuite, nous observons le comportement inverse pour le nombre de flots transportant un débit d'OC-192. Ceci indique que l'heuristique a parfois jugé avantageux de couper des flots qui transportent le débit maximum, pour parfois les recombiner avec d'autres par la suite.

Nous allons maintenant faire une expérimentation concernant la stratégie d'arrangement des cartes. Pour cette expérience, nous avons procédé à plusieurs exécutions

paramétrées de la même façon que pour les exécutions présentées à la Figure 26 sauf que le nombre de sauts optiques a été fixé à 3 et nous avons testé les trois façons d'organiser les cartes de transport (voir la section 3.2.2).

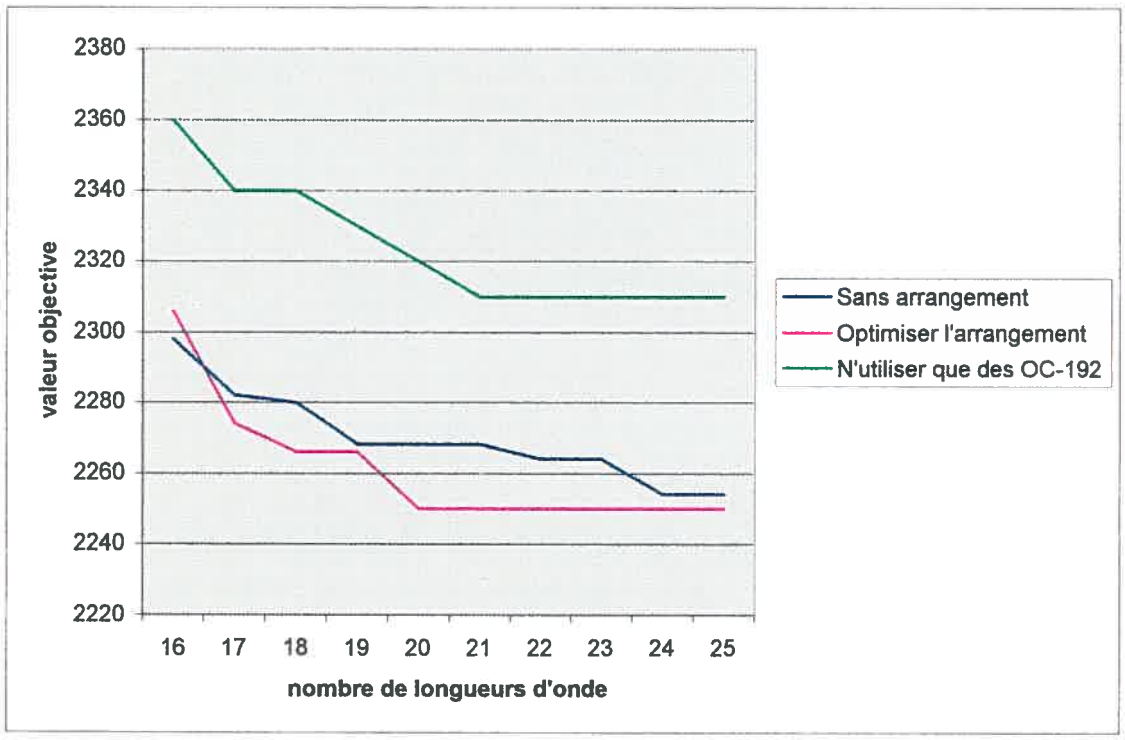

Figure 30 - Évolution selon l'arrangement des cartes

Le graphique à la Figure 30 présente ces trois courbes. La courbe où nous optimisons l'arrangement (la courbe en rouge) est la même que celle de la Figure 26 où le nombre de sauts optiques est fixé à 3. Pour ce qui est de la façon d'organiser les cartes, il s'agit de celle des trois courbes qui présente les valeurs les plus intéressantes. Lorsque nous ne faisons aucun effort pour arranger les cartes OC-48 et OC-192, nous voyons que les solutions sont tout de même appréciables mais l'écart entre les deux courbes, bien que modeste, nous confirme qu'il est avantageux de faire cette optimisation. Ce qui est logique

puisque la procédure d'optimisation que nous proposons ne peut jamais engendrer une augmentation du coût d'une solution. Toutefois, lorsque nous interdisons l'utilisation de ports OC-48, l'écart se creuse de façon évidente. Nous pouvons de ce fait confirmer qu'il est avantageux de considérer simultanément l'utilisation de cartes OC-48 et de cartes OC-192 ; mais l'importance relative d'un groupe de cartes par rapport à l'autre reste dépendante de la nature du trafic.

6.1.3 Analyse du dimensionnement GRWA de l'instance NSF en tenant compte de la possibilité de changer de longueur d'onde

Nous allons maintenant regarder ce qui se produit lorsque nous utilisons l'algorithme qui permet de changer de longueur d'onde pour les différents segments d'un flot (voir section 3.3). Commençons par observer le graphique de la Figure 31.

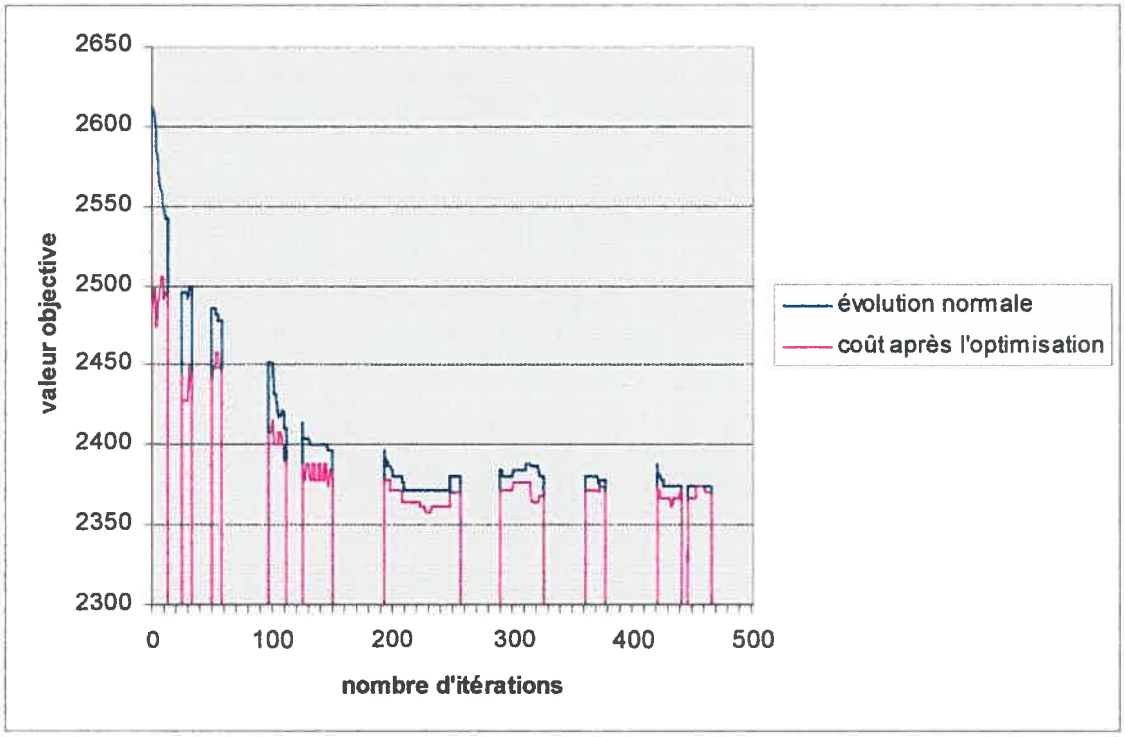

Figure 31 - Évolution des solutions post-optimisées à chaque itération

L'expérience décrite ici consiste à lancer une exécution avec 18 longueurs d'onde et un maximum de 4 sauts optiques, le reste des paramètres est toujours fixé aux valeurs proposées au début de la section 6.1.2, sauf que le critère d'arrêt est de 500 itérations. La courbe intitulée « évolution normale » en bleu foncé correspond à la courbe de l'évolution de la valeur objective au fil des itérations avec la contrainte de continuité de longueurs d'onde. À chaque itération, avant de procéder à la suivante comme d'habitude, nous procédons à une « post-optimisation » de la solution courante avec l'algorithme proposé à la section 3.3 qui permet de relaxer la contrainte sur la continuité de longueurs d'onde. Les points de la deuxième courbe (en mauve) représentent le coût de ces solutions optimisées. Notons que dans le graphique, il n'y a pas de point pour les solutions lorsqu'elles ne sont pas réalisables.

Ce que nous constatons à la lumière de ces résultats est que l'algorithme permettant d'effectuer des changements de longueur d'onde sur une solution produit bel et bien des solutions moins coûteuses ou au pire, de même coût. Nous constatons aussi que plus la solution que nous voulons optimiser est de mauvaise qualité, plus l'algorithme est performant. Nous remarquons effectivement que pour les premières itérations où les solutions sont mauvaises, l'optimisation proposée réduit le coût de 50 à 100 unités. Alors qu'après un certain nombre d'itérations, lorsque les solutions deviennent de meilleure qualité, le gain apporté par cette transformation est plus modeste.

Nous allons maintenant effectuer une seconde expérience. Nous allons reprendre la courbe de la Figure 26 où le nombre de sauts optiques est fixé à 3. Rappelons que les points de cette courbe représentent les meilleures solutions obtenues après 40000 itérations pour un nombre de longueurs d'onde allant de 16 à 25. Ainsi, nous avons repris chacune de ces dix solutions, puis nous y avons appliqué l'algorithme de la section 3.3 qui permet de relaxer la contrainte sur la continuité de longueurs d'onde. Le graphique de la Figure 32 présente ces résultats.

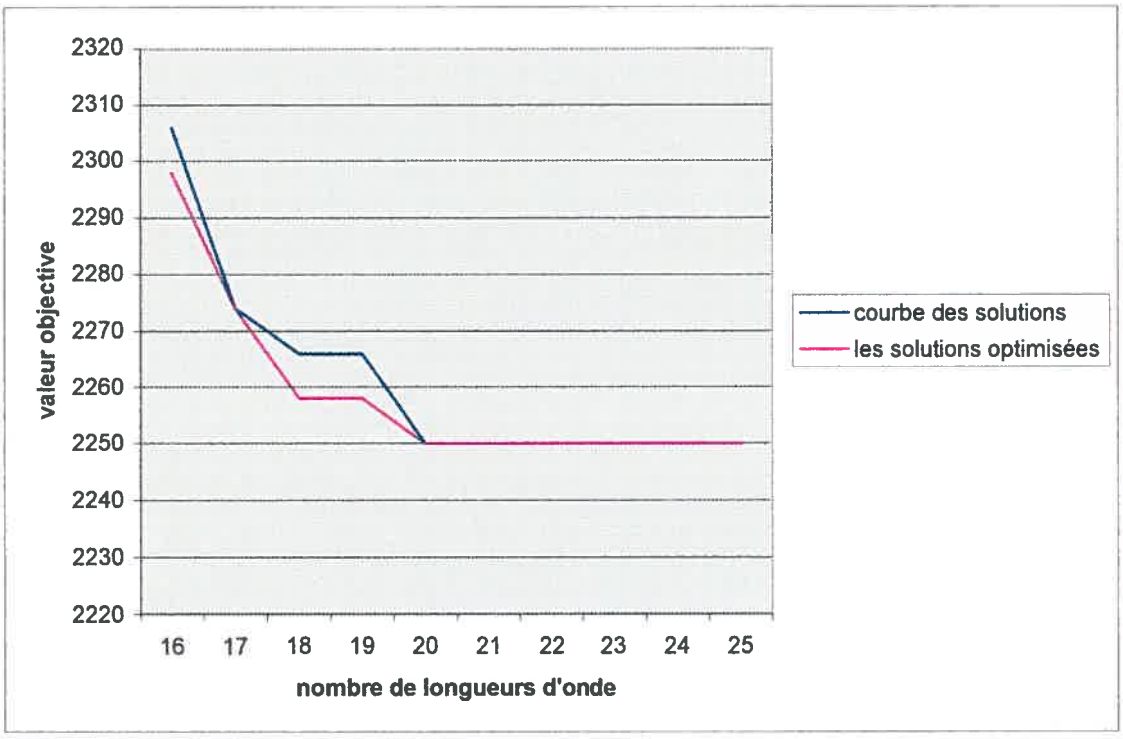

Figure 32 - L'impact de la post-optimisation pour chaque solution

La courbe des solutions en bleu foncé correspond à la même courbe que celle du graphique de la Figure 26. Les points de la courbe des solutions optimisées en rouge sont les valeurs objectives des solutions correspondantes une fois qu'elles ont été optimisées par l'algorithme.

Les conclusions que nous pouvons tirer de cette observation s'apparentent aux conclusions de l'expérience précédente. C'est-à-dire que puisque nous optimisons des solutions de très bonne qualité, le gain apporté par cette optimisation est très faible, voire même nul parfois. Notons que même si elles sont de même coût, les solutions optimisées peuvent différer de la solution originale.

6.1.4 Étude de sensibilité des paramètres en tenant compte de la compensation

Comme nous l'avons fait précédemment, nous allons étudier les effets de différents paramètres sur l'instance NSF où, cette fois, nous tenons compte de la compensation. Puisque l'instance étudiée ici est assez semblable au cas où nous ne tenons pas compte de la compensation, nous prenons pour acquis une bonne partie de l'étude effectuée précédemment. Nous allons en fait étudier deux paramètres propres à l'étude de la compensation et nous allons refaire les tests pour le paramètre `validity_ratio_minimum`. Rappelons que le critère sur la continuité de longueurs d'onde est toujours en vigueur et que, l'arrangement des cartes est toujours effectué de façon optimale.

Avant d'entrer dans les détails, il convient de mentionner que nous avons effectué une petite modification sur le réseau : nous avons divisé par 40 les longueurs de tous les liens du réseau. La raison est que la compensation pour une capacité de transport d'OC-192 devient nécessaire lorsque le signal optique traverse plus de 80 kilomètres de fibres optiques. Or, sans cette modification, tous les liens avait une longueur supérieure à cette valeur ce qui rendait l'étude de la compensation peu intéressante. Avec la modification que nous proposons, la longueur moyenne des liens du réseau devient égale à 32,6 km.

Notons une dernière chose avant d'entrer dans les détails des expérimentations subséquentes. La qualité des expérimentations effectuées pour cette instance est plus modeste que pour l'instance où la compensation n'est pas considérée. C'est-à-dire que nous considérons moins de points dans les graphiques et nous produisons moins d'exécutions pour obtenir chaque point des graphiques. La raison est que de considérer la compensation comme cela est proposé dans la section 4.2 coûte cher à optimiser en termes de temps de calcul. Une itération est de l'ordre de 40 fois plus coûteuse en temps de calcul

si la compensation est considérée que si elle ne l'est pas. C'est pourquoi il nous a fallu réduire le nombre d'exécutions pour effectuer ces expériences.

Le paramètre `validity_ratio_minimum`

Nous choisissons de refaire l'étude de ce paramètre parce qu'en considérant la compensation pour cette instance, il est possible que le fait de rester irréalisable trop ou pas assez longtemps ait un effet différent que si nous ne considérons pas la compensation.

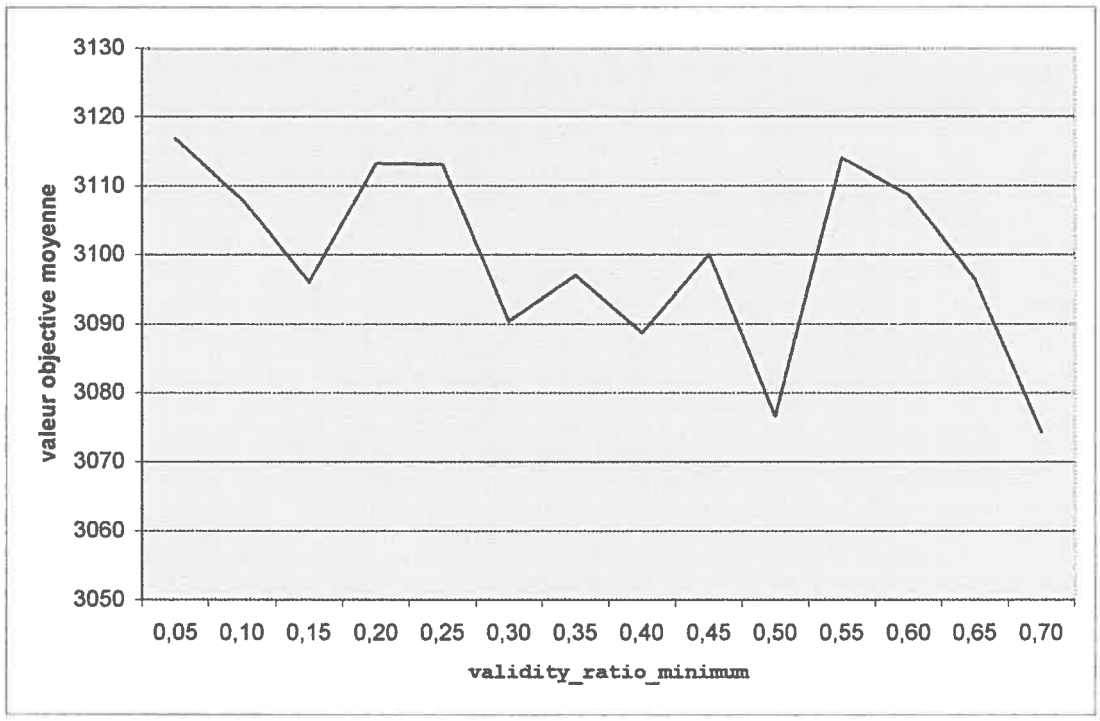

Figure 33 - L'impact du nombre d'éléments dans la liste tabou

Pour le graphique illustré à la Figure 33, les points sont la moyenne de 6 exécutions où le nombre de longueurs d'onde varie entre les valeurs {16, 17, 20, 21, 24, 25}.

Malheureusement, nous ne pouvons pas faire ressortir une tendance sur ce graphique, la valeur objective moyenne oscille trop pour pouvoir conclure quelque chose de constructif. Notons ici que l'écart maximal entre les valeurs présentées est légèrement plus élevé que dans les expérimentations précédentes ; mais il faut garder à l'esprit que le nombre d'échantillons utilisé est plus petit. Nous pouvons toutefois affirmer que pour la plage de valeur testée, le choix du paramètre `validity_ratio_minimum` n'a pas d'impact majeur. Le fait de rester plus ou moins longtemps irréalisable semble avoir très peu d'impact.

La fréquence du mouvement C

Lorsque nous tenons compte de la compensation, il est possible d'effectuer le mouvement C, celui où nous redirigeons plusieurs flots de façon à enlever la nécessité de compenser un lien donné. Or, ce mouvement perturbe énormément la solution lorsqu'il est effectué. Ainsi, dans la boucle principale de chaque scénario, nous choisissons d'effecteur ce mouvement une fois à une certaine fréquence. Puisqu'il ne faut pas le faire trop souvent, nous testons ce qui arrive si nous exécutons ce mouvement une fois sur 3, puis une fois sur 4, et ainsi de suite jusqu'à une fois sur 7.

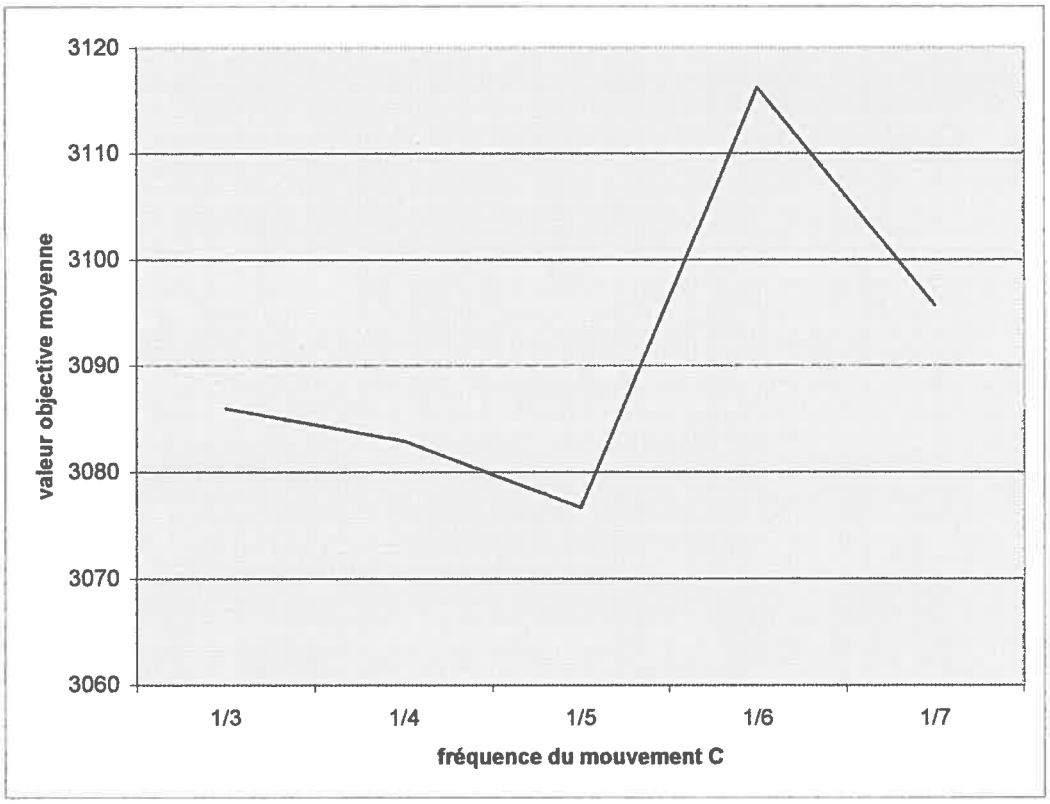

Figure 34 - L'impact de la fréquence du mouvement C

Le graphique de la Figure 34 illustre une courbe où chaque point est la moyenne de 6 exécutions et où le nombre de longueurs d'onde varie de la même façon que pour l'expérience précédente. Encore une fois, la variation est très peu significative ; remarquons que l'effet d'un mouvement C est comparable au fait de voyager pendant un certain nombre d'itérations consécutives dans le domaine irréalisable en ce sens que nous changeons de voisinage. Or, l'explication concernant le fait que la fréquence d'exécution du mouvement C a peu d'impact sur la qualité de la meilleure solution trouvée tient au fait que nous changeons déjà de voisinage relativement régulièrement par un autre mécanisme (le mouvement B). Ainsi, à petite échelle, ce paramètre n'influence pas beaucoup la qualité de la meilleure solution retenue pour les valeurs expérimentées.

Le paramètre `evaluation_ratio_for_compensation_correction`

Le dernier paramètre que nous allons examiner est le paramètre `evaluation_ratio_for_compensation_correction`. Pour deux solutions où le coût de la compensation est identique, la fonction d'évaluation propose une façon de distinguer les deux solutions à partir du nombre de longueurs d'onde qui sont à l'origine de la nécessité de compenser chaque lien qui en a besoin (voir section 5.1.2). Le paramètre que nous étudions ici correspond au poids que nous devons affecter à ce discriminant.

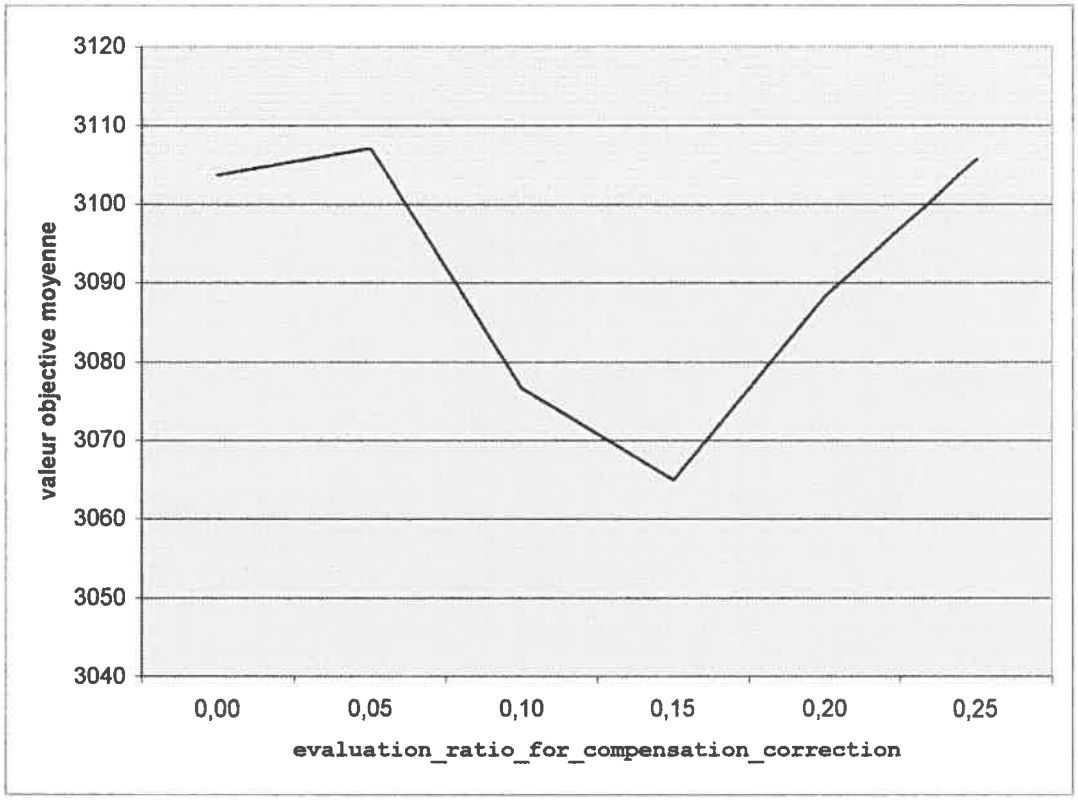

Figure 35 - L'impact du paramètre qui gère l'évaluation de la compensation

Les points du graphique de la Figure 35 sont la moyenne de 6 exécutions de la même façon que pour les deux expérimentations précédentes. Nous nous trouvons encore face au fait que l'écart entre les valeurs est peu significatif : entre le meilleur et le pire point nous trouvons un écart équivalent au coût de 5 cartes de transport OC-192. Toutefois, malgré que cet écart soit faible, la courbe dégage une tendance qui semble logique. Le poids associé au discriminant ne doit pas être nul sans quoi le discriminant ne peut pas avoir d'effet, et il ne doit pas non plus être trop élevé pour ne pas surpasser en valeur les critères primaires (coûts des cartes et coût de la compensation). Ainsi, nous trouvons que la plage de valeur entre 0,10 et 0,15 semble être un bon choix pour la valeur à affecter à ce paramètre.

6.1.5 Analyse du dimensionnement GRWA de l'instance NSF en tenant compte de la compensation

Maintenant, nous retiendrons les résultats obtenus précédemment et nous nous en inspirerons pour fixer les valeurs des paramètres pour résoudre l'instance NSF en considérant la compensation. Ainsi, les paramètres communs à toutes les exécutions de cette section sont définis comme suit :

```
number_of_element_in_tabu_list = 42
number_of_extra_length_for_considering_path = 3.2
maximum_number_of_deterioration = 1
maximum_number_of_stagnation = 4
evaluation_ratio_for_compensation_correction = 0.15
validity_ratio_minimum = 0.5
```

Le critère d'arrêt a été fixé à 6 heures sur l'équivalent d'un Pentium IV 2.4 GHz et la fréquence du mouvement C a été fixée à une fois sur 5.

Pour la première expérience que nous examinons, trois séries d'exécutions ont été lancées. Pour la première série d'exécutions, un maximum de 2 sauts optiques est permis, pour la troisième, un maximum de 3 sauts optiques est permis alors qu'un maximum de 4 sauts optiques est permis pour la dernière. Chacune des courbes est obtenue à partir de 10 points, chacun des points correspond à la meilleure solution trouvée après 4000 itérations pour un nombre de longueurs d'onde qui varie entre 16 et 25. Pour les trois séries d'exécutions, la solution initiale avec 16 longueurs d'onde est générée automatiquement ; alors que pour les exécutions à 17 longueurs d'onde et plus, la solution initiale est la meilleure solution obtenue à partir de l'exécution précédente, c'est-à-dire celle où une longueur d'onde de moins est permise.

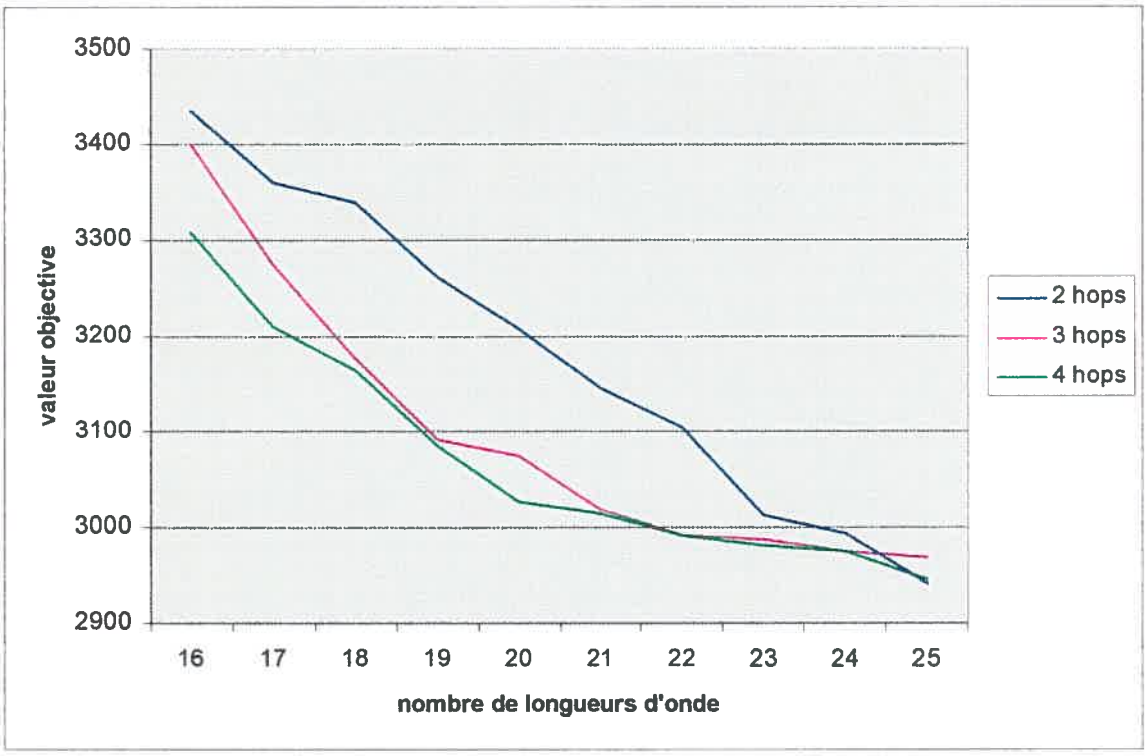

Figure 36 - Évolution de la fonction objective selon le nombre de longueurs d'onde

La Figure 36 illustre ces résultats. Nous constatons tout d'abord que pour les trois courbes, plus le nombre de longueurs d'onde disponibles augmente, meilleure est la valeur objective. Aussi, plus le nombre d'interruptions maximales permise est grand, meilleure est la solution ; cette dernière remarque fait la différence avec la même expérimentation lorsque la compensation n'est pas considérée. Ceci s'explique principalement par le fait que cette marge de manœuvre permet plus souvent d'échanger de la compensation pour des cartes MSPP lorsque cela est avantageux. Notons aussi que la démarcation entre un maximum de 2 et 3 sauts optiques est plus prononcée qu'entre un maximum de 3 et 4 sauts optiques. Nous trouvons donc que lorsque la compensation est considérée pour l'instance NSF, le domaine des solutions réalisables lorsque le nombre maximal de sauts optiques augmente comporte des solutions dont le coût est plus avantageux que si le domaine des solutions était plus restreint.

Maintenant, regardons plus en détail l'exécution avec 16 longueurs d'onde où un maximum de 4 sauts optiques est permis. Notons qu'il y a eu 1324 itérations effectuées en 6 heures et que c'est à l'itération 1309 que la meilleure solution a été trouvée. Pour ce faire, nous analysons plusieurs graphiques qui illustrent différentes métriques.

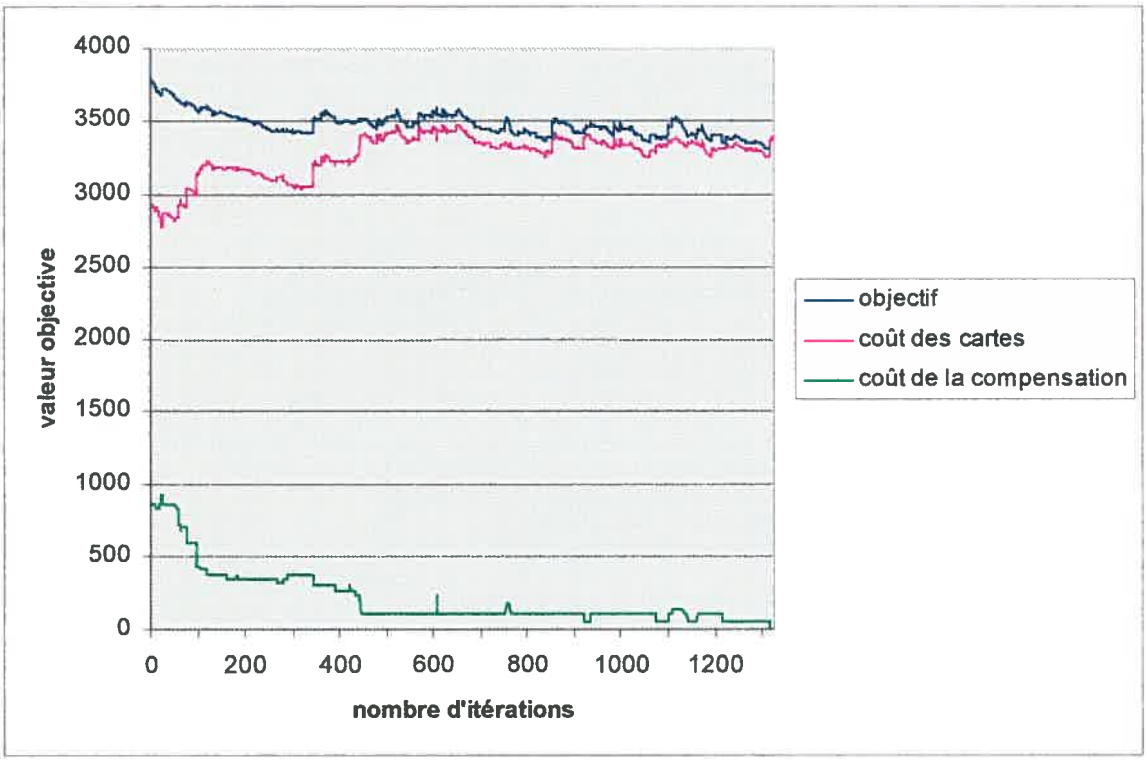

Figure 37 - Exécution avec 16 longueurs d'onde et 4 sauts optiques

Sur le graphique de la Figure 37, nous voyons en bleu foncé l'évolution de la fonction objective au fil des itérations. Nous remarquons aussi la partie de la valeur objective relative aux cartes et celle relative à la compensation. Pour cette exécution, nous constatons que les meilleures solutions sont celles où la compensation est presque entièrement remplacée par l'utilisation de cartes de transport MSPP. Notons que ce comportement est particulier à chaque instance de réseau et de trafic.

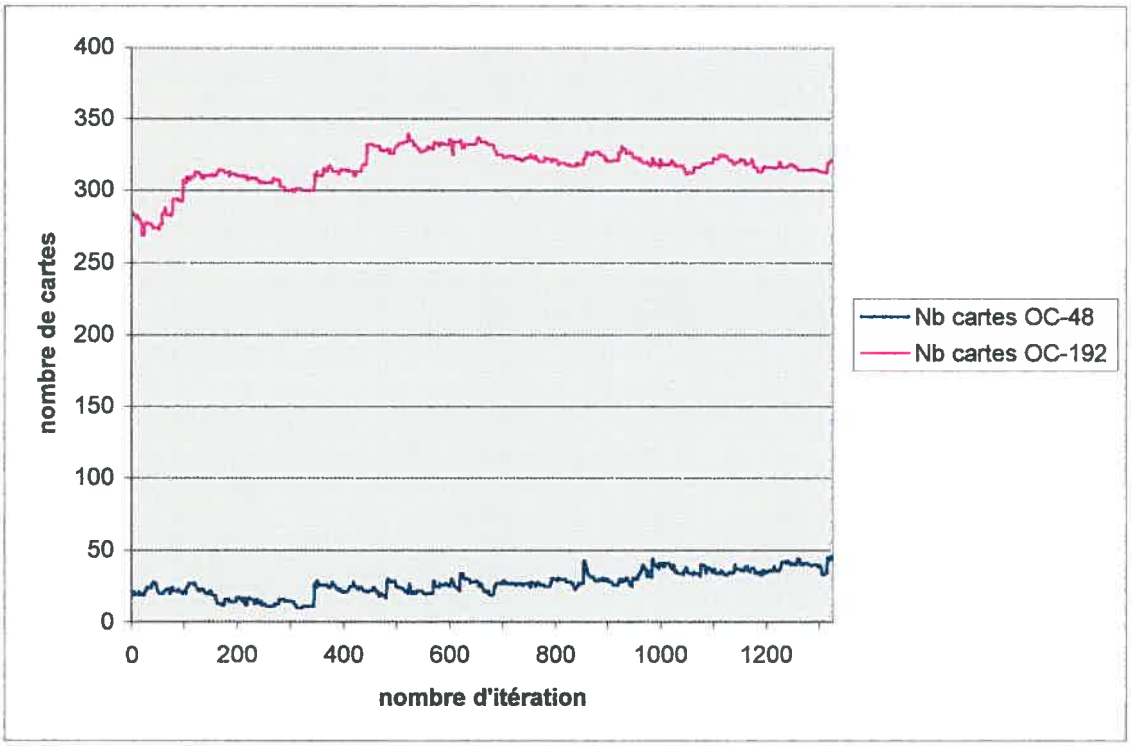

Figure 38 - Évolution du nombre de cartes OC-48 versus OC-192

La Figure 38 présente un autre aspect de l'évolution des solutions au fil des itérations. Il s'agit, pour chaque solution, du nombre de cartes OC-48 utilisées versus le nombre de cartes OC-192 utilisées. Nous constatons bien sûr que le nombre de cartes OC-192 domine largement le nombre de cartes OC-48. Cependant, le nombre de cartes OC-48 tend tout de même à augmenter puisque l'utilisation de la capacité de transport OC-48 permet au signal optique de traverser plus de kilomètres sans avoir à être compensé. Nous voyons aussi que le nombre de cartes OC-192 augmente aussi car nous avons vu dans le graphique de la Figure 37 que le coût des cartes (et donc le nombre de cartes) tend à augmenter.

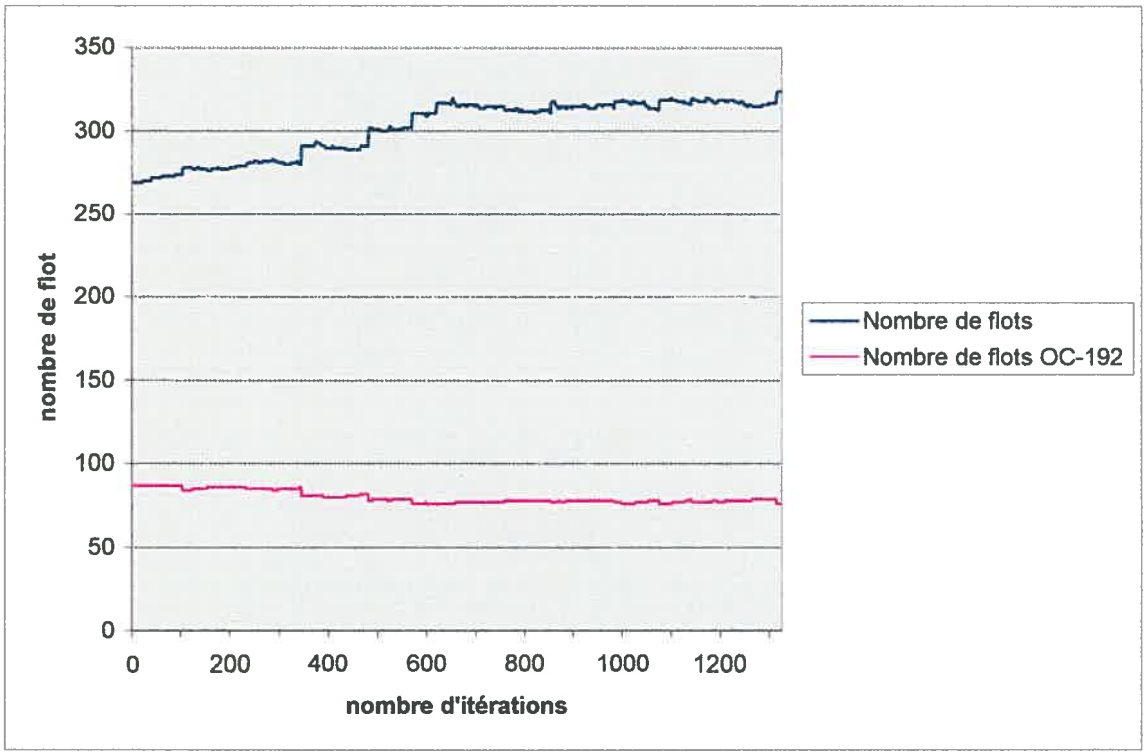

Figure 39 - Évolution du nombre de flots

Dans le graphique à la Figure 39, nous voyons l'évolution du nombre total de flots et du nombre de flots qui ont un débit d'OC-192 (le débit maximal). Le nombre de flots total tend à augmenter de plus en plus, c'est qu'il est avantageux de diviser certains flots en flots de plus faible débit pour pouvoir économiser sur la compensation. Le nombre de flots qui débite l'équivalent d'un OC-192 quant à lui tend à diminuer légèrement, possiblement une conséquence de la division des flots.

Pour conclure sur ces expérimentations, nous analyserons un peu plus en détail la meilleure solution obtenue par l'exécution décortiquée au cours des derniers paragraphes, c'est-à-dire celle avec 16 longueurs d'onde où un maximum de 4 sauts optiques est permis. Nous trouvons que la meilleure solution possède un coût de 3309 où le coût des cartes est de 3256 et le coût de la compensation est de 53. Toutefois, pour la même solution où la

compensation n'est pas considérée, le coût des cartes (et donc le coût total) est de 2796. Cette information nous permet de distinguer plus précisément l'impact de la compensation sur une solution puisqu'en plus des compensateurs, des cartes de transport supplémentaires doivent être ajoutées pour supporter la compensation. Ainsi, nous pouvons conclure que pour cette solution, le coût total de la compensation est de :

$$(3256 - 2796) + 53 = 460 + 53 = 513.$$

Ce résultat signifie que le fait de considérer la compensation pour cette solution a fait grimper le coût de 18,3 %.

6.2 Le réseau EON2004

Le deuxième réseau que nous examinons est le réseau EON2004 (voir [2]) correspondant à un réseau européen qui relie les capitales des différents pays. La Figure 40 illustre ce réseau.

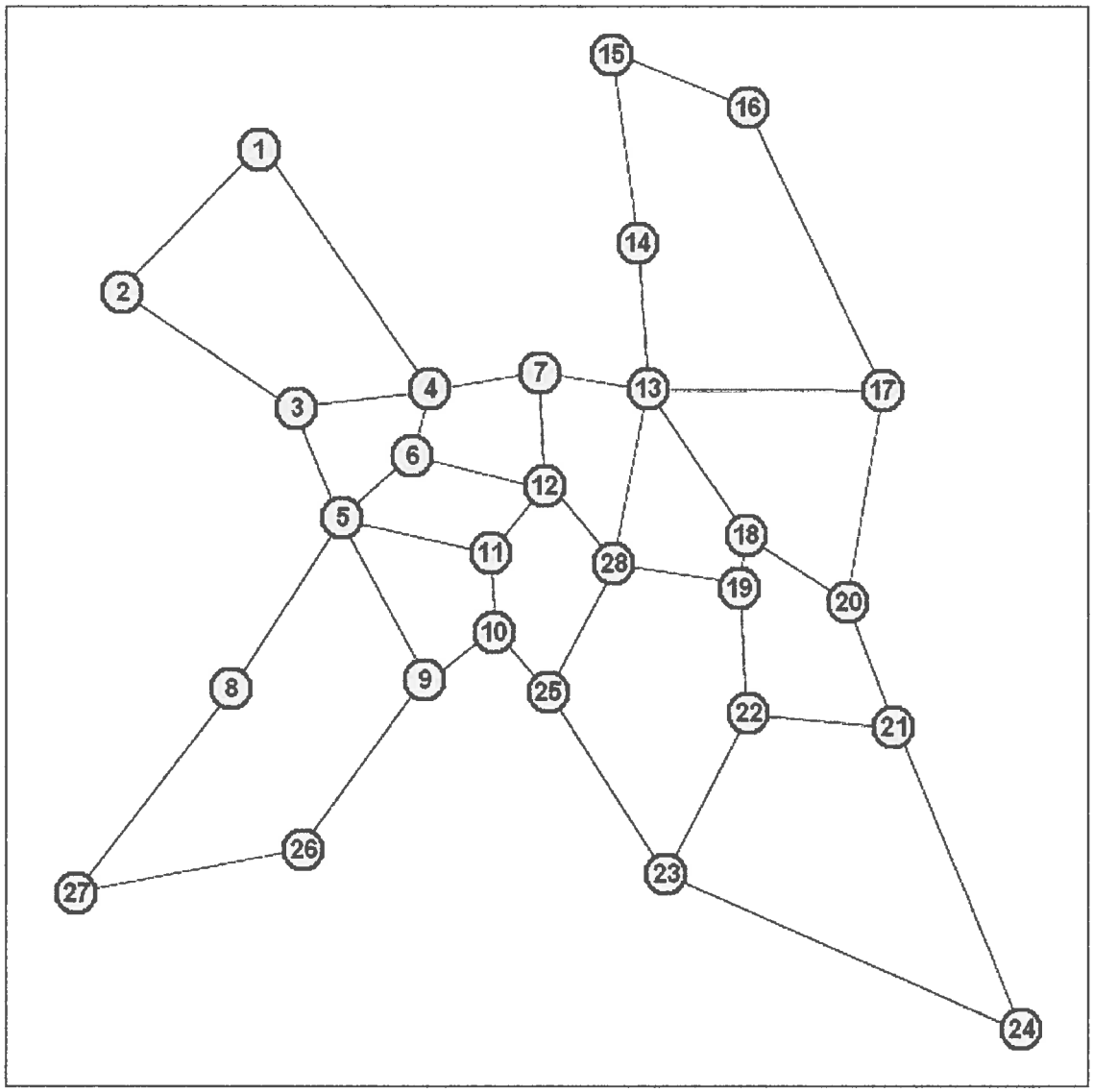

Figure 40 - Le réseau EON2004

Pour cette instance, tout comme les auteurs de [2], nous avons généré le trafic en nous inspirant de la population de chacune des grandes villes représentées par les nœuds du réseau. Ainsi, pour chaque paire de nœuds possible, la somme de la bande passante (que nous appellerons *s*) voyageant dans les deux directions est directement proportionnelle à la somme des populations des deux villes concernées. Ensuite, dans le but d'obtenir une

matrice de trafic asymétrique, nous avons tiré au hasard un nombre x entre 0,20 et 0,80 pour chaque paire de nœuds puis nous avons appliqué cette formule :

bande passante voyageant de A vers B = $x \times s$,
bande passante voyageant de B vers A = $(1 - x) \times s$.

Une fois la matrice générée, le trafic moyen entre deux nœuds est l'équivalent d'un OC-103,4 ; le débit le plus faible que trouvons dans la matrice est un débit d'OC-2, le plus important est un débit d'OC-1082 ; et l'écart moyen des données de cette matrice est l'équivalent d'un OC-78,4. La borne inférieure calculée avec l'algorithme décrit à la section 3.2.3 a une valeur de 4402. Nous calculons aussi que le nombre minimal de flots que peut avoir une solution valide pour cette instance est de 907.

6.2.1 Étude de sensibilité des paramètres

Comme cela a été fait pour l'instance NSF, nous allons procéder à une étude de sensibilité de différents paramètres de l'heuristique GRWABOU. Toutefois, nous exécuterons beaucoup moins d'expériences que ce que nous avons fait pour NSF et ce pour deux raisons. Tout d'abord, avec cette instance, nous nous retrouvons avec le même problème que lorsque la compensation est considérée pour l'instance NSF : le temps de calcul requis pour effectuer une itération est très important. Le grand nombre de nœuds du réseau avec la matrice de trafic considérée en sont à l'origine. De plus, les expérimentations effectuées sur l'instance NSF ont démontré que l'heuristique était peu sensible aux paramètres dans la mesure où ceux-ci sont fixés à des valeurs raisonnables.

L'étude présentée dans cette section a été effectuée dans le contexte où nous ne tenons pas compte ni de la compensation, ni de la possibilité de changer de longueur

d'onde. De plus, l'arrangement des cartes est toujours effectué de façon optimale (voir section 3.2.2). Dans tous les cas, les exécutions sont arrêtées après 1000 itérations. Voici les valeurs par défaut des paramètres pour cette instance.

```
number_of_element_in_tabu_list = 125
number_of_extra_length_for_considering_path = 3.2
maximum_number_of_deterioration = 1
maximum_number_of_stagnation = 4
validity_ratio_minimum = 0.5
```

Le paramètre validity_ratio_minimum

Le premier paramètre que nous examinons est le paramètre `validity_ratio_minimum`.

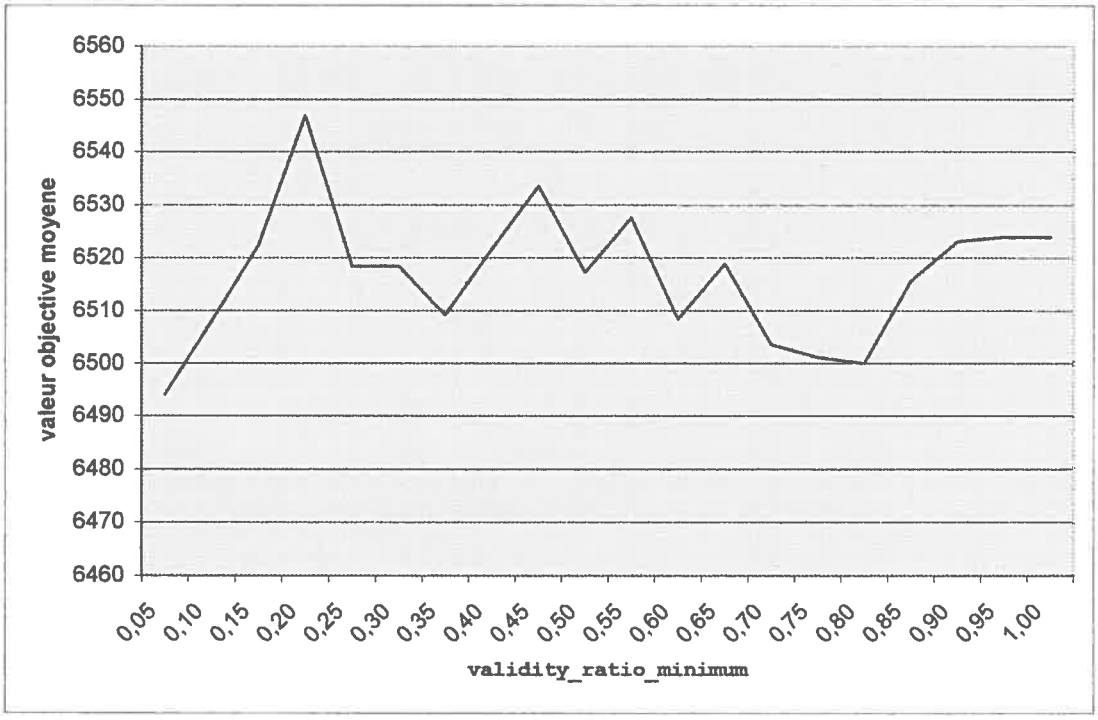

Figure 41 - L'impact du paramètre `validity_ratio_minimum`

Ce paramètre permet de gérer la pénalité à affecter à la non réalisabilité. Sur le graphique illustré à la Figure 41, les points sont la moyenne des valeurs objectives des meilleures solutions de 5 exécutions où le nombre de longueurs d'onde varie entre les valeurs {38, 40, 42, 44, 46}. Nous remarquons qu'il n'y a pas de grande tendance qui se dégage de cette courbe. Le paramètre `validity_ratio_minimum` semble avoir très peu d'influence sur les solutions obtenues par l'heuristique. La différence entre les valeurs extrêmes du graphique est de l'ordre du coût de 5 cartes de transport OC-192. Cette différence est petite compte tenu du fait que les coûts observés sont de l'ordre de 6500 et que l'effort de calcul fourni est assez faible par rapport à la taille de cette instance (comparativement à l'étude réalisée sur l'instance NSF).

Les paramètres qui gèrent l'effort passé sur un nœud

Tentons maintenant de déterminer les meilleures valeurs à affecter aux paramètres `maximum_number_of_deterioration` et `maximum_number_of_stagnation`. Sur le graphique de la Figure 42 ci-après, la première courbe (en bleu foncé) présente les points correspondants aux exécutions où la valeur du paramètre `maximum_number_of_deterioration` est fixée à 1 et la valeur du paramètre `maximum_number_of_stagnation` varie entre 1 et 9. Le même principe s'applique pour les trois autres courbes. Tous les points correspondent à une moyenne sur 5 exécutions où le nombre de longueurs d'onde varie entre les valeurs {38, 40, 42, 44, 46} comme pour l'expérience précédente (celle présentée à la Figure 41).

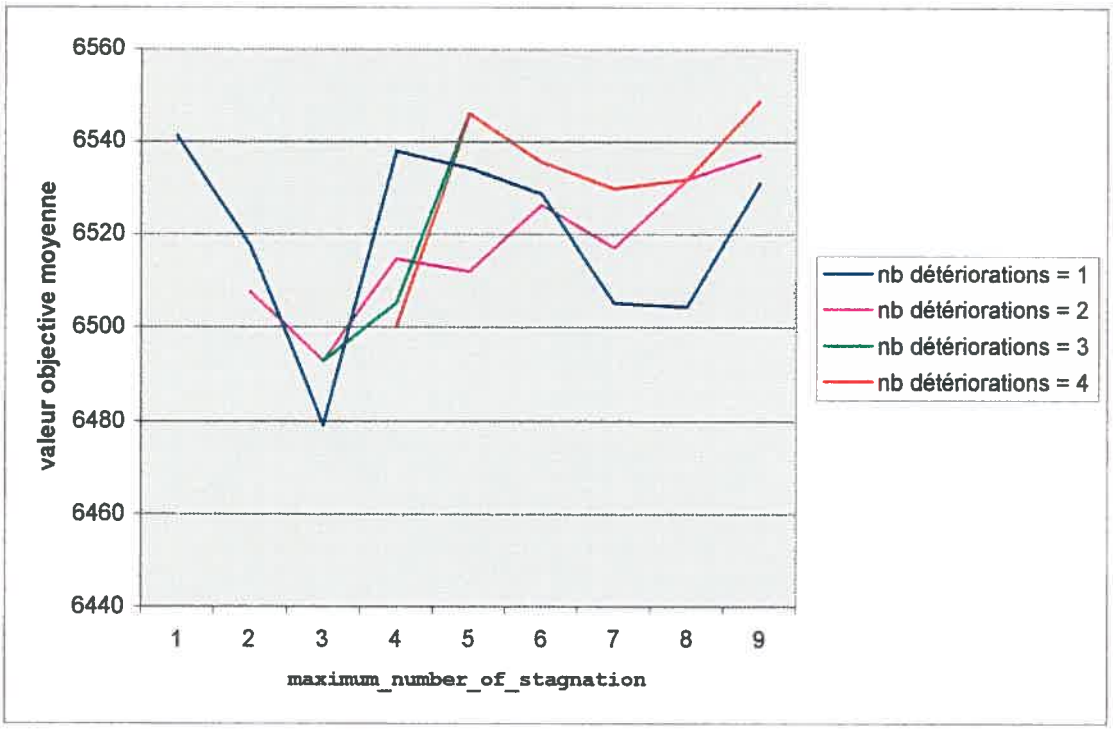

Figure 42 - L'impact des paramètres qui gèrent l'effort passé sur un nœud

Du graphique, nous pouvons déduire qu'il y a une petite différence de 60 unités de coût entre les valeurs extrêmes. Nous voyons aussi que cet ensemble de courbes comporte plusieurs similarités avec son homologue pour l'instance NSF (voir Figure 25) : il semble y avoir une tendance à préférer une valeur de 3 ou de 4 pour le paramètre `maximum_number_of_stagnation`.

L'étude de sensibilité des paramètres que nous venons de présenter est incomplète. Pour avoir des résultats clairs quant à savoir les meilleures valeurs à affecter aux différents paramètres, il faudrait commencer par changer le critère d'arrêt pour un plus grand nombre d'itérations. Il faudrait aussi calculer la moyenne sur un nombre beaucoup plus grand d'exécutions et il faudrait s'intéresser au comportement de l'heuristique lorsque nous fixons des valeurs extrêmes aux différents paramètres. Toutefois, en regardant cette étude

ainsi que les deux précédentes présentées aux sections 6.1.1 et 6.1.4, nous pouvons conclure que tant que les valeurs affectées aux paramètres restent dans des limites que nous qualifierons de raisonnables, l'heuristique GRWABOU devrait être assez stable pour pouvoir fournir des solutions de bonne qualité.

6.2.2 Analyse du dimensionnement GRWA de l'instance EON2004

L'expérience présentée ici a pour objectif de montrer l'évolution de la fonction objective en fonction du nombre de longueurs d'onde et du nombre maximal d'interruptions autorisées. Cette expérience est similaire à celle effectuée sur le réseau NSF à la section 6.1.2 (voir la Figure 26). Les paramètres communs à toutes les exécutions sont définis ci-dessous.

```
number_of_element_in_tabu_list = 125
number_of_extra_length_for_considering_path = 3.2
maximum_number_of_deterioration = 1
maximum_number_of_stagnation = 3
validity_ratio_minimum = 0.5
```

Notons aussi que le critère d'arrêt a été fixé à 5000 itérations.

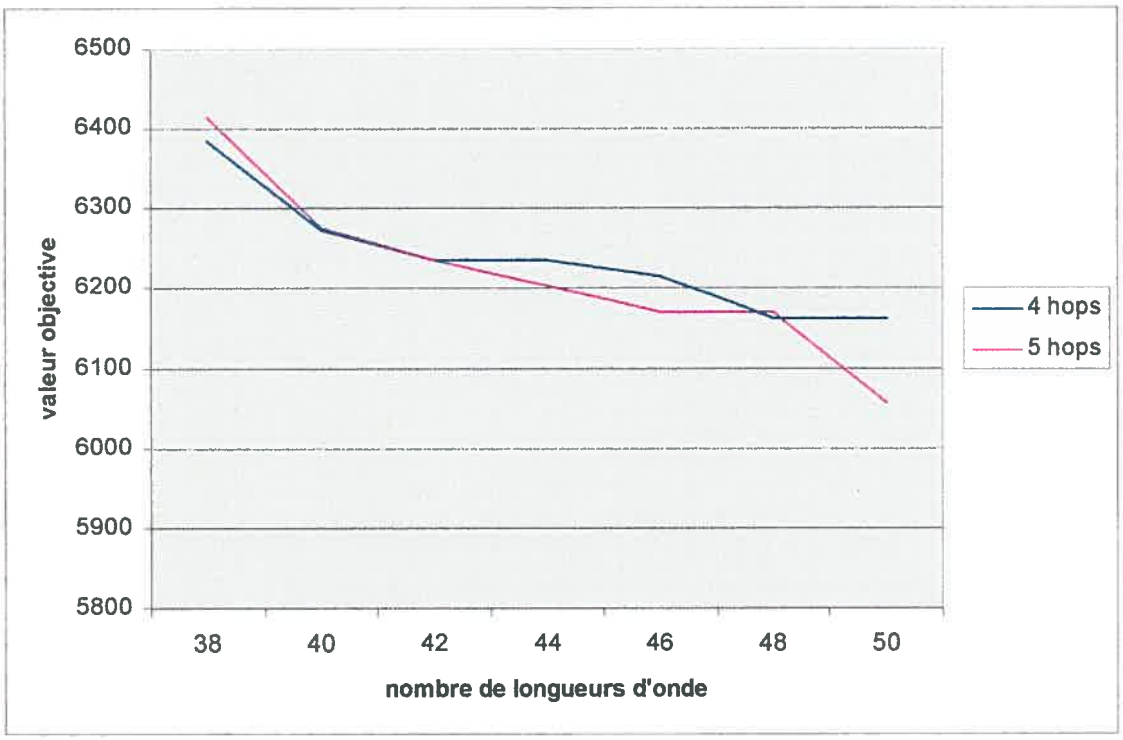

Figure 43 - Évolution de la fonction objective selon le nombre de longueurs d'onde

Pour obtenir les résultats présentés à la Figure 43, nous avons lancé 2 séries d'exécutions, une pour un maximum de 4 sauts optiques, et une autre pour un maximum de 5 sauts optiques. Pour chacune des deux séries, la solution initiale pour 38 longueurs d'onde a été construite selon l'algorithme de la section 5.2. Puis pour 40 longueurs d'onde, la solution initiale utilisée a été la meilleure solution issue de l'exécution précédente (celle avec 38 longueurs d'onde). Nous avons ensuite procédé de la même façon jusqu'à 50 longueurs d'onde.

Comme c'est le cas pour l'instance NSF, les résultats confirment le fait que lorsque le nombre de longueurs d'onde augmente, les meilleures solutions retenues sont généralement de meilleure qualité. Pour ce qui est du nombre maximal de sauts optiques, son impact sur la qualité des meilleures solutions retenues n'est pas évident. Comme pour

l'instance NSF, nous dirons que le domaine des solutions considérées réalisables pour un maximum de 5 sauts optiques et non réalisables pour un maximum de 4 sauts optiques ne contient pas beaucoup de solutions qui se démarquent beaucoup du point de vue du coût. Toutefois, il faut être très prudent pour les conclusions car il faut se rappeler que le nombre maximal d'itérations effectuées ici est assez faible.

La meilleure solution que nous trouvons ici possède un coût de 6058 (5 sauts optiques avec 50 longueurs d'onde). Or, ce coût représente 37,6 % de plus que la borne inférieure mais rappelons que la borne n'est pas de bonne qualité (voir section 3.2.3). Encore une fois, nous compléterons l'analyse de cette solution en regardant les deux métriques que nous avons considérées pour la meilleure solution de l'instance NSF (voir section 6.1.2). Ainsi, dans un premier temps, nous trouvons que le taux de remplissage moyen des ports pour cette solution est de 81,4 %. De plus, en moyenne, les chemins utilisés par les flots de cette solution possèdent 0,41 lien de plus que le plus court chemin, mais rappelons que cette dernière métrique sert comme indicateur de la solution optimale et non pas comme indicateur de la qualité de la solution. Comparativement à l'instance NSF, nous trouvons que la valeur par rapport à la borne inférieure ainsi que le taux de remplissage nous indiquent que la solution étudiée ici n'est pas d'aussi bonne qualité que la meilleure solution trouvée pour l'instance NSF. Ce résultat plus modeste s'explique tout d'abord par le fait qu'il aurait fallu mettre un effort de calcul bien plus considérable pour résoudre cette instance. Idéalement, plus l'instance est de grande taille, plus l'heuristique doit y passer du temps. Aussi, le fait que le trafic soit asymétrique ajoute un degré de difficulté, en ce sens qu'il y aura toujours un grand nombre de ports inutilisés dans une solution.

6.2.3 Analyse du dimensionnement GRWA de l'instance NSF en tenant compte de la possibilité de changer de longueur d'onde

Pour cette expérience, nous allons reprendre la courbe de la Figure 43 où le nombre de sauts optiques est fixé à 4. Ensuite, à chacune des meilleures solutions qui correspondent aux différents points de la courbe, nous avons appliqué l'algorithme qui rend possible le changement de longueur d'onde. Le graphique de la Figure 44 présente les résultats obtenus.

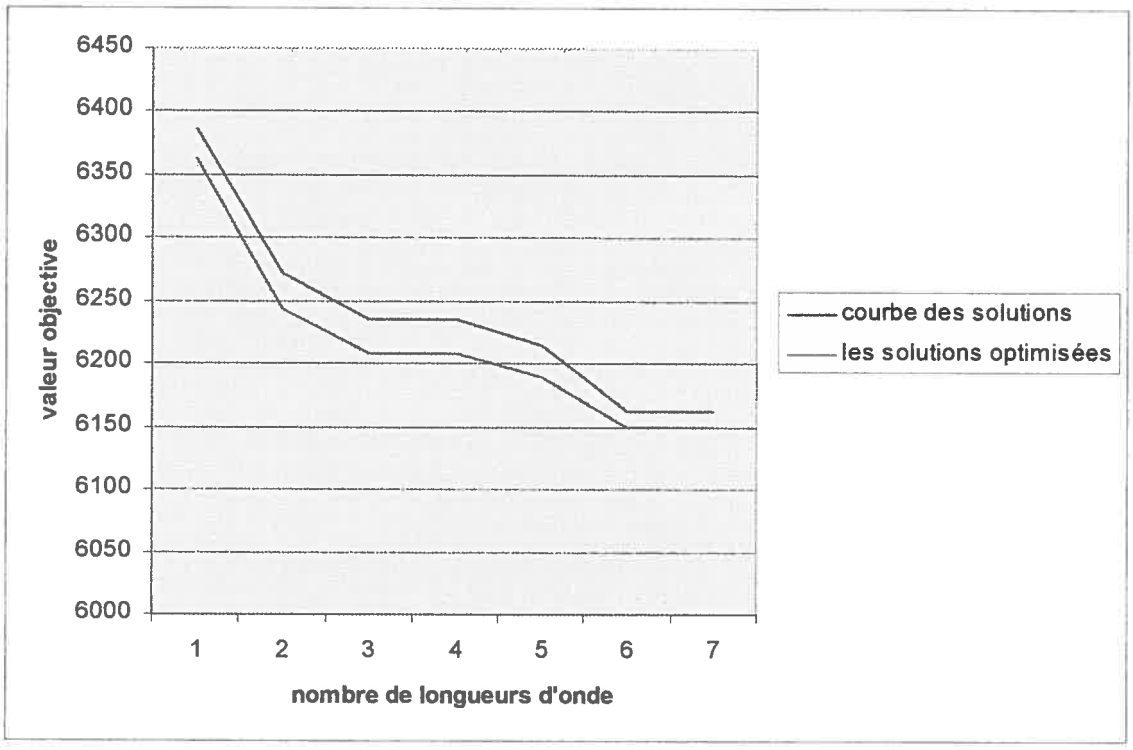

Figure 44 - L'impact de la post-optimisation pour chaque solution

La courbe des solutions en bleu foncé correspond à la même courbe que celle du graphique de la Figure 43. Les points de la courbe des solutions optimisées en rouge sont les valeurs objectives des solutions correspondantes une fois qu'elles ont été optimisées par l'algorithme. Les gains ne sont pas exceptionnels mais nous voyons que pour les solutions testées, l'optimisation sur la non continuité de longueurs d'onde arrive à trouver des solutions à meilleurs coûts.

6.2.4 Analyse du dimensionnement GRWA de l'instance NSF en considérant la compensation

Lorsque nous considérons la compensation sur une instance de grande taille telle l'instance EON2004, le temps de calcul devient rapidement un problème. Ainsi, pour alléger un peu le trafic, nous avons divisé par 10 toutes les entrées de la matrice de trafic pour cette expérience. De cette façon, le débit moyen entre deux nœuds devient l'équivalent d'un OC-10,3. En effectuant ce changement, nous remarquons que les solutions obtenues comportent un bien plus grand nombre de cartes OC-48 comparativement au nombre de cartes OC-192. Rappelons qu'un signal optique modulé en OC-48 peut voyager sur une fibre optique sur plus de 500 km sans avoir à être compensé. Ainsi, puisque cette longueur est de l'ordre de la longueur moyenne des liens du réseau, nous avons choisi de conserver les distances réelles des liens qui relient les différentes villes européennes, soit une moyenne d'environ 625 km par lien. Cette moyenne est certes supérieure à 500 km mais il reste que 30 % des liens sont en dessous de cette limite. À présent, regardons plus en détail les résultats obtenus pour cette expérimentation.

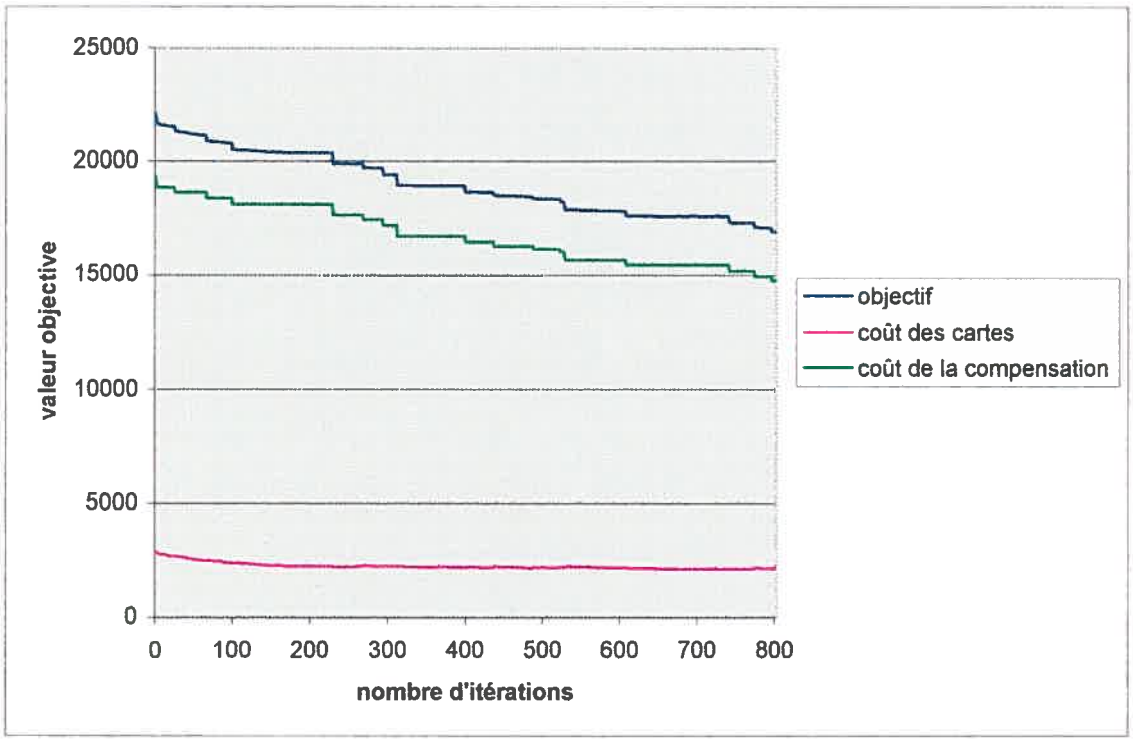

Figure 45 - Exécution avec 10 longueurs d'onde et 4 sauts optiques

Le graphique illustré à la Figure 45 présente l'évolution de la solution au fil des itérations de l'exécution où nous avons fixé le nombre de longueurs d'onde à 10 et le nombre maximal de sauts optique à 4. Contrairement à l'instance NSF, nous remarquons que le coût des solutions est ici dominé par les compensateurs et non par les cartes. Ceci est principalement dû au fait que la longueur moyenne des liens fait en sorte que pour la majorité des liens, l'utilisation de compensateurs est presque inévitable. Nous voyons aussi que le coût moyen des solutions ne s'est pas stabilisé au cours des 800 itérations. Ceci signifie qu'il faudrait laisser beaucoup plus de temps à l'algorithme pour obtenir de meilleures solutions.

Chapitre 7 – Conclusion

Nous avons développé l'heuristique GRWABOU avec un modèle très général et réaliste des contraintes du problème GRWA. Le plan d'expérimentation que nous avons proposé nous a permis de tester les multiples fonctionnalités développées tout au long de ce travail. Nous avons vu que l'heuristique s'est révélée être très performante en termes de qualité des solutions obtenues. De plus, les résultats ont permis de quantifier les effets des différentes formes d'optimisation proposées tout au long de ce mémoire sur la qualité des meilleures solutions trouvées par l'heuristique.

Selon le contexte dans lequel ce travail a été effectué ainsi que les hypothèses considérées, nous voyons que le présent mémoire constitue un pas en avant par rapport à ce qui a été exploré dans la littérature. La présente étude se démarque particulièrement par le fait que nous avons considéré explicitement la compensation. Ceci nous a permis de mieux comprendre et de préciser quel compromis il faut faire entre l'utilisation des cartes de transport versus la compensation dépendant des capacités de transport utilisées. Cette étude combinée avec la considération de deux capacités de transport possibles constitue une voie de recherche encore inexplorée dans ce domaine. Ce raffinement de la modélisation va dans la direction où les modèles proposés incluent un plus grand nombre de paramètres et de caractéristiques des réseaux.

Toutefois, pour améliorer ce travail, il faudrait porter une attention plus particulière au temps de calcul pris par l'algorithme. En fait, nous avons proposé, pour la compensation comme pour la possibilité de changer de longueur d'onde lors de la régénération d'un signal, des algorithmes d'optimisation des solutions pour pouvoir traiter

ces aspects du problème. Les algorithmes sont efficaces certes, mais l'idéal serait de modifier la structure des données pour mieux intégrer ces problématiques de façon à ce qu'aucune forme supplémentaire ou différé d'optimisation ne soit nécessaire (pour ajouter la compensation par exemple) après avoir effectué un mouvement quelconque sur une solution.

Pour les travaux futurs, l'objectif principal est d'aboutir à une modélisation de plus en plus réaliste. Pour atteindre cet objectif, il convient de prendre en compte explicitement plusieurs paramètres de la couche physique. Le fait de considérer la compensation constitue en fait un premier pas dans cette direction. À titre d'exemple, il faudrait tenir compte du phénomène de la dispersion chromatique, il convient de mentionner que cet effet est variable selon la longueur d'onde considérée. Il y aussi le phénomène de la perte de puissance du signal optique en fonction de la distance parcourue dans une fibre optique. Notons aussi que notre étude avait pour objectif de minimiser le coût du dimensionnement d'un réseau optique, or il faudrait idéalement pouvoir considérer simultanément plusieurs objectifs tels que maximiser le nombre de connexions acceptées ou encore minimiser le délai de propagation moyen. Bref, l'univers des possibilités à explorer est encore très vaste, et ceci sans compter le fait que la technologie évolue assez rapidement pour offrir continuellement de nouvelles opportunités de recherche.

Bibliographie

[1] R.A. Berry et E. Modiano. *The role of switching in reducing the number of electronic ports in WDM networks.* IEEE Journal on Selected Areas in Communications, vol. 22, no. 8, pp. 1396–1405, Octobre 2004.

[2] A. Betker, C. Gerlach, R. Hulsermann, M. Jager, M. Barry, S. Bodamer, J. Spath, C. Gauger et M. Kohn, *Reference transport network scenarios.* MultiTeraNet Project, Tech. Rep., 2004.

[3] L. Chiewon et E.K. Park. *A genetic algorithm for traffic grooming in all-optical mesh networks.* IEEE International Conference on Systems, Man and Cybernetics, vol. 7, 6 pages, 6-9 Octobre 2002.

[4] X. Chunsheng, Q. Chunming et D. Sudhir. *Traffic grooming in mesh WDM optical networks - performance analysis.* IEEE Journal on Selected Areas in Communications, vol. 22, no. 9, pp. 1658–1669, Novembre 2004.

[5] F. Glover. *Tabu Search: A Tutorial.* INTERFACES, pp. 74–94, 1990.

[6] F. Glover. *Tabu Search--Part I.* ORSA J. on Computing, pp. 190–206, 1989.

[7] A.H. Gnauck et R.M. Jopson. *Optical Fiber Telecommunications – IIIA.* Dispersion Compensation for Optical Fiber Systems, chapitre 7 (pp.162-195), édité par I.P. Kaminow et T.L. Koch, Academic Press, 1997.

[8] H. Helvoort. *Next Generation SDH/SONET: Evolution or Revolution ?* 254 pages, Chichester : Wiley, Avril 2005.

[9] H. Hong et J.A. Copeland. *Hybrid wavelength and sub-wavelength routed optical networks.* GLOBECOM, vol. 4, pp. 2119–2123, 25-29 Novembre 2001.

[10] A. Houle, B. Jaumard et Y. Solari. *Dimensioning WDM Optical Network with Minimum MSPP Configuration.* IASTED International Conference OCSN 2004 (Optical Communications Systems and Networks) Proceedings, Juillet 2004.

[11] J.Q. Hu et B. Leida. *Traffic grooming, routing, and wavelength assignment in optical WDM mesh networks.* 23th Annual Conference, INFOCOM 2004, pp 495–501, 7-11 Mars 2004.

[12] B. Jaumard, F. Vanderbeck et B. Vignac. *Decomposition Formulations for the GRWA Problem.* En préparation, dans le cadre de la thèse de doctorat de Benoît Vignac.

[13] Z. Keyao et B. Mukherjee. *Traffic grooming in an optical WDM mesh network.* IEEE Journal on Selected Areas in Communications, vol. 20, no. 1, pp. 122–133, Janvier 2002.

[14] R. Krishnaswamy et K. Sivarajan. *Design of logical topologies: A linear formulation for wavelength routed optical networks with no wavelength changers.* IEEE/ACM Transactions on Networking, vol. 9, no. 2, pp. 184–198, Avril 2001.

[15] P. Prathombutr, J. Stach et E.K. Park. *An algorithm for traffic grooming in WDM optical mesh networks with multiple objectives.* 12th International Conference, ICCCN 2003, pp. 405–411, 20-22 Octobre 2003.

[16] R. Ramaswami et K. N. Sivarajan. *Optical Networks: A Practical Perspective, Second Edition.* 864 pages, San Mateo CA : Morgan Kaufmann, 2001.

[17] Z. Shu et B. Ramamurthy. *Dynamic traffic grooming algorithms for reconfigurable SONET over WDM networks.* IEEE Journal on Selected Areas in Communications, vol. 21, no. 7, pp. 1165–1172, Septembre 2003.