UNIVERSITE SAAD DAHLAB DE BLIDA

Faculté des Sciences

Département de mathématiques

MEMOIRE DE MAGISTER

Spécialité : Recherche Opérationnelle

CONTRIBUTION A L'ETUDE ALGORITHMIQUE DES PROBLEMES DE CHEMINEMENT MULTICRITERE

Par

HOUAS Mohamed

Devant le jury composé de :

F. HANNANE	Professeur, U. de Blida	Président
M. ABBAS	Professeur, USTHB, Alger	Promoteur
M. BLIDIA	Professeur, U. de Blida	Examinateur
A. DERBALA	Maître de conférence, U. de Blida	Examinateur
M. CHELLALI	Maître de conférence, U. de Blida	Examinateur

Blida, mai 2008

RESUME

Martins [11] a proposé un algorithme pour déterminer l'ensemble maximal complet des chemins efficaces lorsque toutes les évaluations sont positives.

L'algorithme de Martins calcule l'ensemble maximal complet des chemins efficaces d'un sommet source à tout autre sommet du réseau et utilise plusieurs critères S-type. Ces caractéristiques ont permit d'étendre cet algorithme pour optimiser simultanément les critères S-type et M-type. Gandibleux et al., [34] ont proposé une version révisée de l'algorithme de Martins pour les problèmes du type (σ-S | 1-M). Le changement principal concerne le test de dominance pour s'assurer que l'ensemble des chemins efficaces pour les problèmes de type (σ-S | 1-M) est maximal complet.

Dans ce travail, on présente une version révisée de l'algorithme de Martins pour les problèmes de plus court chemin multicritère de type (σ-S | 2-M). Cette version calcule tous les chemins efficaces d'un sommet donné source, à tous les autres sommets du réseau. Cette version permet de prendre en compte plusieurs fonctions critères de type S-type et deux fonctions critères de type M-type. Elle peut résoudre aussi les problèmes dans lesquels les évaluations des arcs du réseau sont toutes positives, et optimiser plusieurs critères simultanément. La version proposée manipule deux fonctions critères de type max-min et plusieurs fonctions critères linéaires. La révision de l'algorithme de Martins touche le test de dominance et la procédure d'identification des chemins efficaces.

ABSTRACT

Martins [11] proposed an algorithm to determine the maximal complete set of efficient paths when all the valuations are positives.

The Martins algorithm calculates the maximal complete set of efficient paths and able to use several S-type criterion. These characteristics have makes it possible to extend this algorithm to optimize simultaneously the S-type and M-type criterion. Gandibleux and al, [34] proposed a revised version of the Martins algorithm for (σ-S|1-M) problems. The principal change relates to the of dominance test to make sure that the set of efficient paths for (σ-S|1-M) problems is maximal complete.

In this work, we introduce a revise version of Martins algorithm for the multicriteria shortest path for (σ-S|2-M) problems. This version calculates all the efficient paths from a given source vertex, to all other vertices of network. This version allow to take into account several criteria functions of type S-type and tow criteria functions of type M-type. She can resolve the problems in which the valuations of network arcs are all positives, and it optimizes several criteria functions simultaneously. The suggested version manipulates two criteria functions of type max-min and several linear criteria functions. The revision of the Martins algorithm touch the domination test and the procedure for identification efficient paths.

ملخص

مارتن [11] اقترح منهاجا لتحديد المجموعة القصوى الكاملة من المسارات الفعالة، عندما تكون كافة تغيرات منحنى القوس في الشبكة موجبة.

منهاج مارتن يحسب المجموعة القصوى لقيمة مصدر معطى بالنسبة لكل القيم الأخرى للشبكة، ويستخدم عدة دوال المقياس من صنف S. وقد مكنت هذه السمات لتوسيع نطاق هذا المنهاج بالأخذ بعين الاعتبار عدة دوال من صنف S والصنف M .اقترح قونديبلو وآخرون [34] إصدار معدل لمنهاج مارتن لإشكالية المسار الأقصر المتعدد المقياس من نوع $(\sigma\text{-S} \mid \text{1-M})$. تعديل منهاج مارتن يمس اختيار الإحاطة و إجراء تعيين المسارات الفعالة لضمان أن المجموعة القصوى المسارات الفعالة لمشاكل من نوع $(\sigma\text{-S} \mid \text{1-M})$ كاملة.

في هذا العمل نعرض إصدارا معدلا لمنهاج مارتن لإشكالية المسار الأقصر المتعدد المقياس من نوع $(\sigma\text{-S} \mid \text{2-M})$، في هده الأخيرة تحسب كل المسارات الفعالة لقيمة مصدر معطى بالنسبة لكل القيم الأخرى للشبكة، هذا الإصدار يسمح بالأخذ بعين الاعتبار عدة دوال المقياس من صنف S، و دالتين من صنف M. بإمكانها أيضا حل الإشكاليات حيث تكون تغيرات منحنى القوس في الشبكة كلها موجبة، و تحدد أكبر قيمة لعدة مقاييس في آن واحد. الإصدار المقترح يستعمل دالتين مقياس من صنف أقص-أدنى و عدة دوال مقياسات خطية. تعديل منهاج مارتن يمس اختيار الإحاطة و إجراء تعيين المسارات الفعالة.

REMERCIEMENTS

Je remercie **ALLAH** de m'avoir donné la force et le courage de réaliser ce modeste travail.

Je tiens en premier lieu à exprimer toute ma reconnaissance à Monsieur **Moncef Abbas**, Professeur à l' USTHB, pour la confiance qu'il m'a accordée en me permettant de réaliser ce mémoire sous sa direction, pour ses précieux conseils, sa disponibilité et son aide.

Je remercie également Monsieur **Hannane Farouk**, Professeur à l'université Saad Dahlab de Blida, pour l'honneur d'avoir présider ce jury.

Je remercie aussi les membres du jury: Monsieur **Blidia Mostafa**, Professeur à l'université Saad Dahlab de Blida, Monsieur **Derbala Ali**, Maître de conférences à l'université Saad Dahlab de Blida, Monsieur **Chellali Mustapha**, Maître de conférences à l'université Saad Dahlab de Blida.

Je remercie tous les professeurs de l'institut de Mathématiques de l'université Saad Dahlab de Blida, particulièrement le chef de département Monsieur Tami Omar, pour ses encouragements.

Je remercier aussi Noureddine, Amina et Ouafia pour leur soutien moral, sans oublier toutes les personnes qui m'ont aidé par leurs encouragements et qui m'ont permis de mener mon travail à son terme.

Un grand merci à ma famille qui a été là pour m'encourager et m'inciter à faire de mon mieux.

TABLE DES MATIERES

LISTE DES ILLUSTRATIONS, GRAPHIQUES ET TABLEAUX

INTRODUCTION

De nombreux secteurs de l'industrie (télécommunication, environnement, transport,... etc.) sont concernés par des problèmes complexes de grandes dimension et multicritère (coût de parcours, temps de parcours, qualité de service,... etc.) pour lesquels les décisions doivent être prises de façon optimale. Les problèmes d'optimisation rencontrés en pratique sont rarement monocritère. Il y a généralement plusieurs critères contradictoires à satisfaire simultanément. L'optimisation multicritère s'intéresse à la résolution de ce type de problèmes.

L'optimisation multicritère cherche donc à optimiser plusieurs composantes d'un vecteur fonction coût. Contrairement à l'optimisation monocritère, la solution d'un problème multicritère n'est pas une solution optimale, mais un ensemble de solutions, connu comme l'ensemble de solutions de meilleurs compromis (Pareto optimales). Toute solution de cet ensemble est optimale dans le sens qu'aucune amélioration ne peut être faite sur une composante du vecteur sans dégradation d'au moins une autre de ses composantes.

Un grand nombre d'approches existent pour résoudre les problèmes multicritère. Certaines utilisent des connaissances du problème pour fixer des préférences sur les critères et ainsi contourner l'aspect multicritère du problème. D'autres mettent tous les critères au même niveau d'importance, mais là aussi il existe plusieurs façons de réaliser une telle opération.

Parmi toutes ces approches, il faut distinguer deux catégories: les approches non Pareto et les approches Pareto. Les approches non Pareto ne traitent pas le problème comme un véritable problème multicritère. Elles cherchent à ramener le problème initial à un ou plusieurs problèmes monocritères. Par opposition aux méthodes non Pareto, les approches Pareto ne transforment pas les critères du problème, ceux-ci sont traités sans aucune distinction pendant la résolution.

Dans ce travail, on s'intéresse au problème du cheminement classique de l'optimisation dans les réseaux mais dans le cas multicritère, en définissant un réseau multicritère $R = (N, A, \upsilon)$ comme étant la donnée d'un graphe $G = (N, A)$ et d'une application poids:

$\upsilon : A \rightarrow \mathrm{R}^r$

$a \rightarrow \upsilon(a) = (\upsilon^1(a), \upsilon^2(a), ..., \upsilon^r(a))$, où r étant le nombre de critères.

Le problème de cheminement multicritère consiste en la recherche d'un *plus court chemin* dans un réseau valué par un vecteur à r critères.

La variante monocritère du problème du plus court chemin est considérée comme étant pratiquement résolue. Mais, le cas *multicritère* est une variante NP-complète [1] qui fait encore l'objet de travaux de recherche. Généralement, les algorithmes de résolution proposés pour cette variante se basent, essentiellement, sur l'utilisation des algorithmes du problème classique monocritère ou sur des versions légèrement modifiées. Il existe deux familles d'algorithmes pour la résolution du problème du plus court chemin multicritère: les algorithmes de marquage permanent (fixation d'étiquettes) dans lesquels une étiquette est maintenue de façon permanente dans chaque itération, et les algorithmes de marquage temporaire (correction d'étiquettes) dans lesquels les étiquettes ne deviennent permanentes que dans la dernière itération. Dans ce travail on s'intéresse à la famille des algorithmes de marquage permanent.

Deux types de fonctions critères sont considérés dans les problèmes du plus court chemin multicritère: *fonctions linéaires* (de type *somme*) et *fonction du type max-min*, notés respectivement par (S-type) et (M-type). Dans le cas général, la notation (σ-S $|$ μ-M) signifie que le problème concerne respectivement σ critères du premier type et μ critères du second type.

Il existe deux classes d'algorithmes pour la résolution des problèmes du plus court chemin multicritère du type S-type: les algorithmes de marquage (les algorithmes d'étiquetage) et les algorithmes de rangement des chemins. Lorsque toutes les fonctions critères sont du type S-type, l'ensemble des chemins efficaces, peut

être déterminé en utilisant l'algorithme de Martins. Une version révisée de l'algorithme de Martins, récemment proposée par Gandibleux et al. , pour résoudre les problèmes de type (σ-S｜1-M). Nous proposons Une version révisée de l'algorithme de Martins, pour résoudre les problèmes de type (σ-S｜2-M). Cette version permet de prendre en compte plusieurs fonctions critères du type S-type et deux fonctions critères du type M-type.

Ce travail est organisé en cinq chapitres.

Dans le chapitre 1, on introduit le domaine de l'optimisation multicritère. Nous présentons les principes et les concepts mathématiques relatifs à l'optimisation multicritère, ensuite nous abordons les différentes approches de leur résolution. Dans la section 2, on décrit les définitions de base de la théorie des graphes, de l'optimisation dans les réseaux et on rappelle quelques résultats connus qui seront utilisés tout au long de ce travail.

Le chapitre 2 est consacré à l'étude du problème de recherche d'un plus court chemin joignant un sommet source s à un sommet puits t dans un réseau multicritère et du problème de recherche d'un plus court chemin joignant un sommet source s à tous autre sommet du réseau multicritère. Après l'introduction du problème de cheminement monocritère et la définition des deux problèmes, on donne ensuite les conditions nécessaires et suffisantes d'existence des solutions efficaces.

Au chapitre 3, nous donnons les caractéristiques du problème du plus court chemin. Ensuite, nous étudions la complexité du problème du plus court chemin et on termine par la présentation des algorithmes de recherche du plus court chemin monocritère.

Dans le chapitre 4, nous donnons un aperçu sur les algorithmes de résolution des deux problèmes traités dans le chapitre 2.

Le chapitre 5 est consacré à l'étude de la révision de l'algorithme de Martins pour les problèmes du plus court chemin multicritère avec fonction coût max-min. En effet, nous présentons d'abord une nouvelle version révisée de l'algorithme de Martins pour les problèmes de type (σ-S | 1-M) ensuite, nous proposons une version révisée de l'algorithme de Martins pour les problèmes de type (σ-S | 2-M).

CHAPITRE 1
OPTIMISATIUON MULTICRITERE ET CONCEPTS DE LA THEORIE DES GRAPHES

1.1. Introduction

Les problèmes de cheminement constituent un thème classique en recherche opérationnelle dont les applications sont très nombreuses, notamment en transport et en télécommunications. Nous considérons, dans ce travail, l'un des plus importants problèmes de cheminement: le problème de *plus court chemin*. Soit un réseau $R = (N, A, \upsilon)$. À chaque arc $a = (i, j)$ de R est associé un coût υ_{ij}. Le problème de plus court chemin consiste à trouver un chemin entre un sommet source s donné et sommet puits t donné qui minimise la somme des coûts des arcs. Il existe plusieurs algorithmes pour résoudre ce problème, le plus connu étant celui de Dijkstra [2]. Cependant, son application est limitée au cas classique du problème.

Dans la première section de ce chapitre, nous présentons les principes et les concepts mathématiques relatifs à l'optimisation multicritère, ensuite nous abordons les différentes approches de leur résolution. Dans le deuxième paragraphe, nous donnons les définitions de base de la théorie des graphes.

1.2. Optimisation multicritère [3]

Les problèmes d'optimisation combinatoire issus des problématiques réelles sont la plupart du temps, de nature multicritère (multiobjectif) car plusieurs critères d'évaluation souvent contradictoires sont à considérer simultanément. Optimiser un tel problème relève donc de l'optimisation combinatoire multicritère.

1.2.1. Problème d'optimisation multicritère (multiobjectif)

L'optimisation multicritère consiste à optimiser plusieurs composantes d'un vecteur de fonction de coût, chaque composante de ce vecteur correspondant à un critère (objectif). Un problème d'optimisation multicritère (multiobjectif) POM peut être défini comme suit:

$$(POM) \quad \begin{cases} Optimiser \quad \upsilon(x) = (\upsilon^1(x),...,\upsilon^r(x)) \\ sous \quad x \in D \end{cases}$$

Où r est le nombre des critères (r $\geq$ 2), $x = (x_1, x_2, ..., x_k)$ est le vecteur représentant les variables de décision, D représente l'ensemble des solutions réalisables et chacune des fonctions $\upsilon^i(x)$ est à optimiser.

La solution d'un problème multicritère n'est pas unique, mais c'est un ensemble de solutions *efficaces* (*non-dominées*), connu comme l'ensemble des solutions Pareto Optimales.

1.2.2. Concepts de base

Soient x_1 et x_2 deux variables de D, et $\upsilon(x_1), \upsilon(x_2)$ leurs vecteurs de performance (vecteur des fonctions critères ou vecteur critères).

Définition 1.1. (Dominance)

On dit que $\upsilon(x_1)$ *domine* $\upsilon(x_2)$ si et seulement si pour tout k = 1,…,r, $\upsilon^k(x_1) \leq \upsilon^k(x_2)$ avec au moins une inégalité stricte.

Si $\upsilon(x_1)$ domine $\upsilon(x_2)$, alors $\upsilon(x_1)$ est au moins aussi bon que $\upsilon(x_2)$ sur tous les critères, et meilleur que lui sur au moins un critère.

Définition 1.2. (Dominance stricte)

On dit que $\upsilon(x_1)$ *domine strictement* $\upsilon(x_2)$ si et seulement si pour tout k = 1,…,r, $\upsilon^k(x_1) < \upsilon^k(x_2)$.

Si $\upsilon(x_1)$ domine strictement $\upsilon(x_2)$, alors $\upsilon(x_1)$ est meilleur que $\upsilon(x_2)$ sur tous les critères.

Définition 1.3. (Efficacité)

Une solution $x^* \in D$ est une solution efficace si et seulement si son vecteur de performances $\upsilon(x^*)$ est non-dominé.

Une définition équivalente de l'efficacité est:

Définition 1.4.

Une solution $x^* \in D$ est une *solution efficace* si et seulement si il n'existe pas de $x \in D$ tel que $\upsilon^k(x) \leq \upsilon^k(x^*),\ k = 1,...,r$, avec au moins une inégalité stricte.

Remarque 1.1.

A partir d'un point efficace, il est impossible d'augmenter la valeur d'un des critères sans diminuer la valeur d'au moins un autre critère.

Définition 1.5. (Efficacité faible)

Une solution $x^* \in D$ est une solution *faiblement efficace* s'il n'existe pas de $x \in D$ tel que $\upsilon^k(x) < \upsilon^k(x^*),\ k = 1,...,r$.

Une solution est faiblement efficace si son vecteur de performances n'est pas strictement dominé.

Remarque 1.2.

Un vecteur de performance d'une variable de décision est défini dans l'espace des critères, et a pour coordonnées les valeurs de cette variable sur les différents critères.

L'ensemble des solutions *efficaces* (*non-dominées*) d'un problème d'optimisation multicritère constitue le *front de Pareto* FP (figure 1.1).

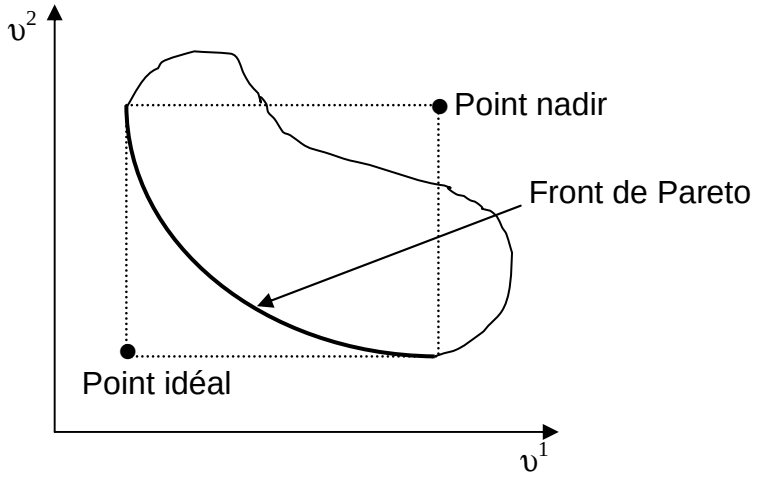

Figure 1.1. Représentation du front de Pareto (FP).

La figure 1.1 fait également apparaître deux points particuliers: le *point idéal* et le *point nadir*, définis comme ci-dessous.

Définition 1.6. (Point idéal)

Les coordonnées du point idéal (P_i) correspondent aux meilleures valeurs de chaque critère des points du front de Pareto.

$$P_i = \min\{x_i / x \in FP\}$$

Définition 1.7. (Point nadir)

Les coordonnées du point nadir (P_n) correspondent aux valeurs maximales de chaque critère des points du front de Pareto.

$$P_n = \max\{x_i / x \in FP\}$$

Remarque 1.3.

En fait, le vecteur $(\upsilon^1(x^*), \upsilon^2(x^*), ..., \upsilon^r(x^*))$ qui représente l'image de la solution efficace dans l'espace des critères tient compte de l'importance relative des

critères. Autrement-dit le rapport des poids associés aux critères i et j est une constante, i.e $\left(\frac{\alpha_i}{\alpha_j}\right) = w_{ij}$.

En tenant compte aussi des poids des critères, il est possible de définir la notion de fonction scalarisante. Il s'agit d'une fonction : $S(\upsilon, \alpha)$: $R^r \rightarrow R$

$$S(\upsilon, \alpha) = \sum_{k=1}^{r} \alpha_k \upsilon^k \quad \text{avec} \quad \sum_{k=1}^{r} \alpha_k = 1, \quad \alpha_k > 0 \; ; \forall \; k = 1, ..., r.$$

Théorème1.1. [4]

Soit le problème:

$$\min_{\substack{x \in D \\ \alpha \in \Lambda}} (\upsilon(x), \alpha) \text{ avec } \Lambda = \left\{ \alpha_k : \sum_{k=1}^{r} \alpha_k = 1 \;\; et \;\; \alpha_k > 0, k = 1, ..., r \right\}$$

Alors υ^* est un point efficace si et seulement si υ^* est une solution optimale du problème paramétrique précédent.

Ce théorème concerne la minimisation d'une combinaison convexe des critères. Il s'agit d'une condition nécessaire et suffisante pour la détermination des solutions Pareto optimales propres (les solutions décrivant la fermeture convexe de la frontière efficace).

1.2.3. Structure du front de Pareto (FP)

L'objectif de la résolution d'un problème multicritère est de fournir aux décideurs un ensemble (le plus complet possible) de solutions Pareto (efficaces), afin qu'ils puissent ensuite choisir les solutions qui les intéressent le plus.

Une question se pose donc sur la nature de ces solutions efficaces et la nécessité de les obtenir toutes. Une étude de la frontière Pareto doit donc être réalisée.

1.2.3.1. Front minimal / front maximal complet

La définition de front se réfère à l'espace des critères. Une solution appartient au front si elle n'est dominée par aucune autre solution réalisable.

Lorsque deux solutions ont exactement les mêmes valeurs pour l'ensemble des critères, elles sont équivalentes dans l'espace critère, mais peuvent correspondre à deux solutions différentes dans l'espace décisionnel. Une question importante est de savoir s'il est intéressant de garder ces deux différentes solutions.

La réponse peut dépendre du contexte du problème (type de problème étudié) en plus de la volonté des décideurs:

- Lors de la résolution d'un problème comportant énormément de solutions Pareto, il est peut être préférable de privilégier une bonne approximation de l'ensemble de la frontière et donc favoriser la diversité des solutions retenues.
- Au contraire, lorsque la frontière Pareto comporte peu de solutions, afin d'avoir une bonne représentation de l'ensemble des solutions non-dominées, il sera intéressant de rechercher les solutions de même valeur.

Nous parlerons alors de recherche du *front minimal*, dans le premier cas, et du *front maximal complet*, dans le second.

1.2.4. Concepts de résolution

La littérature consacrée à l'optimisation multicritère est très riche en terme d'approches de résolutions. Il existe de nombreuses stratégies: certaines sont basées sur la notion de dominance, d'autres procèdent par une transformation du problème [5].

Parmi ces dernières approches figurent la somme pondérée [6] et la méthode lexicographique [7]. La première est définie comme une approche naïve qui transforme un problème multicritère en problème monocritère. La fonction υ est représentée par une agrégation des fonctions $\upsilon^1, \upsilon^2, \ldots, \upsilon^r$ pondérées par des poids dont la valeur appartient à l'intervalle $[0,1]$ et la somme est égale à 1 [6] (équation 1.1)

$$\upsilon(x) = \alpha_1.\upsilon^1(x) + \alpha_2.\upsilon^2(x) + \cdots + \alpha_r.\upsilon^r(x)$$
$$tel\ que\ \ \alpha_i \in [0,1],\ \forall\ i = 1,\cdots,r\ et\ \sum_{i=1}^{r} \alpha_i = 1 \qquad (1.1)$$

Cette approche a l'avantage évident de pouvoir réutiliser tous les algorithmes classiques dédiés aux problèmes d'optimisation à un seul critère. Cependant, cette approche a aussi deux inconvénients importants. Le premier est dû au fait que pour avoir un ensemble de points bien répartis sur le front Pareto, les différents vecteurs de poids doivent être choisis judicieusement. Il est donc nécessaire d'avoir une bonne connaissance du problème. Le deuxième inconvénient provient du fait que cette méthode ne permet pas, de calculer intégralement la surface de compromis lorsque celle-ci n'est pas convexe.

La méthode lexicographique, classe les critères en fonction d'un ordre d'importance proposé par le décideur. Ensuite, les fonctions critères sont traitées dans cet ordre pour obtenir l'optimum. Le principe est schématisé sur la figure 1.2. L'optimisation séquentielle des différents critères aboutit à la découverte d'une seule solution optimale. L'inconvénient majeur de cette méthode est qu'elle tend à générer des solutions qui sont largement optimisées pour certains critères et très peu pour les autres critères. Les solutions de compromis sont négligées.

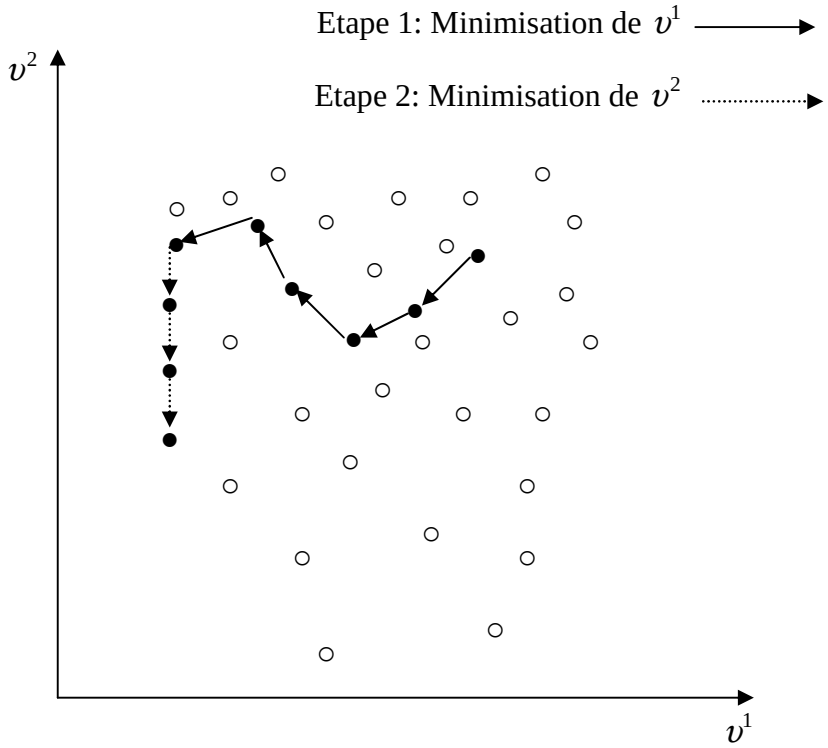

Figure 1.2. La méthode lexicographique: la fonction v^1 est optimisée, puis la fonction v^2, en ne tenant compte que des solutions de coût optimal pour la fonction v^1.

1.3. Concepts de la théorie des graphes

Les graphes représentent un instrument puissant pour modéliser de nombreux problèmes combinatoires, qui serait sans cela difficilement abordables par des techniques classiques.

1.3.1. Définitions

Définition 1.8.

Un graphe orienté $G = (N, A)$ est défini par la donnée d'un ensemble fini et non vide $N = \{1,...,n\}$ appelés ensemble de *sommets* et un ensemble fini $A \subset N \times N$ appelé ensemble d'arcs. Chaque arc est noté par un couple de sommets (i, j) où $i \in N,\ \ j \in N$. On note : $|N| = n$ et $|A| = m$.

Définition 1.9.

Un chemin p_{ij} ou $p\,(i, j)$ joignant deux sommets i et j d'un graphe est une suite d'arcs:

$$p\,(i, j) = \big((i, j_1),(j_1, j_2),...,(j_{t-1}, j)\big).$$

Le chemin est souvent représenté par la suite associée de sommets:

$$(i,\ j_1,..., j_{t-1},\ j)$$

- Un chemin p_{ij} dont tous les sommets sont distincts est un chemin *élémentaire*.
- Le chemin p_{ij} est dit *simple* s'il ne passe pas deux fois par un même arc.
- Un *circuit* Γ est un chemin fermé, i.e un chemin dont l'extrémité terminale coïncide avec l'extrémité initiale.

Définition 1.10.

Un *réseau* (graphe valué) est un triplet (N, A, υ) où (N, A) est un graphe et $\upsilon: A \rightarrow$ R est une fonction de valuation des arcs, associant selon les besoins à chaque arc sa longueur. Pour désigner la longueur d'arc $a = (i, j)$, on utilisera les notations υ_a , $\upsilon(a)$, $\upsilon(i, j)$ ou υ_{ij} . Cette structure est notée $R = (N, A, \upsilon)$.

Définition 1.11.

Le *coût* ou *longueur* d'un chemin p, noté $\upsilon(p)$ dans un réseau est la somme du coût de ses arcs (comptés avec l'ordre de multiplicité de ceux-ci dans p) qui constituent le chemin:

$$\upsilon(p) = \sum_{a \in p} K_a \upsilon(a) = \sum_{(i,j) \in p} K_{ij} \upsilon_{ij}$$

Où k_{ij} est l'ordre de multiplicité de l'arc (i, j) dans le chemin p.

Dans le cas particulier où p est un chemin *simple* : $\upsilon(p) = \sum_{(i,j) \in p} \upsilon_{ij}$

Par convention un chemin qui ne comporte aucun arc est dit *chemin de longueur nulle*.

Remarque 1.4.

Tous les chemins considérés par la suite sont supposés simples.

Définition 1.12.

Un *circuit* Γ tel que $\upsilon(\Gamma) = \sum_{(i,j) \in \Gamma} \upsilon_{ij} < 0$ est appelé circuit absorbant.

1.3.2. Représentation des graphes en machine [8]

Les grandes familles d'implémentation les plus adaptées pour représenter un graphe se basent sur l'utilisation des matrices *d'adjacence* et *d'incidence*. Ces dernières modélisent les existants entre les sommets du graphe.

Les qualificatifs, adjacence et incidence, attribués aux matrices représentant le graphe sont définis en fonction des éléments des matrices. Si ces derniers traduisent la connexion entre les sommets du graphe, la matrice est dite *matrice d'adjacence sommets-sommets*. Dans le cas où ces éléments représentent les liens entre les sommets et les arcs du graphe, la matrice est dite *matrice d'incidences sommets-arcs où matrice d'incidence sommets-arêtes* si les arcs ne sont pas orientés. Dans notre modèle, les arcs du graphe sont orientés et par conséquent nous ne présentons dans ce qui suit que le cas qui nous concerne.

1.3.2.1. Matrice d'adjacence sommets-sommets

La matrice *d'adjacence sommets-sommets* d'un graphe est une matrice de taille $n \times n$ dont les coefficients sont déterminés comme suit :

$$a_{ij} = \begin{cases} 1 & si\ (i,\ j) \in\ A \\ 0 & si\ non \end{cases}$$

Elle ne permet de représenter que des graphes simples ou des 1-graphes, c'est-à-dire des graphes dont chaque paire de sommets est connectée dans chaque sens au plus par un seul arc. Gondran et Minoux dans [8] indiquent que l'utilisation de cette matrice pour la représentation de graphes peu denses peut être améliorée à l'aide de deux tableaux $\overline{\alpha}$ (de dimension $n+1=|N|+1$) et $\overline{\beta}$ (de dimension $m=|A|$).

Pour chaque sommet i la liste de ses successeurs est mémorisée dans le tableau $\overline{\beta}$ entre la position $\overline{\alpha}(i)$ et la position $\overline{\alpha}(i+1)$. Ainsi, un algorithme de recherche du plus court chemin qui énumère toutes les solutions possibles dans le graphe effectue un calcul de cette position pour chaque sommet traité. Ceci peut se révéler coûteux en terme de temps car un sommet est manipulé de très nombreuses fois. De plus, dans le cas où le graphe est valué, la méthode propose l'utilisation d'un troisième tableau de la même taille que $\overline{\beta}$ afin de mémoriser l'évaluation de chaque arc (Figure 1.3).

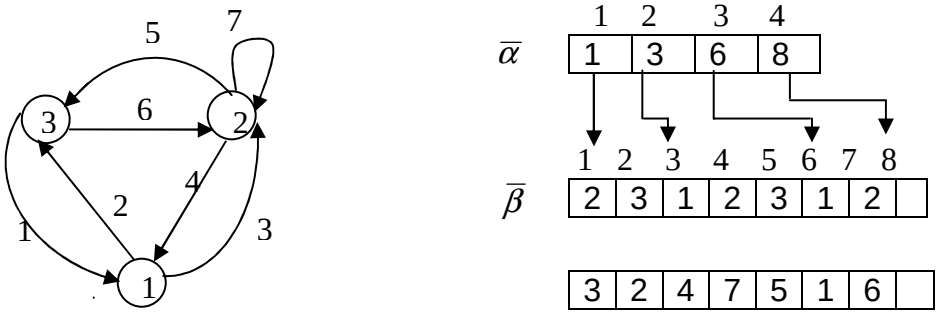

Figure 1.3. Matrice d'adjacence pour un 1-graphe valué.

1.3.2.2. Matrice d'incidence sommets-arcs

La matrice d'incidence *sommets-arcs* d'un graphe orienté est une matrice de taille $m \times n$ (avec $m = |A|$ et $n = |N|$). Les lignes correspondent aux arcs du graphe et les colonnes représentent les sommets. Les éléments de cette matrice U_{ei} (avec $e = (j,k) \in A$ et $i \in N$) sont égaux à :

$$U_{ei} = \begin{cases} 1 & si\ le\ sommet\ i = j \\ -1 & si\ le\ sommet\ i = k \\ 0 & si\ le\ sommet\ i \notin \{j,k\} \end{cases}$$

L'utilisation de la matrice d'incidence permet de décrire un multi-graphe sans cycle mais elle ne permet pas de pallier tous les inconvénients de la matrice d'adjacence. La solution proposée par [8] pour ce cas est à nouveau l'utilisation de deux tableaux dont chacun mémorise une extrémité (base ou tête) d'un arc. De ce fait, la taille de chaque tableau est égale au nombre d'arcs dans le graphe. Dans le cas d'un graphe valué, la méthode utilise un troisième tableau pour la mémorisation des valeurs associées aux arcs. Le modèle résultant pour le graphe de la (figure 1.3) est représenté par les tableaux de la figure 1.4

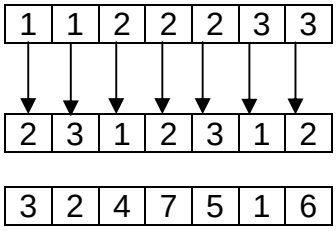

Figure 1.4. Matrice d'incidence pour un graphe valué.

L'implantation choisie joue également un rôle important pour optimiser les performances. De telles matrices, implantées sous forme de tableaux, par exemple, présentent de nombreux éléments nuls dans le cas de graphes peu denses. De nombreux type d'implantation sont donc proposés dans la littérature [9]. Ils reposent sur l'utilisation de structures plus complexes pour l'implémentation des matrices et permettent de réduire, voire même d'éliminer, les éléments nuls.

Parmi ces structures nous pouvons citer, à titre d'exemple, l'utilisation des listes de successeurs. Elle consiste à connecter chaque sommet ses points successeurs concaténés dans une liste chaînée (figure1.5). Cette structure permet un gain important au niveau mémoire. Alors que les matrices d'adjacence et d'incidence sous forme de tableaux représentent un graphe de n sommet et m arcs avec, respectivement, n^2 et $n \times m$ éléments, la liste chaînée nécessite $n+m$ éléments. Ceci est lié à la suppression des zéros ou à l'expansion du graphe en fonction des liens entre ses sommets et arcs.

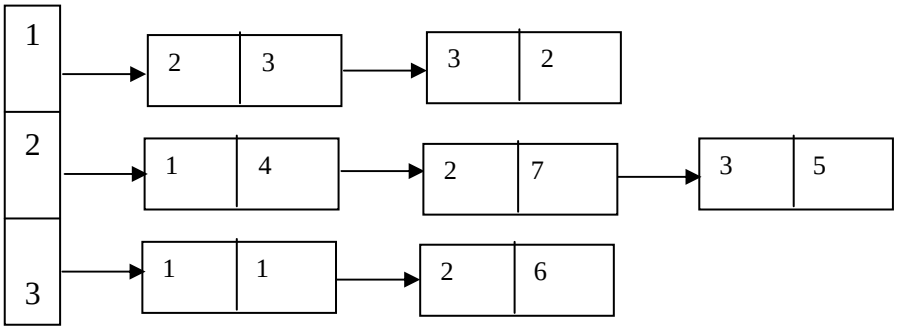

Figure 1.5. Liste de successeurs.

1.3.3. Parcours d'un graphe [10]

La recherche des sommets qui peuvent être atteints dans un graphe à partir d'un sommet donné, est un problème qui se présente fréquemment à l'intérieur d'algorithmes d'optimisation combinatoire. Nous allons décrire ici le cadre général des méthodes d'exploration des sommets d'un graphe.

Nous nous proposons de déterminer dans un graphe G l'ensemble des sommets i tels qu'il existe un chemin de s à i où s est un sommet source.

Le graphe sera donné par ses listes de successeurs, nous noterons $LS(i)$ la liste des successeurs du sommet i.

L'algorithme est donné à la figure1.6. Il s'agit d'une description générale de la méthode de parcours: en effet, lorsque l'on retire de la liste Q le sommet i, il y a plusieurs possibilités de choix du sommet i qui conduisent à des méthodes différentes.

Début

$Q = \{s\}$

Tant que $Q \neq \phi$ **faire**

Début

soit i un élément de Q;

retirer i de Q;

marquer i ;

Pour tout $j \in LS(i)$ ***faire***

Début

Si j n'est pas marquer: introduire j dans Q

fin

fin

fin

Figure 1.6. Algorithme de parcours d'un graphe.

Considérons maintenant les règles de choix du sommet i à extraire de la liste Q.

- Première méthode. Retirer le plus ancien sommet de Q (règle First In - First Out ou FIFO): Q est alors une queue. On obtient un algorithme de parcours en largeur (Breadth First Search ou BFS). On visite les sommets dans l'ordre croissant de leur distance à partir de s, où la distance entre s et i est le nombre minimum d'arcs sur un chemin de s à i.

- Seconde méthode. Retirer le dernier sommet introduit dans Q (règle Last In - First Out ou LIFO): Q est alors une pile et l'on a un algorithme de parcours en profondeur (Depth First Search ou DFS). En partant de s, on construit un chemin aussi long que possible et on revient en arrière lorsqu'il n'est pas possible de continuer à partir du sommet atteint.

Exemple 1.1.

La figure 1.7 représente un graphe orienté G de sommets 1, 2, 3, 4, 5, 6, 7.

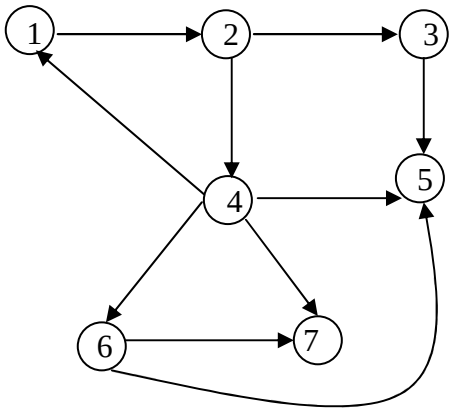

Figure 1.7. Graphe orienté

Un exemple de recherches possibles (figure 1.8 et figure 1.9) en largeur et profondeur. Les résultas de chacune des recherches sont représentés par son arbre de recherche. Un arc existe entre deux sommets i et j si la visite (marquage) de j se fait à partir de i.

La figure 1.8 représente un parcours en largeur, les sommets sont visités depuis le sommet 1 dans l'ordre 2, 3, 4, 5, 6,7.

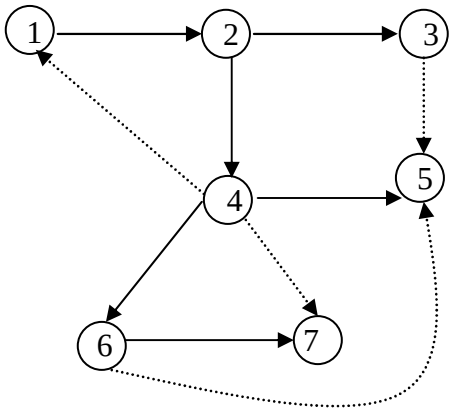

Figure 1.8. Parcours en largeur- ordre du parcours: 1, 2, 3, 4, 5, 6,7.

La figure 1.9 représente, un parcours en profondeur les sommets sont visités depuis 1 dans l'ordre 2, 4, 6, 7, 3, 5.

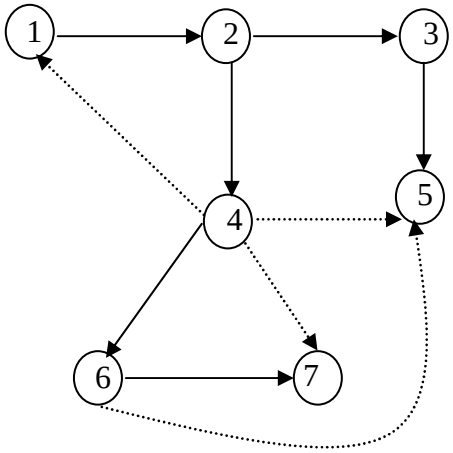

Figure 1.9. Parcours en profondeur- ordre du parcours : 1, 2, 4, 6, 7, 3, 5.

CHAPITRE 2
PROBLEME DE CHEMINEMENT MULTICRITERE

2.1. Introduction

Les problèmes de cheminement constituent un thème classique en recherche opérationnelle dont les applications sont très nombreuses, notamment en transport et en télécommunication. Parmi les problèmes de cheminement les plus anciens et les plus traités on trouve le problème de *plus court chemin*. Il consiste à chercher le meilleur chemin entre deux points, source et puits, afin de minimiser un critère précis, généralement, le temps, la distance ou le coût. Il existe différentes variantes selon la configuration du réseau étudié (avec ou sans circuit, statique ou dynamique,…etc.) et les contraintes définies.

Avec le développement fulgurant de l'aide multicritère à la décision, et, en particulier, suite à la nouvelle démarche dans la formulation des problèmes concrets de décision qui tient compte de tous les points de vue, le problème de cheminement classique monocritère, ne caractérise pas complètement les problèmes pratiques de cheminement.

En effet, dans un réseau de transport ou de télécommunication, plusieurs paramètres peuvent être associés à chaque arc comme le temps, la distance, le coût,…etc.

Il est donc clair qu'on est amené à un problème de décision multicritère particulier, celui de la recherche d'un plus court chemin dans un réseau multicritère.

Dans la première section de ce chapitre, nous introduisons le problème de cheminement monocritère, ensuite nous étudions le problème de plus court chemin de s à t dans un réseau multicritère, puis le problème de plus court chemin de s à tout autre sommet du réseau multicritère.

2.2. Problème de cheminement monocritère

2.2.1 Définitions

Définition 2.1.

Le problème de la recherche d'un plus court chemin joignant un sommet s à un sommet t dans un réseau orienté monocritère, c'est-à-dire dans un graphe G où à chaque arc (i, j) est associé à une seule valeur qui peut représenter une distance, un temps, ou un coût,... etc., est appelé *problème de cheminement monocritère.*

Définition 2.2.

Etant donné un réseau monocritère $R = (N, A, \upsilon)$ et deux sommets s et t. Le problème de plus court chemin entre s et t revient à trouver un chemin p_{st} ou $p(s,t)$ de s à t tel que :

$$\upsilon(p_{st}) = \sum_{(s,t)\in p} \upsilon_{st} \quad \text{soit minimal.}$$

Une définition équivalente de plus court chemin est:

Définition 2.3.

Un plus court chemin (PCC, chemin de longueur minimale) entre deux sommets s et t d'un graphe valué par des valeurs réelles est un chemin dont la longueur est minimale parmi tous les chemins de s à t.

2.2.2. Condition d'existence de la solution

La condition d'existence du plus court chemin est donnée par le théorème 2. 1 [8] ci-dessous:

Théorème 2.1.

Un plus court chemin entre deux sommets s et t d'un graphe existe si parmi tous les chemins p_{st} aucun d'eux ne contient un circuit absorbant.

Preuve

Considérons un chemin p de s à t comprenant un circuit Γ et un chemin p' associé à p et ne comprenant pas de circuit Γ $(p' = p - \Gamma)$. La longueur de p est la somme des longueurs de p' et Γ :

$$v(p) = v(p') + v(\Gamma).$$

Deux cas doivent alors être pris en compte:

- Si $v(\Gamma) \geq 0$, on a $v(p) \geq v(p')$, donc un chemin empruntant un circuit n'est jamais strictement plus court;
- Si $v(\Gamma) < 0$, on a $v(p) < v(p')$, la longueur de chemin p peut diminuer infiniment, donc le réseau n'admet pas de plus court chemin.

Suivant le théorème 2.1, lors de la recherche des plus courts chemins on peut se limiter aux chemins élémentaires, d'où la propriété 2.1

Propriété 2.1.

Tout plus court chemin dans un graphe est un chemin élémentaire (en l'absence de boucles valuées par 0).

Etant donné deux sommets s et t d'un réseau R, trois cas peuvent se présenter:

- Il n'existe pas de chemin de s à t dans R.
- Il existe un chemin de s à t dans R mais pas de chemin de longueur minimum.
- Dans l'ensemble des chemins joignant s à t dans R, il en existe un, de longueur minimum.

Exemple 2.1.

On considère le réseau monocritère suivant (les nombres qui figurent à coté des arcs représentent la longueur de ceux-ci) figure 2.1.

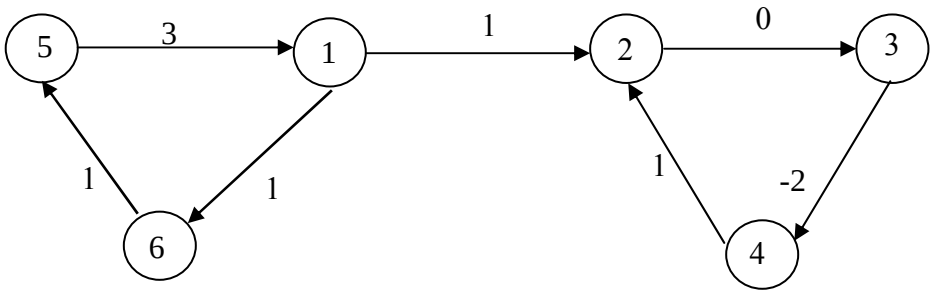

Figure 2.1. Réseau monocritère.

- Il n'existe pas de chemin de 2 à 1 dans R.
- Il existe un chemin de 1 à 2 dans R mais pas de chemin de longueur minimum (considéré l'ensemble des chemins constitués par l'arc (1,2) auquel on adjoint *h* fois le circuit p_{22} = (2, (2,3), 3, (3,4), 4, (4,2) ,2).
- Il existe un plus court chemin de 1 à 5 dans R, le chemin p_{15}=1, (1,6), 6, (6,5) ,5.

2.3. Problème de recherche d'un plus court chemin d'un sommet source s à un sommet puits t dans un réseau multicritère

2.3.1. L'aspect multicritère dans un réseau

Nous appelons $R = (N, A, \upsilon)$ un réseau orienté multicritère, la donnée d'un graphe $G = (N, A)$ où à chaque arc (i, j) est associé un vecteur réel à r composantes. Ce vecteur sera désigné par $\upsilon_{ij} = (\upsilon_{ij}^1, \upsilon_{ij}^2, ... \upsilon_{ij}^r)$. La valeur d'un chemin p est représentée par un vecteur, noté $\upsilon(p)$:

$$\upsilon(p) = (\upsilon^1(p), ..., \upsilon^r(p)) = (\sum_{(i,j)\in p} \upsilon_{ij}^1, ..., \sum_{(i,j)\in p} \upsilon_{ij}^r).$$

Soient p et p' deux chemins, et $\upsilon(p)$, $\upsilon(p')$ leur vecteurs de performance (vecteurs critères).

Définition 2.4.

On dit que $\upsilon(p)$ *domine* $\upsilon(p')$ si et seulement si, pour tout k = 1,…, r, $\upsilon^k(p) \leq \upsilon^k(p')$ avec au moins une inégalité stricte.

Définition 2.5.

Un vecteur de performances $\upsilon(p)$, avec $p \in P_{s,t}$ est dit *non-dominé* s'il n'existe pas un autre chemin $p' \in P_{s,t}$ tel que $\upsilon(p')$ domine $\upsilon(p)$.

Définition 2.6.

Un chemin $p \in P_{s,t}$ est un chemin *efficace* si et seulement si son vecteur de performances $\upsilon(p)$ est non-dominé.

Exemple 2.2.

Considérons le réseau bicritère de sommets s, 1, 2, t de la figure 2.2. suivante

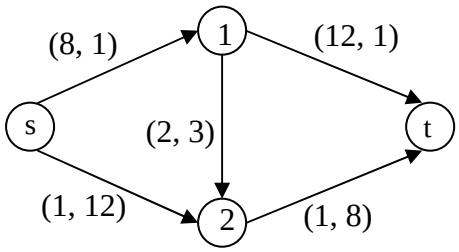

Figure 2.2. Un réseau bicritère.

Il y a trois chemins possibles: p_1= (s ,1, t), p_2 = (s ,1,2, t) et p_3 = (s ,2, t), de vecteur de performances respectifs $\upsilon(p_1)$= $(\upsilon^1(p_1), \upsilon^2(p_1))$ = (20, 2), $\upsilon(p_2)$= $(\upsilon^1(p_2), \upsilon^2(p_2))$ = (11, 12) et $\upsilon(p_3)$= $(\upsilon^1(p_3), \upsilon^2(p_3))$= (2, 20).

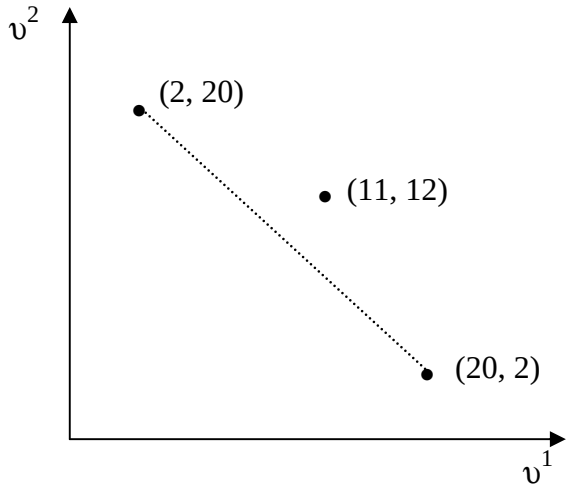

Figure 2.3. Représentation dans l'espace des critères.

Définition 2.7.

On définit les deux problèmes suivants:

(A_1): "trouver un chemin *élémentaire* de "*longueur minimum*" joignant un sommet s à un sommet t dans le réseau multicritère $R = (N, A, \upsilon)$".

(A_2): "trouver un chemin de "*longueur minimum*" joignant un sommet s à un sommet t dans le réseau multicritère $R = (N, A, \upsilon)$".

Soient :

$P'_{s,t}$: l'ensemble de tous les chemins *élémentaires* du sommet s au sommet t dans le réseau R.

$P_{s,t}$: l'ensemble de tous les chemins du sommet s au sommet t dans le réseau R.

Il est clair qu'un chemin réalisant *le minimum d'une composante* du vecteur ne réalise pas nécessairement *le minimum pour l'autre composante*. Le problème posé est typiquement un *problème d'analyse multicritère*.

Remarque 2.1.

Trouver le minimum revient à trouver la solution de *meilleur compromis* dans le contexte multicritère.

Définition 2.8.

Un circuit Γ tel que $\upsilon^k(\Gamma) = \sum_{(i,j)\in\Gamma} \upsilon_{ij}^k < 0$ est appelé *circuit absorbant pour la k^{ème} composante.*

Définition 2.9.

Soit Γ un circuit dans le réseau R. Γ est un circuit de longueur nulle pour le problème de plus court chemin multicritère si et seulement si

$$\upsilon(\Gamma) = (\upsilon^1(\Gamma), \upsilon^2(\Gamma), ..., \upsilon^r(\Gamma)) = (0, 0, ..., 0).$$

Définition 2.10.

Soit Γ un circuit dans le réseau R. Γ est un circuit absorbant pour le problème de plus court chemin multicritère si et seulement si il existe k,

$$k \in \{1,..., r\} \quad tel\ que \quad v^k(\Gamma) < 0 .$$

2.3.2. Conditions nécessaires et suffisantes d'existence de solutions efficaces du problème de plus court chemin de s à t.

2.3.2.1. Cas d'un réseau bicritère

Nous considérons un réseau bicritère $R = (N, A, v)$, avec N= {1,…, n}, dans lequel chaque arc $(i, j) \in A$ est valué par un vecteur à deux composantes, noté $v_{ij} = (v_{ij}^1, v_{ij}^2)$. La valeur d'un chemin p est alors :

$$v(p) = (v^1(p), v^2(p)) = (\sum_{(i,j)\in p} v_{ij}^1, \sum_{(i,j)\in p} v_{ij}^2).$$

Définition 2.11.

On définit les deux problèmes suivants:

(B_1): "trouver un chemin *élémentaire* de "*longueur minimum*" joignant un sommet s à un sommet t dans le réseau bicritère $R = (N, A, v)$".

(B_2): " trouver un chemin de "*longueur minimum*" joignant un sommet s à sommet t dans réseau bicritère $R = (N, A, v)$".

L'existence de solutions admissibles et des solutions efficaces est donnée par la proposition 2.1 ci-dessous.

Proposition 2.1. [33]

Le problème (B_2) admet:

(a)- une solution admissible si et seulement si l'ensemble des sommets $\boldsymbol{y}$ qui sont à la fois des *descendant*s de s et des *ascendants* de t soit non vide.

(b)- *une solution efficace* si et seulement si les conditions suivantes sont satisfaites:

(b1) - le sous-réseau construit sur $\boldsymbol{y}$ ne contient pas de *circuit absorbant* pour chacun des deux critères.

(b2) - le sous-réseau construit sur $\boldsymbol{y}$ ne contient pas *deux circuits* Γ et Γ' vérifiants:

$(v^1(\Gamma),\ v^2(\Gamma)) = (\alpha_1,\ \alpha_2)$

$(v^1(\Gamma'),\ v^2(\Gamma')) = (\beta_1, \beta_2)$ avec $\alpha_i < 0$ *ou* $\beta_i < 0,\ i = 1, 2$

parcourus avec le *même nombre de fois* tels que:

$\alpha_i + \beta_i < 0,\ i = 1, 2.$

Preuve

* (a)- $\boldsymbol{y}$ = $D_{esc}(s) \cap A_{sc}(t)$ où

- $D_{esc}(s)$ = { $i \in N$ / il existe un chemin de s à i} $\cup$ {s}
  = l'ensemble des descendants de s.

- $A_{sc}(t)$ = {$i \in N$ / il existe un chemin de i à t} $\cup$ {t}
  = l'ensemble des ascendants de t.

$\boldsymbol{y} \neq \varnothing \Leftrightarrow$ il existe un chemin de s à t dans R.

* * (b)- ● Montrons la condition nécessaire:

(b1)- supposons que le sous-réseau construit sur $\boldsymbol{y}$ admet un circuit absorbant pour le premier critère (ou pour le second critère).
Sans perte de généralité, nous supposons que:

$$(v^1(\Gamma),\ v^2(\Gamma)) = (\alpha_1,\ \alpha_2)\ \ avec\ \ \alpha_1 < 0\ \ et\ \ \alpha_2 \geq 0$$

S'il existe un chemin p_{si} de s à un sommet i du circuit Γ et un chemin p_{it} de i à un sommet t, alors le chemin obtenu en parcourant le chemin p_{si} puis *h* fois le circuit Γ $(h \in N)$ et ensuite le chemin p_{it} est un chemin de s à t de longueur *négative* et est aussi grand qu'on veut en valeur absolue (en prenant *h* suffisamment grand), pour le premier critère, et, chemin de longueur *positive* ou *constante* pour le second critère.

Donc le chemin $p_{st} = p_{si} \cup \Gamma \cup p_{it}$ a pour longueur à *la première étape* (étape où il est considéré comme étant un chemin élémentaire):

$$(\upsilon^1(p_{st}), \upsilon^2(p_{st})) = (\upsilon^1(p_{si}) + \upsilon^1(p_{it}),\ \upsilon^2(p_{si}) + \upsilon^2(p_{it}))$$

et à la $h^{éme}$ *étape* a pour longueur.

$$\begin{aligned}(\upsilon^1(p_{st}),\ \upsilon^2(p_{st})) &= (\upsilon^1(p_{si}) + \upsilon^1(\Gamma) + \upsilon^1(p_{it}),\ \upsilon^2(p_{si}) + \upsilon^2(\Gamma) + \upsilon^2(p_{it})) \\ &= (\upsilon^1(p_{si}) + h\alpha_1 + \upsilon^1(p_{it}),\ \upsilon^2(p_{si}) + h\alpha_2 + \upsilon^2(p_{it}))\end{aligned}$$

Dans le cas où le chemin élémentaire est le seul chemin efficace, alors lorsque *h* tend vers l'infini, le problème (B_2) n'admet pas de solution.
Dans le cas où il existe un autre chemin efficace p'_{st}, alors on a:

<u>1er cas:</u>

$$\upsilon^1(p_{st}) << \upsilon^1(p'_{st}) \quad et \quad \upsilon^2(p_{st}) \leq \upsilon^2(p'_{st})$$

Et dans ce cas, le chemin p_{st} devient le seul chemin efficace et lorsque *h* tend vers l'infini, le problème (B_2) n'admet pas de solution.

<u>2éme cas:</u>

$$\upsilon^1(p_{st}) << \upsilon^1(p'_{st}) \quad et \quad \upsilon^2(p_{st}) > \upsilon^2(p'_{st})$$

En tenant compte de l'importance relative des critères (remarque 1.2 et théorème1.1. du chapitre 1) nous pouvons conclure qu'à la $h^{éme}$ étape (*h*

suffisamment grand) le chemin p_{st} sera plus efficace que le chemin $p_{st}^{'}$, et lorsque *h* tend vers l'infini, le problème (B_2) n'admet pas de solution.

(b2)- supposons, sans perte de généralité, que le sous-réseau construit sur ***y*** admet seulement deux circuits.

Soient Γ et $\Gamma^{'}$ deux circuits tels que :

$(\upsilon^1(\Gamma),\ \upsilon^2(\Gamma)) = (\alpha_1,\ \alpha_2)$

$(\upsilon^1(\Gamma^{'}),\ \upsilon^2(\Gamma^{'})) = (\beta_1, \beta_2)$ avec $\alpha_1 < 0$ *ou* $\beta_1 < 0$

parcourus avec le *même nombre de fois* tels que:

$\alpha_1 + \beta_1 < 0$.

S'il existe un chemin p_{si} de s à un sommet i du circuit Γ puis un chemin p_{ij} de i à un sommet j du circuit $\Gamma^{'}$ et un chemin p_{jt} de j à sommet t, alors le chemin obtenu en parcourant le chemin p_{si} puis *h* fois le circuit Γ, ensuite le chemin p_{ij} puis *h* fois le circuit $\Gamma^{'}$ et enfin le chemin p_{jt} est un chemin de s à t de longueur *négative* et est aussi grand qu'on veut en valeur absolue (en prenant *h* suffisamment grand), pour le premier critère, et chemin de longueur *positive* ou *constante* pour le second critère.

Donc le chemin $p_{st} = p_{si} \cup \Gamma \cup p_{ij} \cup \Gamma^{'} \cup p_{jt}$ a pour longueur à *la première étape* (étape où il est considéré comme étant un chemin élémentaire):

$$(\upsilon^1(p_{st}), \upsilon^2(p_{st})) = (\upsilon^1(p_{si}) + \upsilon^1(p_{ij}) + \upsilon^1(p_{jt}),\ \upsilon^2(p_{si}) + \upsilon^2(p_{ij}) + \upsilon^2(p_{jt}))$$

et à la $h^{éme}$ *étape* a pour longueur:

$$\begin{pmatrix} \upsilon^1(p_{st}) \\ \upsilon^2(p_{st}) \end{pmatrix} = \begin{pmatrix} \upsilon^1(p_{si}) + \upsilon^1(\Gamma) + \upsilon^1(p_{ij}) + \upsilon^1(\Gamma^{'}) + \upsilon^1(p_{jt}) \\ \upsilon^2(p_{si}) + \upsilon^2(\Gamma) + \upsilon^2(p_{ij}) + \upsilon^2(\Gamma^{'}) + \upsilon^2(p_{jt}) \end{pmatrix}$$

$$= \begin{pmatrix} \upsilon^1(p_{si}) + h(\alpha_1 + \beta_1) + \upsilon^1(p_{ij}) + \upsilon^1(p_{jt}) \\ \upsilon^2(p_{si}) + h(\alpha_2 + \beta_2) + \upsilon^2(p_{ij}) + \upsilon^2(p_{jt}) \end{pmatrix}$$

Nous achevons la preuve de la condition nécessaire (b2) avec un raisonnement analogue à celui de (b1).

●● Montrons la condition suffisante.

Sans perte de généralité, nous supposons que le sous-réseau construit sur ***y*** possède un seul circuit tel que:

$$(\upsilon^1(\Gamma),\ \upsilon^2(\Gamma\ )) = (\alpha_1,\ \alpha_2)$$

alors nous aurons les cas suivants:

- $\alpha_1 > 0,\quad \alpha_2 > 0$
- $\alpha_1 > 0,\quad \alpha_2 = 0$ (ou vice versa)
- $\alpha_1 = 0,\quad \alpha_2 = 0$

S'il existe un chemin p_{st} passant par ce circuit, c'est-à-dire $p_{st} = p_{si} \cup \Gamma \cup p_{it}$, alors son vecteur des longueurs sera:

$$(\upsilon^1(p_{st}),\ \upsilon^2(p_{st})) = (\upsilon^1(p_{si}) + \upsilon^1(\Gamma) + \upsilon^1(p_{it}),\ \upsilon^2(p_{si}) + \upsilon^2(\Gamma) + \upsilon^2(p_{it}))$$

Ce chemin n'est pas un chemin élémentaire, appelons p'_{st} le chemin p_{st} auquel on a retiré le circuit Γ, on aura :

$$(\upsilon^1(p'_{st}),\ \upsilon^2(p'_{st})) = (\upsilon^1(p_{si}) + \upsilon^1(p_{it}),\ \upsilon^2(p_{si}) + \upsilon^2(p_{it}))$$

Dans les trois cas, on a:

$$\upsilon^k(p'_{st}) \le \upsilon^k(p_{st}) \quad k = 1, 2$$

Donc p'_{st} est *la solution efficace*. Ce qui montre que le problème (B_2) possède des solutions.

Le problème (B_1) est plus difficile que le problème (B_2) mais, on peut montrer, que sous certaines conditions, ils peuvent avoir le même ensemble de solutions efficaces.

Proposition 2.2.

Si les conditions suivantes sont satisfaites:

(1)- pour tout circuit Γ dans un réseau bicritère $R = (N, A, \upsilon)$,

$$\upsilon^1(\Gamma) \geq 0 \ \ et \ \ \upsilon^2(\Gamma) \geq 0$$

(2)- pour deux circuits Γ et Γ' dans un réseau bicritère $R = (N, A, \upsilon)$ vérifiants:

$(\upsilon^1(\Gamma), \ \upsilon^2(\Gamma)) = (\alpha_1, \ \alpha_2)$

$(\upsilon^1(\Gamma'), \ \upsilon^2(\Gamma')) = (\beta_1, \beta_2)$ avec $\alpha_1 \geq 0 \ et \ \beta_2 \geq 0$

parcourus avec le *même nombre de fois* tels que:

$\alpha_1 + \beta_1 \geq 0 \quad et \quad \alpha_2 + \beta_2 \geq 0$.

alors, les problèmes (B_2) et (B_1) sont équivalents et dans ce sens chaque chemin efficace pour le problème (B_2) est un chemin efficace pour le problème (B_1) et inversement.

Preuve

(1)- Soit un circuit Γ dans un réseau bicritère R tel que:

$$(\upsilon^1(\Gamma), \ \upsilon^2(\Gamma)) = (\alpha_1, \ \alpha_2) \quad avec \ \ \alpha_1 \geq 0 \, et \ \ \alpha_2 \geq 0$$

et soit $i \in \Gamma$, il existe un chemin p_{si} de s à i et un chemin p_{it} de i à t, dans un réseau $R = (N, A, \upsilon)$ bicritère. Ainsi, nous pouvons définir le chemin $p_{st} = p_{si} \cup \Gamma \cup p_{it}$, alors son vecteur des longueurs sera :

$$(\upsilon^1(p_{st}), \upsilon^2(p_{st})) = (\upsilon^1(p_{si}) + \upsilon^1(\Gamma) + \upsilon^1(p_{it}), \ \upsilon^2(p_{si}) + \upsilon^2(\Gamma) + \upsilon^2(p_{it})).$$

Ce chemin n'est pas élémentaire; appelons p'_{st} le chemin p_{st} auquel on a retiré le circuit Γ, on aura :

$$(\upsilon^1(p'_{st}), \upsilon^2(p'_{st})) = (\upsilon^1(p_{si}) + \upsilon^1(p_{it}), \ \upsilon^2(p_{si}) + \upsilon^2(p_{it})).$$

Dans ce cas, on a :

$$\upsilon^k(p'_{st}) \leq \upsilon^k(p_{st}) \quad k = 1, 2$$

Donc le chemin p'_{st} est une solution efficace. Dans ce cas les deux problèmes sont équivalents.

S'il existe un circuit Γ avec $v^l(\Gamma) < 0$ pour au moins un $l \in \{1, 2\}$, alors il existe un plus court chemin contenant ce circuit et le problème (B_1) n'admet pas une solution réalisable. Dans ce cas, les deux problèmes ne sont pas équivalents.

(2)- Soient Γ et Γ' deux circuits dans un réseau $R = (N, A, v)$ bicritère tels que:

$(v^1(\Gamma),\ v^2(\Gamma)) = (\alpha_1,\ \alpha_2)$

$(v^1(\Gamma'),\ v^2(\Gamma')) = (\beta_1, \beta_2)$ avec $\alpha_1 \geq 0$ *et* $\beta_2 \geq 0$

parcourus avec le *même nombre de fois* tels que:

$\alpha_1 + \beta_1 \geq 0$ *et* $\alpha_2 + \beta_2 \geq 0$.

Avec le même raisonnement que dans (1), on aboutira à ce que les deux problèmes sont équivalents.

2.3.2.2. Cas d'un réseau multicritère

l'existence de solutions admissibles et efficaces dans le cas d'un réseau multicritère est donnée par la proposition suivante.

Proposition 2.3. [33]

Le problème (A_2) admet:

(a)- *une solution admissible* si et seulement si l'ensemble des sommets ***y*** qui sont à la fois des *descendant*s de s et des *ascendants* de t soit non vide.

(b)- *une solution efficace* si et seulement si les conditions suivantes sont satisfaites:

(b1)- le sous-réseau construit sur ***y*** ne contient pas de *circuit absorbant* pour au moins un critère.

(b2)- le sous-réseau construit sur ***y*** ne contient pas *deux circuits* Γ et Γ' vérifiants:

$(\upsilon^1(\Gamma),\ \upsilon^2(\Gamma),\ ...,\ \upsilon^r(\Gamma)) = (\alpha_1, \alpha_2, ..., \alpha_r)$

$(\upsilon^1(\Gamma'),\ \upsilon^2(\upsilon(\Gamma'), ..., \upsilon^r(\Gamma')) = (\beta_1, \beta_2, ..., \beta_r)$ avec

$\alpha_i < 0$ ou $\beta_i < 0$, pour $i \in I$ tels que : $I \subset \{1,...,r\}$

parcourus avec le *même nombre de fois* tels que:

$\alpha_i + \beta_i < 0, \quad i \in I$.

Preuve

Analogue à la proposition 2.1.

Le problème (A_1) est plus difficile que le problème (A_2) mais, on peut montrer, que sous certaines conditions, ils peuvent avoir le même ensemble de solutions efficaces.

Proposition 2.4.

Si les conditions suivantes sont satisfaites:

(1)- pour tout circuit Γ dans un réseau multicritère $R = (N, A, \upsilon)$, $\upsilon^k(\Gamma) \geq 0$ pour tout k = 1,…., r.

(2)- pour deux circuits Γ et Γ' dans un réseau multicritère $R = (N, A, \upsilon)$, vérifiants:

$(\upsilon^1(\Gamma),\ \upsilon^2(\Gamma),\ ...,\ \upsilon^r(\Gamma)) = (\alpha_1,\ \alpha_2, ..., \alpha_r)$

$(\upsilon^1(\Gamma'),\ \upsilon^2(\Gamma'), ..., \upsilon^r(\Gamma')) = (\beta_1,\ \beta_2, ..., \beta_r)$ avec

$\alpha_i \geq 0$, pour $i \in I_1$; $\beta_i \geq 0$, pour $i \in I_2$ tels que : $I_1 \cup I_2 = \{1,...,r\}$

parcourus avec le *même nombre de fois* tels que:

$\alpha_i + \beta_i \geq 0, \quad i = 1,..., r$.

alors, les problèmes (A_2) et (A_1) sont équivalents et dans ce sens chaque chemin efficace pour le problème (A_2) est un chemin efficace pour le problème (A_1) et inversement.

Preuve

Analogue à la proposition 2.2.

2.4 Problème de recherche d'un plus court chemin d'un sommet source s à tout autre sommet dans un réseau multicritère

Dans cette section, on s'intéresse à un autre type de problème de plus court chemin multicritère: la recherche d'un plus court chemin d'un sommet s donné à tout autre sommet i du réseau multicritère R.

Définition 2.12.

On définit le problème suivant:

(C_1): "A chaque sommet i, trouver un chemin *élémentaire* de *"longueur minimum*" joignant un sommet s donné à i dans le réseau multicritère $R = (N, A, \upsilon)$".

Et soit

$P_{s,i}$: l'ensemble de tous les chemins élémentaires de s à i, $\forall\, i \in N - \{s\}$ dans le réseau multicritère $R = (N, A, \upsilon)$.

Définition 2.13.

Un sommet s d'un graphe $G = (N, A)$ est une *racine* s'il existe dans le graphe G un chemin joignant s à i, pour tout $i \in N - \{s\}$.

Définition 2.14.

Un *arbre* est un graphe connexe sans cycle.

Définition 2.15.

Un graphe $G = (N, A)$ sur $n \geq 2$ sommets est une *arborescence* de racine s si et seulement si:

- s est une racine dans G.
- G est un arbre.

Définition 2.16.

On appelle *source* un sommet à partir duquel on peut atteindre tous les autres sommets du graphe.

Définition 2.17.

On appelle *arborescence des plus courts chemins* une arborescence de G, de racine s et dont chaque chemin de s à i est un plus court chemin de s à i dans G, pour tout sommet i dans G.

2.4.1. Conditions nécessaires et suffisantes d'existence de solutions efficaces du problème de plus court chemin d'un sommet source s à tout autre sommet

Définition 2.18. [11]

Une arborescence T est *dominée* s'il existe un sommet $i \in N$ (i est un sommet pendant de T) pour lequel le chemin p_{si} est dominé.

Définition 2.19.

On dit qu'une arborescence est *efficace* (*non-dominée*) si toutes ses branches sont des chemins efficaces.

2.4.1.1. Cas d'un réseau bicritère:

On définit le problème suivant:

(C_2): "A chaque sommet i, trouver un chemin de *"longueur minimum*" joignant un sommet s donné à i dans le réseau bicritère $R = (N, A, v)$".

L'existence de solutions admissibles et efficaces est donnée par la proposition 2.5.

Proposition 2.5.

Le problème (C_2) admet :

(a)- une solution admissible si et seulement si s est racine dans le réseau R.

(b)- une solution efficace si et seulement si les conditions suivantes sont satisfaites:

(b1) - le réseau R ne contient pas de *circuit absorbant* pour chacun des deux critères.

(b2) - le réseau R ne contient pas *deux circuits* Γ et Γ' vérifiants:

$(\upsilon^1(\Gamma),\ \upsilon^2(\Gamma)) = (\alpha_1,\ \alpha_2)$

$(\upsilon^1(\Gamma'),\ \upsilon^2(\Gamma')) = (\beta_1, \beta_2)$ avec $\alpha_i < 0$ *ou* $\beta_i < 0,\ i = 1, 2$

parcourus avec le *même nombre de fois* tels que:

$\alpha_i + \beta_i < 0,\ i = 1, 2.$

Preuve

* (a)- est évident.

** (b)- voir la proposition 2.1. de la section 2.3.

2.4.1.2. Cas d'un réseau multicritère:

On définit le problème suivant:

(C_3): "A chaque sommet i, trouver un chemin de *longueur minimum* joignant un sommet s donné à i dans le réseau multicritère $R = (N, A, \upsilon)$".

l'existence de solutions admissibles et efficaces dans le cas d'un réseau multicritère est donnée par la proposition 2.6.

Proposition 2.6.

Le problème (C_3) admet:

(a)- une solution admissible si et seulement si s est racine dans le réseau R.

(b)- une solution efficace si et seulement si les conditions suivantes sont satisfaites:

(b1) - le réseau R ne contient pas de *circuit absorbant* pour au moins un critère.

(b2) - le réseau R ne contient pas *deux circuits* Γ et Γ' vérifiants:

$(v^1(\Gamma),\ v^2(\Gamma),\ ...,\ v^r(\Gamma)) = (\alpha_1, \alpha_2, ..., \alpha_r)$

$(v^1(\Gamma'),\ v^2(v(\Gamma'), ...,\ v^r(\Gamma')) = (\beta_1, \beta_2, ..., \beta_r)$ avec

$\alpha_i < 0$ ou $\beta_i < 0$, pour $i \in I$ tels que : $I \subset \{1,...,r\}$

parcourus avec le *même nombre de fois* tels que:

$\alpha_i + \beta_i < 0, \quad i \in I$.

Preuve

* (a)- est évident.

** (b)- voir la proposition 2.1. de la section 2.3.

CHAPITRE 3
PROBLEME DE PLUS COURT CHEMIN : SITUATION DU POINT DE VUE SCIENTIFIQUE

3.1. Introduction

Le problème de chemin optimum est considéré comme l'un des problèmes d'optimisation combinatoire les plus importants [1] et les plus étudiés [12]. La littérature consacrée à ce problème est conséquente. En effet, pendant plus de cinquante ans, plusieurs types de solutions ont été étudiés et proposés pour un domaine large et varié d'application. Cette diversité se traduit par l'existence de nombreuses variantes du problème de chemin optimum en fonction des hypothèses et contraintes considérées. Cependant, jusqu'à présent, il n'existe pas de solution générique capable de résoudre le problème de plus court chemin quelque soit le nombre de paramètres et quelque soit la configuration du réseau [13]. En effet, l'utilisation des algorithmes pionniers nécessite généralement des adaptations en fonction du problème traité. Ainsi, il existe quasiment autant de versions modifiées de l'algorithme de Dijkstra que de problématiques concrètes ayant été résolues avec cet algorithme.

Ce chapitre se décompose en 4 sections: dans la section 3.2, on donne les caractéristiques du problème de plus court chemin. La section 3.3 est consacrée à la complexité du problème de plus court chemin pour les deux variantes monocritère et multicritère. Dans la section 3.4, on présente les algorithmes de recherche de plus court chemin monocritère.

3.2. Caractéristiques du problème

Le problème de plus court chemin dépend:

- du type de réseau considéré;
- du nombre des critères à minimiser;
- de la façon dont la recherche de l'itinéraire est effectuée.

Comme le réseau se distingue par le type de données disponibles, le problème de plus court chemin est, de plus, qualifié de statique, dynamique ou stochastique,...etc., en fonction du type de données disponibles.

Dans tous les cas, trouver le chemin optimum revient à chercher l'itinéraire qui minimise ou qui maximise une fonction coût spécifique. Cette recherche est définie en fonction des critères considérés. On parle du chemin le plus rapide, le moins cher ou le plus court selon que l'objectif visé est la minimisation du temps, du coût ou de la distance. Dans le cas d'une maximisation, on parle alors du chemin le plus lent, le plus cher ou le plus long. Parmi ces variétés d'objectifs de recherche, il est possible d'en choisir une pour évaluer la qualité du chemin. Dans ce cas, le problème traité est *monocritère* [14]. Dans la réalité, les décideurs utilisent plusieurs critères afin de déterminer le meilleur itinéraire. On parle alors de recherche *bicritère* si le nombre des critères est égal à deux [15], [16], [17] ou *multicritère*, [18], [4], [11], s'il est supérieur à deux.

Finalement, la recherche du chemin diffère selon le nombre de points de départ (ou points source) et de points d'arrivée (ou points puits) sélectionnés. Elle peut être effectuée:

- d'un point source unique à un point puits unique lui aussi, et dans ce cas le problème est dit *single source single destination shortest path* ou *one-to - one shortest path* ;

- d'un point à tout autre point du réseau et dans ce cas, on parle de *one-to-all shortest path.* Ce type de recherche est l'exemple type du problème d'arbre de couverture, *the shortest path tree* (SPT);

- entre toutes les paires de sommets dans le réseau, ce qui correspond à *all-to-all shortest path* ou encor *All-pairs shortest path*. Dans le cas *monocritère* où une résolution polynomiale est possible, cette méthode peut être considérée comme m exécutions de la méthode *one-to-one* (où m est le nombre de paires possibles) ou alors n exécutions de la méthode *one-to- all* (où n est le nombre de sommets dans le réseau) [19].

En fonction de l'approche de résolution utilisée, ces caractéristiques nous ont permis, de grouper la plupart des travaux réalisés dans le domaine, en trois classes principales: ceux qui se basent sur l'utilisation d'approches combinatoires, telles que la méthode de fixation d'étiquettes *label-setting* [2], la méthode de correction d'étiquettes *label-correcting* [19] ou leur hybridation, les travaux qui utilisent la programmation linéaire (PL) [21], par exemple la méthode du simplexe, et finalement, ceux à base de techniques algébriques, par exemple l'algorithme de Floyd-warshall [22]. Les approches des deux premières classes sont principalement conçues pour la résolution du plus court chemin pour un point source, *single source shortest path* (SSSP). Les algorithmes algébriques sont plus appropriés pour résoudre le problème de plus court chemin *all-to-all*.

Les algorithmes de recherche de plus court chemin multicritère se décomposent en trois classes :

- les algorithmes marquage (d'étiquetage) multiples qui correspondent à une généralisation au contexte multicritère des algorithmes de problèmes classiques;
- les algorithmes basés sur une énumération des k plus courts chemins (les algorithmes de rangement des chemins);
- les algorithmes en deux phases qui, en une première phase, utilisent la programmation linéaire pour déterminer un sous-ensemble de solutions efficaces (à savoir les solutions dites supportées) puis, en une deuxième phase, s'appuient sur ces solutions pour exhiber les solutions efficaces restantes (dites non-supportées).

Toutefois, la première classe peut être divisée en deux familles: les algorithmes de marquage permanent (fixation d'étiquettes) dans lesquels une étiquette est maintenue de façon permanente dans chaque itération, et les algorithmes de marquage temporaire (correction d'étiquettes) dans lesquels les étiquettes ne deviennent permanentes que dans la dernière itération.

Dans ce travail, nous nous intéressons alors à la famille des algorithmes de marquage permanent.

3.3. Complexité du problème de plus court chemin

Le problème de plus court chemin monocritère est un problème efficacement résolu avec des algorithmes polynomiaux [8]. La complexité maximale des algorithmes pionniers pour cette variante du problème est estimée à $O(|A||N|)$, avec $|A|$ le nombre d'arcs dans le graphe et $|N|$ le nombre de sommets. Cette valeur de complexité est vraie pour toutes les versions d'algorithmes à l'exception de l'algorithme d'Esopo et Pape [23], dont la complexité est polynomiale, mais qui engendre une complexité exponentielle maximale égale à $O\left(|A|2^{|N|}\right)$ pour des réseaux particuliers [9]. Cependant, même un algorithme de complexité polynomiale peut nécessiter un temps de calcul très important dans le cas de graphes denses

Dans le cas d'un réseau multicritère, la variante *multicritère* du problème de plus court chemin est NP-complète [1]. Toutefois, utiliser une stratégie de résolution qui transforme le problème *multicritère* en un problème *monocritère* permet de conserver une complexité polynomiale. C'est le cas, par exemple, pour les algorithmes qui se basent sur la stratégie *ordre lexicographique* [24] ou sur l'agrégation des critères [6].

En l'absence de décomposition ou de transformation du problème, la résolution exacte devient impossible en pratique. L'application du concept de Pareto dominance (voir chapitre.1) par exemple, correspond à une complexité exponentielle d'ordre $O\left(2^{|N|}\right)$, avec N l'ensemble des sommets [25]. Cette complexité provient du nombre de longueurs non dominées au niveau de chaque sommet, qui peut être exponentiel. En effet, trouver l'ensemble de chemins non dominés revient à déterminer toutes les parties de l'ensemble de sommets N. Une résolution exacte est donc d'autant moins envisageable, comme le montre le tableau (3.1) suivant, extrait de [9] qui indique déjà 36 années de calcul pour une complexité en $O\left(2^n\right)$ lorsque n vaut 50. Ceci condamne par avance toute tentative de résolution *multicritère* avec un algorithme itératif cherchant l'ensemble des chemins non-dominés dans le graphe.

Tableau 3.1. Croissance du temps de calcul pour différentes complexités

taille / complexité	20	50	100	200	500	1000
$10\ n^3$	0,02 s	1 s	10 s	1 min	21 min	27 h
2^n	1 s	36 ans	----	----	----	----
3^n	58 min	2.10^{11} ans	----	----	----	----

3.4. Algorithmes de recherche de plus court chemin monocritère.

3.4.1. Notations et définitions

Les données d'entrée pour un algorithme de recherche du plus court chemin sont représentées par le triplet (G, s, υ), où $G(N, A)$ est un graphe orienté, $s \in N$ est le sommet source et $\upsilon : A \to \mathrm{R}$ est une fonction coût (poids) qui attribue à chaque arc (i, j) du graphe son évaluation υ_{ij}. La majorité des algorithmes de recherche de plus court chemin utilise la notion de fonction potentielle. Cette dernière est une fonction réelle f, évaluée sur les sommets, dont la valeur est fonction du coût attribué par la fonction υ aux arcs du graphe. Soit la fonction υ, la fonction $f : N \to \mathrm{R}$ est définie par:

$$f(j) = f(i) + \upsilon_{ij} \quad , \quad \forall \quad (i, j) \in A.$$

Nous supposons que le graphe $G(N, A)$ est connexe et nous définissons, pour chaque sommet $i \in N$, l'ensemble $FS(i)$ de ses arcs sortants, *forward star*, tel que: $FS(i) = \{(i, j) \in A\}$, et l'ensemble $BS(i)$ des arcs entrants au sommet i, *backwar star*, tel que: $BS(i) = \{(j, i) \in A\}$.

3.4.2. Problème de l'arbre de couverture

Le problème de l'arbre de couverture (*shortest path tree* SPT) est un problème *one-to-all*. Il est défini par construction, à partir d'un point source s du graphe G, de la meilleure arborescence T, autrement dit, du sous graphe dans lequel le chemin reliant le sommet s à tous les autres sommets du graphe est le meilleur chemin. Dans le cas où le graphe contient un cycle à coût négatif, le problème d'arbre de couverture n'a pas de solution.

Généralement, afin de résoudre le SPT, les travaux présentés dans la littérature se basent sur des méthodes d'étiquetage (marquage): la méthode de fixation d'étiquettes *label-setting method* et la méthode de correction d'étiquettes *label-correcting method*. Le corps principal de ces deux méthodes est la fonction d'étiquetage dont le principe est le suivant:

chaque sommet de G est étiqueté (marqué) par le triplet $(f(i), B(i), S(i))$ où $f(i)$ est la valeur potentielle du sommet i, $B(i)$ le sommet base du meilleur arc entrant dans i et $S(i) \in$ {non- atteint, étiqueté, scanné} le statut du sommet. La valeur potentielle du sommet i est appelée distance du chemin par rapport au sommet source. Initialement, pour chaque sommet i, $f(i) = \infty$, $B(i) = nul$ et $S(i) =$ non-atteint. L'algorithme initialise les deux paramètres $f(s)$ et $S(s)$ du sommet source s à $f(s) = 0$ et $S(s) =$ étiqueté. La procédure scanne() décrite par le pseudo-code de la figure 3.1 est ensuite appliquée pour mettre à jour les étiquettes $f(i)$ des autres sommets.

```
Procédure scanne (i)
   Pour tout (i, j) ∈ FS(i) faire
      Si f(i) + υ_ij < f(j) alors
         f(j) ← f(i) + υ_ij ;
         S(j) ← étiqueté;
         B(j) ← i ;
      fin_si
   fin_pour
      S(i) ← scanne;
fin
```

Figure.3.1. Code principal de la procédure d'étiquetage.

A chaque itération, la fonction scanne() explore les arcs de l'ensemble $FS(i)$ afin de mettre à jour la valeur de $f(j)$ de leur sommet tête. Si cette valeur est supérieure à la somme de $f(i)$ et de v_{ij}, alors elle est remplacée par la valeur de la somme, et le sommet i est ajouté à $B(j)$ comme meilleur prédécesseur. Cette dernière opération est nécessaire pour la construction du chemin final.

La fonction s'arrête si et seulement si tous les sommets du graphe sont couverts par l'arbre ou si un cycle à valeur négative est détecté. Une opération commune à tous les algorithmes qui se basent sur la méthode d'étiquetage, est l'utilisation d'une liste Q qui sert à mémoriser les sommets déjà étiquetés ainsi que leurs statuts. La principale différence entre les algorithmes de fixation et de correction d'étiquettes réside dans la manipulation de cette liste et de ses éléments. Généralement, les sommets sont traités en fonction de leur valeur de distance. Celui pour lequel cette valeur est minimale, est sélectionné pour être scanné. Ceci constitue le principe de base de l'algorithme de fixation d'étiquettes. Les algorithmes utilisant d'autres stratégies pour traiter les sommets sont classés comme des algorithmes de correction d'étiquettes. Beaucoup de structures des données ont été développées pour améliorer les performances de ces algorithmes. Ce qui s'est traduit par le développement de plusieurs stratégies nouvelles.

3.4.2.1. Algorithme à fixation d'étiquettes.

Cette méthode repose sur l'approche du proche voisin (*shortest first search*). Sa première version à été proposée par Dijkstra en 1959 [2]. Pour sélectionner le sommet à scanner, cet algorithme utilise la fonction f associée aux sommets. Le sommet sélectionné de la liste Q est celui qui possède la plus petite valeur de f. Le corps principal de cet algorithme est présenté dans la figure 3. 2 suivante.

$f(s) \leftarrow 0$;
Pour tout $i \in N - \{s\}$ **faire** $f(i) \leftarrow \infty$;
$Q \leftarrow \{1 = s\}$;
$\overline{Q} = \{2,...,n\}$;
scanne (s);
tant que $|\overline{Q}| \neq 0$ **faire**
$j =$ Meilleur_sommet $(\overline{Q})$;
scanne (j);
$Q \leftarrow Q \cup \{j\}$;
$\overline{Q} \leftarrow \overline{Q} - \{j\}$;
fin_tant que

Figure.3.2. Algorithme de Dijkstra.

Dans le pseudo-code de la figure 3.2, la fonction Meilleur_sommet$(\overline{Q})$ est une fonction de sélection qui renvoie le sommet qui possède la valeur de f la plus petite. La liste $\overline{Q}$ est une liste complémentaire à la liste Q, c'est-à-dire, $\overline{Q} = N - Q$. Le fonctionnement de cet algorithme est lié aux valeurs de υ. En effet, l'algorithme n'est valide que si ses valeurs sont positives. Comme on peut le constater dans le pseudo-code présenté dans la figure 3.2, la complexité des algorithmes basée sur la méthode de fixation des étiquettes dépend de la procédure de sélection utilisée (Meilleur_sommet()) ainsi que de son implémentation. Une méthode naïve qui consiste à comparer tous les sommets de la liste est de complexité $O\left(|N|^2\right)$. Une implémentation basée sur la stratégie de Fibonacci [13] est de complexité $O\left(|A| + |N| \log |N|\right)$. D'autres implémentations ont été également proposées, pour plus de détails, le lecteur peut se référer à [26].

3.4.2.2. Algorithme à correction d'étiquettes.

Dans cette méthode, l'opération de sélection est plus simple que celle de l'algorithme de fixation d'étiquettes. Généralement, des stratégies de type FIFO ou LIFO sont adoptées pour manipuler la liste Q. Les algorithmes les plus connus qui se basent sur l'utilisation de cette stratégie, sont les algorithmes de Bellman [27], Ford et Fulkerson [28], Moore [29], l'algorithme Esopo-pape proposé par Pape [23]

et l'algorithme de seuil développé par Glover et *al* [30]. Le schéma général de ce type d'algorithme est présenté dans le pseudo-code de la figure 3.3 suivante.

$f(s) \leftarrow 0$;
Pour tout $i \in N - \{s\}$ **faire** $f(i) \leftarrow \infty$;
$Q \leftarrow \{1 = s\}$;
tant que $Q \neq \phi$ **faire**
$v = \text{Selectionne\_sommet}(Q)$;
$Q \leftarrow Q - \{v\}$;
Pour tout $(v, j) \in FS(v)$ **faire**
Si $f(v) + \upsilon_{vj} < f(j)$ **alors**
$f(j) \leftarrow f(v) + \upsilon_{vj}$;
$S(j) \leftarrow$ étiqueté;
$B(j) \leftarrow v$;
fin_si
Si $j \notin Q$ **alors**
$Q \leftarrow Q \cup \{j\}$;
fin_si
fin_pour
fin_tant que

Figure.3.3. Schéma général de l'algorithme correction d'étiquettes.

Les algorithmes de Bellman, Ford et Moore représentent les algorithmes de base qui utilisent des listes FIFO. Le sommet à scanner est sélectionné en tête de la liste Q, les nouveaux sommets sont insérés en fin de liste. Ce type de méthode nécessite au plus N^2 itérations (insertion, sélection) pour étiqueter tous les sommets du graphe. Cependant, la complexité des algorithmes utilisant cette méthode est estimée à $O(|A||N|)$. Cette relation est liée au fait qu'il y a aux plus $N-1$ itérations principales pour consulter tous les successeurs de tous les sommets, c'est-à-dire les $|A|$ arcs [9]. L'algorithme Esopo-Pape [23] modifie la stratégie FIFO en changeant l'opération d'insertion des nouveaux sommets. Ceux insérés pour la première fois dans Q sont insérés en fin, par contre, ceux réinsérés pour la deuxième fois sont insérés en tête de Q. En effet, la réinsertion d'un sommet i dans la liste Q témoigne d'une amélioration de sa valeur $f(i)$. Cette amélioration engendrera certainement une diminution de f sur les sommets

successeurs de i, et il est donc judicieux de le traiter en priorité. La sélection, elle, reste la même: le sommet sélectionné est celui qui se trouve en tête de liste. A l'inverse de la version originale FIFO, pour certains réseaux particuliers, l'algorithme Esopo-Pape a un comportement exponentiel de complexité maximale égale à $O\left(|A|2^{|N|}\right)$ [9]. D'autres versions de cette méthode sont proposées par Pallotino, Gallo et Pallotino [13]. Dans ces versions, la liste Q est divisée en deux listes Q_1 et Q_2. Les sommets sont sélectionnés en tête de Q_1 si elle n'est pas vide, sinon, ils sont retirés en tête de Q_2. Les sommets insérés pour la première fois dans la liste sont insérés en fin de Q_2 sinon ils sont insérés en fin de Q_1. Ces modifications ont permis, d'une part, de garder le concept utilisé par Pape [23] améliorant ainsi la construction du chemin, et d'autre part, d'utiliser la stratégie FIFO, ce qui réduit la complexité de l'algorithme à $O\left(|A||N|^2\right)$ [14]. Dans [30], Glover et *al*., proposent un algorithme similaire à celui de Pallottino [14]. Le concept de division de la liste Q en deux listes Q_1 et Q_2 est conservé. Mais à l'inverse de la méthode proposée par Pallotino, l'opération de sélection est toujours effectuée sur les sommets de la liste Q_1. Si cette liste est vide, les sommets de Q_2 dont la valeur de f est supérieure à une valeur de seuil sont transférés de Q_2 vers Q_1. L'opération d'insertion est effectuée en fonction de la valeur de f. Si cette valeur est supérieure à la valeur du seuil, alors les nouveaux sommets sont insérés en fin de Q_2, sinon ils sont insérés en fin de Q_1. A chaque itération, la valeur du seuil est ajustée en fonction de la plus petite valeur de f dans Q_2. La méthode nécessite N^3 itérations d'insertion, sélection et transfert de Q_2 vers Q_1 et sa complexité est estimée à $O\left(|A||N|^2\right)$. Dans [14], Bertsekas propose une stratégie encore plus précise (*Small label to the front SLF*) basée sur la supposition suivante: *le nombre d'itérations d'une méthode de correction d'étiquettes dépend du rang du sommet dans la liste triée en fonction de f*. Ainsi, pour améliorer les performances de l'algorithme, la stratégie d'insertion doit assurer que les sommets présentant une petite valeur de f se trouvent en tête de liste. Ceci garantit une sorte de couplage entre les principes de la méthode de fixation d'étiquettes et ceux de l'algorithme de seuil. La mise en place de SLF [14]

est réalisée en appliquant la règle suivante: si la valeur de f du nouveau sommet est inférieure à celle du sommet en tête de liste, alors le nouveau sommet est inséré au début de la liste, sinon il est inséré en fin de liste. La complexité de cet algorithme est estimée par l'auteur en $O\left(|N||A|\right)$. Les tests effectués par Bertsekas [14] montrent que cette stratégie est plus rapide que celles présentées par Bellman [27], Ford et Fulkerson [28], Glover et *al* [30], Pape [23].

CHAPITRE 4

ALGORITHMES DE RESOLUTION DU PROBLEME DE CHEMINEMENT MULTICRITERE

4.1. Introduction

La variante *monocritère* du problème de plus court chemin est considérée comme étant pratiquement résolue. Mais, le cas *multicritère* est une variante NP - complète [1] qui fait encore l'objet de travaux de recherche. Généralement, les algorithmes de résolution proposés pour le cas *multicritère* se basent, essentiellement, sur l'utilisation des algorithmes pionniers (chapitre 3) que nous avons présentés ou sur des versions légèrement modifiées. Vincke [4], Brumbaugh-Smith et Shier [16] et Skriver et Andersen [15] proposent des algorithmes de résolution du problème de plus court chemin *multicritère* ou *bicritère* basés sur la méthode de marquage temporaire (correction d'étiquettes). Hansen [31], Martins [11], Hansen et *al* [17] et Gabrel [32] proposent des algorithmes basés sur la méthode de marquage permanent (fixation d'étiquettes). Dans la suite de ce chapitre on s'intéresse aux algorithmes qui se basent sur la méthode de marquage permanent.

Dans la première partie de ce chapitre, nous donnons les algorithmes de résolution du problème de recherche d'un plus court chemin multicritère d'un sommet source s à un sommet puits t. Dans la deuxième partie, nous présentons les algorithmes de résolution du problème de recherche d'un plus court chemin multicritère d'un sommet source s à tout autre sommet du réseau.

4.2. Algorithmes de résolution du problème de recherche d'un plus court chemin multicritère d'un sommet source s à un sommet puits t

Dans le cas bicritère, Hansen [31], est le premier à avoir proposé un algorithme pour résoudre le problème du plus court chemin bicritère. Son algorithme est une généralisation de l'algorithme classique du plus court chemin de Dijkstra [2]. Dans

cet algorithme une ou plusieurs étiquettes sont associées à chaque sommet $i \in N$, l'ensemble d'étiquettes étant divisé en deux sous-ensembles: ensemble d'étiquettes permanentes et ensemble d'étiquettes temporaires. Chaque étiquette associée à un sommet $i \in N$, est une paire (π_i, ξ_i), où $\pi_i = [\upsilon^1(p_{si}), \upsilon^2(p_{si})]$ est la valeur du chemin p_{si} de s à i et ξ_i est le prédécesseur de sommet i dans le chemin p_{si}. Les étiquettes permanentes correspondent à des chemins efficaces, alors que chaque étiquette temporaire correspond à un chemin qui peut être dominé ou pas.

A chaque étape de l'algorithme, on sélectionne l'étiquette temporaire lexicographique petite, associée à un sommet $i \in N$, pour la rendre permanente et elle est utilisée pour associer une nouvelle étiquette temporaire à tous les sommets j pour lesquels $(i, j) \in A$. Après cette étape, le test de dominance est utilisé pour supprimer toutes les étiquettes temporaires correspondant aux chemins dominés.

Martins [11] a proposé deux algorithmes pour la résolution du problème du plus court chemin multicritère. Un des ces algorithmes est une généralisation pour le cas multicritère de l'algorithme présenté par Hansen [31].

4.2.1. Algorithme de Martins

Hypothèse 1:

Il est supposé qu'il existe un entier $k \in \{1, 2..., r\}$ pour lequel $\upsilon_{ij}^k \geq 0$ pour tout $(i, j) \in A$. Sans perte de généralité, il suppose que $k = 1$.

Hypothèse 2:

Pour chaque circuit Γ dans le réseau R = (N, A,υ), il est supposé que pour tout k = 1,…, r, $\upsilon^k(\Gamma) \geq 0$ avec au moins une inégalité stricte.

Théorème 4.1. [11]

Toute paire p et p' de chemins efficaces de s à t est connectée par une suite $\{p, p_1, p_2, ..., p_l, p'\}$ de chemins adjacents efficaces.

4.2.1.1. Approche d'étiquetages multiples

Martins [11] a proposé un algorithme pour déterminer l'ensemble des chemins efficaces lorsque toutes les valuations sont positives ($\upsilon_{ij}^k \geq 0$ pour tout (i, j) ∈ A, et k = 1,…, r) avec au moins une inégalité stricte. L'algorithme de Martins, présenté ci-après (figure 4.1) est une extension multicritères de l'algorithme de Dijkstra dans lequel l'opérateur "min" est remplacé par le *test de dominance*. L'idée de l'algorithme de Martins est assez intuitive. À chaque itération, et pour chaque sommet, deux types de marquages sont utilisés: *marquage permanent* et *marquage temporaire*. L'algorithme sélectionne l'étiquette lexicographiquement petite à partir des étiquettes temporaires, la convertit en une étiquette permanente, et propage l'information contenue dans cette étiquette à toutes les étiquettes temporaires de ses successeurs. La procédure s'arrête lorsqu'il n'existe aucune étiquette temporaire.

Soit un sommet j∈N du réseau R. La *l* ème étiquette associée au sommet j est définie comme suit: $[\upsilon^1(p_{sj}),...,\upsilon^r(p_{sj}), i, h]_l$; où $(\upsilon^1(p_{sj}),...,\upsilon^r(p_{sj}))$ est un vecteur des performances; i est le sommet précédent à partir duquel le sommet j a été marqué; et *h* est la position de cette étiquette dans la liste d'étiquettes sur le sommet i pour lequel:

$$\upsilon_l^k(p_{sj}) = \upsilon_h^k(p_{si}) + \upsilon_{ij}^k \ , \quad pour\ tout\ k \in \{1,...,r\}$$

Où

- $\upsilon_l^k(p_{sj})$ noté aussi $\upsilon_{j,l}^k$ est la *k*éme composante de la *l*éme étiquette associée au sommet j.

- $\upsilon_l^k(p_{sj})$ est la valeur du chemin p_{sj} suivant le critère *k* lorsqu'on suit le chemin de s à j suivant l'étiquette *l*, défini comme le chemin de s à i, avec l'étiquette *h*, suivi l'arc (i, j).

Pour un sommet j, on dit que $[\upsilon^1(p_{sj}),...,\upsilon^r(p_{sj}),*,*]_\alpha$ est *lexicographiquement petite* que $[\upsilon^1(p'_{sj}),...,\upsilon^r(p'_{sj}),*,*]_\beta$ si

$\upsilon^1_\alpha(p_{sj}) = \upsilon^1_\beta(p'_{sj})$,..., $\upsilon^{k-1}_\alpha(p_{sj}) = \upsilon^{k-1}_\beta(p'_{sj})$; $\upsilon^k_\alpha(p_{sj}) < \upsilon^k_\beta(p'_{sj})$ pour un certain k $\in$ {1,…, r}.

Etape 0: Associer l'étiquette temporaire $[0,...,0,*,*]_1$ au sommet s.

Etape1:

(1)- Si l'ensemble des étiquettes temporaires est vide, aller à étape 3.
Sinon, déterminer parmi les étiquettes temporaires lexicographiquement petite. Désignons la par la $h^{ème}$ étiquette associée au sommet i et considérons la comme une étiquette permanente.

(2)- Tant qu'il existe un sommet $j \in N$ tel que $(i, j) \in A$, faire:

(i) $\upsilon^k(p_{sj}) = \upsilon^k_h(p_{si}) + \upsilon^k_{ij}$, *pour tout* $k \in \{1,...,r\}$,
et soit $[\upsilon^1(p_{sj}),...,\upsilon^r(p_{sj}), i, h]_\beta$ la nouvelle étiquette temporaire associée au sommet j.

(ii) supprimer toutes les étiquettes temporaires du sommet j dominées par la nouvelle étiquette,
supprimer la nouvelle étiquette si elle est dominée.

Etape 2: Aller à l'étape 1.

Etape 3: Déterminer les chemins efficaces de s à t.

Etape 4: Fin de l'algorithme.

Figure 4.1. Algorithme1 de Martins.

L'étape 3 de l'algorithme est facilement exécutée. En effet à la fin de l'algorithme, il existe autant des étiquettes permanentes associées au sommet t que le nombre de chemins efficaces de s à t. Cependant chaque étiquette permanente correspondra à un chemin efficace unique.

Pour déterminer tous les chemins efficaces, on choisit une étiquette permanente associée au sommet t et on suppose que (j,h) est l'étiquette qui lui est associée.

Donc j est le sommet juste avant t dans le chemin efficace et le sommet juste avant j est donné par la $h^{éme}$ étiquette. Par, une remontée, le premier sommet s du chemin est retrouvé.

Remarque 4.1.

• Tous les chemins efficaces du sommet s à i, pour tout $i \in N - \{s\}$, peuvent être déterminés par l'algorithme1 de la figure 4.1.

• Si nous appellerons s-efficaces les arcs du graphe G empruntés au moins par un chemin efficaces d'extrémité initiale le sommet s, alors le sous graphe partiel G_s formé des seuls descendants de s et des seuls arcs s-efficaces de G contient l'essentiel d'information concernant les chemins efficaces, puisque tout chemin (de s à $i, \forall i \in N - \{s\}$) efficace dans G est un chemin de G_s, la réciproque est fausse.

Définition 4.1.

Une arborescence lexicographiquement petite est une arborescence pour laquelle chaque chemin lexicographiquement petit du sommet s à i est déterminé pour tout $i \in N - \{s\}$.

Proposition 4.1. [11]

Une arborescence lexicographiquement petite est une arborescence efficace (non-dominée).

• Soit T une arborescence et soit $\Pi_i^k(T)$ (ou simplement Π_i^k) la valeur de $v^k(p_{si})$ où p_{si} est un chemin de s à i dans T.

• Ainsi, $\Pi_s^k(T) = 0$ pour tout $k \in \{1,...,r\}$ et pour toute arborescence T.

• Pour tout $k \in \{1,...,r\}$ et pour tout arc $(i, j) \in A$, on définit le coût réduit $\overline{v}_{ij}^k(T)$ (ou simplement $\overline{v}_{ij}^k$) par : $\overline{v}_{ij}^k = v_{ij}^k + \Pi_i^k - \Pi_j^k$.

• Pour une arborescence T et un arc $(i, j) \notin T$ on a:

$T' = T \cup \{(i, j)\} - \{(h, j)\}$ où $(h, j) \in T$ est une arborescence adjacente à T.

• Dans l'algorithme suivant, un vecteur $V \neq 0$ est dit *lexicographiquement positif* si sa première composante non nulle est positive. On utilise ainsi la notation $V >^L 0$ pour dire que le vecteur V est lexicographiquement positif.

Etape0:

(1) Déterminer une arborescence T lexicographiquement petite.

(2) Calculer $[\Pi_i^1(T),\ldots,\Pi_i^r(T)]$ pour tout $i \in N$.

(3) Placer T dans LIST.

Etape1: Si LIST est vide alors fin d'algorithme.

Sinon supprimer de la LIST l'arborescence T' avec

$[\Pi_t^1(T'),\ldots,\Pi_t^r(T')]$ lexicographiquement petite.

Calculer $[\overline{\upsilon}_{ij}^1(T'),\ldots,\overline{\upsilon}_{ij}^r(T')]$ pour tout $(i,j) \notin T'$.

Etape2: Si p' (le chemin de s à t dans T') n'est pas dans LIST1, alors placer le dans LIST1.

Etape3: Tant qu'il existe un arc $(i,j) \notin T'$ tel que l'une des deux conditions suivantes est satisfaite:

$[\overline{\upsilon}_{ij}^1(T'),\ldots,\overline{\upsilon}_{ij}^r(T')] >^L 0$ et $\overline{\upsilon}_{ij}^k < 0$ pour certain $k \in \{1,\ldots,r\}$

ou $\overline{\upsilon}_{ij}^k = 0$ pour tout $k \in \{1,\ldots,r\}$ faire:

(1) $T = T' \cup \{(i,j)\} - \{(h,j)\}$ où $(h,j) \in T'$

(2) Calculer $[\Pi_i^1(T),\ldots,\Pi_i^r(T)]$ pour tout $i \in N$.

(3) Placer T dans LIST si elle n'existe pas.

Etape4: Supprimer de LIST toute arborescence dominée.

Etape5: Aller à l'étape 1.

Figure 4.2. Algorithme2 de Martins.

• La condition imposée à l'étape 3 est une condition nécessaire pour qu'une arborescence adjacente à une arborescence efficace est efficace.

• Or certaines arborescences générées à l'étape 3 peuvent être dominées, ce qui explique l'existence de l'étape 4.

Remarque 4.2.

• Le principal problème de cet algorithme est l'exécution de l'étape 4, malgré que le test de dominance peut être exécuté seulement pour les branches représentant les chemins du sommet s à t.

• L'idéal est de trouver une condition suffisante pour déterminer les arborescences efficaces adjacentes à une arborescence efficace.

4.2.2. Méthode d'étiquetage à partir de deux extrémités du réseau

Pour le problème de plus court chemin bicritère Hansen et *al.* [17] ont proposé un algorithme qui exploite l'information provenant des deux extrémités du réseau. L'algorithme prolonge des sous-chemins efficaces à partir de la source et puits, en utilisant des extensions non-dominées déjà calculées pour faire la dominance.

- Soit p_{ij} un chemin de i et j, la valeur du chemin p_{ij} est notée par l'étiquette $[\upsilon^1(p_{ij}), \upsilon^2(p_{ij})]$.
- L'étiquette lexicographiquement petite correspond à un plus court chemin p_{ij} de i à j pour le $r^{\text{ème}}$ critère $(r = 1, 2)$, est notée par $[\upsilon^1_{(r)}(p_{ij}), \upsilon^2_{(r)}(p_{ij})]$.
- Un chemin correspondant à une étiquette efficace représente un chemin efficace, et vice versa.
- Pour chaque sommet $i \in N$ on associe quatre ensembles L^T_{si}, L^T_{it}, L^P_{si} et L^P_{it} des étiquettes efficaces. Les ensembles L^T_{si} et L^T_{it} appelés ensembles des étiquettes efficaces temporaires du chemin p_{si} de s à i et du chemin p_{it} de i à t. L^P_{si} et L^P_{it} appelées ensembles des étiquettes efficaces permanentes du chemin p_{si} de s à i et du chemin p_{it} de i à t.
- L'ensemble L^T_{st} (resp. L^P_{st}) appelé ensemble des étiquettes efficaces temporaires du chemin p_{st} de s à t (resp. appelé ensembles des étiquettes efficaces permanentes du chemin p_{st} de s à t).

- L'étiquette lexicographiquement petite $[\upsilon^1_{(1)}(p_{ij}), \upsilon^2_{(1)}(p_{ij})]$ correspond à un plus court chemin de i et j pour le critère υ^1 et $[\upsilon^1_{(2)}(p_{ij}), \upsilon^2_{(2)}(p_{ij})]$ correspond à un plus court chemin de i et j pour le critère υ^2.
- A chaque itération de l'algorithme, la $k^{ème}$ étiquette associé à l'ensemble des étiquettes efficaces permanentes du chemin p_{ij} de i à j en augmentant les valeurs de critère υ^1, sera noté $[\upsilon^1_k(p_{ij}), \upsilon^2_k(p_{ij})]$.
- Pour un sommet $i \in N$, un sous-chemin p_{si} ou p_{it} qui ne peut pas être prolongé à un chemin efficace p_{st} de s à t, est appelé sous-chemin dominé.

Proposition 4.2. [17]

Considérons un sommet $\bar{i}$ correspondant à une étiquette lexicographiquement petite dans $\bigcup_{i \in N} L^T_{si}$ et soit $[\upsilon^1(p_{\bar{j}t}), \upsilon^2(p_{\bar{j}t})]$ la dernière étiquette sélectionnée à partir de $\bigcup_{i \in N} L^T_{it}$ pour l'extension. Si $\upsilon^1(p_{\bar{j}t}) \geq \upsilon^1_{(2)}(p_{\bar{i}t})$, où $\upsilon^1_{(2)}(p_{\bar{i}t})$ est la valeur de critère υ^1 du chemin lexicographiquement petit $p_{\bar{i}t}$ pour le critère υ^2, alors toutes les étiquettes efficaces $[\upsilon^1(p_{st}), \upsilon^2(p_{st})]$ du chemin p_{st} de s à t obtenues par l'extension d'un chemin correspondant à $[\upsilon^1(p_{s\bar{i}}), \upsilon^2(p_{s\bar{i}})]$ sont examinées.

Proposition 4.3. [17]

Considérons une étiquette lexicographiquement petite $[\upsilon^1(p_{s\bar{i}}), \upsilon^2(p_{s\bar{i}})]$ dans $\bigcup_{i \in N} L^T_{si}$ et soit $[\upsilon^1_k(p_{\bar{i}t}), \upsilon^2_k(p_{\bar{i}t})]$, pour $k = 1, 2, ..., \left|L^P_{\bar{i}t}\right|$, est la $k^{éme}$ étiquette associée à $L^P_{\bar{i}t}$. Si toutes les étiquettes $[\upsilon^1(p_{s\bar{i}}), \upsilon^2(p_{s\bar{i}})] + [\upsilon^1_k(p_{\bar{i}t}), \upsilon^2_{k+1}(p_{\bar{i}t})]$, pour $k = 1, 2, ..., \left|L^P_{\bar{i}t}\right| - 1$ sont dominées par rapport à $L^T_{st} \bigcup L^P_{st}$, alors toutes les étiquettes efficaces $[\upsilon^1(p_{st}), \upsilon^2(p_{st})]$ du chemin p_{st} de s à t obtenues par l'extension d'un chemin correspondant à $[\upsilon^1(p_{s\bar{i}}), \upsilon^2(p_{s\bar{i}})]$ sont examinées.

Remarque 4.3.

Les propositions 1 et 2 restent valables si on considère l'étiquette lexicographiquement petite $[\upsilon^1(p_{\bar{j}t}), \upsilon^2(p_{\bar{j}t})]$ dans $\bigcup_{i \in N} L^T_{\bar{j}t}$.

Théorème 4.2. [17]

Soit $[v^1(p'), v^2(p')]$ et $[v^1(p''), v^2(p'')]$ deux étiquettes efficaces extrêmes avec $v^1(p') < v^1(p'')$ et soit $[v^1(p), v^2(p)]$ l'étiquette optimale du problème de plus court chemin paramétrique pour $\alpha = (v^2(p') - v^2(p'')) / (v^1(p'') - v^1(p') + v^2(p') - v^2(p''))$. Alors $[v^1(p), v^2(p)]$ est une étiquette efficace extrême et $v^1(p') \leq v^1(p) \leq v^1(p'')$. De plus, si $\alpha v^1(p) + (1-\alpha)v^2(p) = \alpha v^1(p') + (1-\alpha)v^2(p')$, aucune étiquette efficace ne se trouve entre $[v^1(p'), v^2(p')]$ et $[v^1(p), v^2(p)]$ ni entre $[v^1(p), v^2(p)]$ et $[v^1(p''), v^2(p'')]$.

4.2.2.1 Algorithme

Hansen et *al*. [17] proposent une approche à double parcours basée sur la méthode de fixation d'étiquettes pour résoudre le problème de plus court chemin *bicritère*. Les différences majeures de cette approche avec celle de Dijkstra sont principalement l'utilisation d'un parcours à double sens du réseau et la sélection des sommets à traiter en se basant sur l'ordre lexicographique [25]. Plus formellement, dans le cas de Hansen les arcs de R sont évalués par deux paramètres entiers v^1 et v^2 liés aux deux critères à minimiser. De ce fait, chaque chemin p est évalué par deux valeurs $v^1(p)$ et $v^2(p)$, chacune associée à un critère donné. L'ordre lexicographique utilisé par la méthode consiste à définir un ordre de préférence entre les critères. Pendant la sélection, l'algorithme choisit le sommet qui possède la plus petite valeur de critère dont la préférence est la plus grande. Si deux sommets possèdent la même valeur, l'algorithme utilise la longueur du deuxième objectif pour les départager.

Chaque sommet $i \in N$ possède quatre ensembles $L_{si}^T, L_{it}^T, L_{si}^P$ et L_{it}^P qui mémorisent les étiquettes efficaces.

Les deux premiers ensembles mémorisent les étiquettes efficaces temporaires en cours de construction. L_{si}^P et L_{it}^P mémorisent les étiquettes efficaces permanentes. Elles sont initialisées par les valeurs des chemins liant les sommets s et t au sommet $i \in N$. Ces chemins sont construits :

- soit en minimisant le critère v^1 et en évaluant v^2, puis en minimisant le critère v^2 et en évaluant v^1 .
- soit en minimisant l'agrégation des deux critères dans une fonction $\alpha v^1 + (1-\alpha) v^2$ pour des valeurs de α spécifiées (théorème 4.2).

Les ensembles $L_{ss}^T, L_{tt}^T, L_{ss}^P$ et L_{tt}^P sont initialisées par des étiquettes correspondant aux circuits (s,s) et (t,t) minimisant le critère v^1. L'ajout et la suppression de nouvelles étiquettes aux ensembles $L_{si}^T, L_{it}^T, L_{si}^P$ et L_{it}^P sont effectués en utilisant le concept de Pareto dominance: l'étiquette $[v^1(p), v^2(p)]$ domine $[v^1(p'), v^2(p')]$ si $v^1(p) \leq v^1(p')$ et $v^2(p) \leq v^2(p')$ avec au moins une inégalité stricte (pour plus de détails voir chapitre 1). Toutes les étiquettes dominées ainsi que toutes les étiquettes rendantes sont supprimées des ensembles L_{si}^T, L_{it}^T, L_{si}^P et L_{it}^P. Après cette phase d'initialisation, l'algorithme procède à la construction des chemins *bicritère* utilisant l'algorithme de fixation d'étiquettes. Pour chaque sommet $i \in N$, l'algorithme :

- sélectionne l'étiquette lexicographiquement petite $[v^1(p), v^2(p)]$ de l'ensemble $\bigcup_{i \in N} L_{si}^T$ (respectivement de $\bigcup_{i \in N} L_{it}^T$) ;
- transfère $[v^1(p), v^2(p)]$ de $\bigcup_{i \in N} L_{si}^T$ à L_{si}^P (respectivement de $\bigcup_{i \in N} L_{it}^T$ à L_{it}^P) ;
- met à jour les étiquettes des successeurs (respectivement prédécesseurs) de i ;
- insère les nouvelles étiquettes dans les ensembles L_{si}^T et L_{si}^P du sommet successeur (respectivement dans les ensembles L_{it}^T et L_{it}^P du sommet prédécesseur) en vérifiant la condition de dominance.

Ces opérations sont répétées tant que les ensembles $\bigcup_{i \in N} L_{si}^T$ et $\bigcup_{i \in N} L_{it}^T$ ne sont pas vides. Sachant que la taille maximale de ces ensembles est égale à $n^2 q$, où q est la plus grande valeur que peuvent avoir v^1 ou v^2, $O(n^2 q)$ opérations sont nécessaires pour que tous les sommets des listes $\bigcup_{i \in N} L_{si}^T$ et $\bigcup_{i \in N} L_{it}^T$ soient

traités. La sélection du meilleur sommet nécessite $O(nq)$ opérations de comparaison. Finalement, la mise à jour des étiquettes et des listes requiert $O(nq\log(nq))$ opérations. De ce fait, les auteurs estiment la complexité de cet algorithme en $O(n^4q^3\log(nq))$ où $n=|N|$.

Algorithme

1. Initialisation
   (a) Exécuter **Initialisation de base** ou **Initialisation d'extension**
   (b) Pour $i\in N-\{s,t\}$, $[v^1(p),v^2(p)]\in L_{si}^P$ et $[v^1(p'),v^2(p')]\in L_{it}^P$, supprimer de L_{st}^T toutes les étiquettes dominées par $[v^1(p),v^2(p)]+[v^1(p'),v^2(p')]$ et ajouter la nouvelle étiquette à L_{st}^T si elle est efficace par rapport à L_{st}^T et L_{st}^P.
2. Etape principale

Tant que $\bigcup_{i\in N} L_{si}^T\neq\phi$ et $\bigcup_{i\in N} L_{it}^T\neq\phi$:

Si $\left|\bigcup_{i\in N} L_{si}^T\right|\leq\left|\bigcup_{i\in N} L_{it}^T\right|$, aller à l'étape **en avant**, si non aller à l'étape **en arrière**.

Initialisation de base

1. plus court chemin avec un seul critère

Calculer les étiquettes

$[v_{(1)}^1(p_{si}),v_{(1)}^2(p_{si})],[v_{(1)}^1(p_{it}),v_{(1)}^2(p_{it})],[v_{(2)}^1(p_{si}),v_{(2)}^2(p_{si})],[v_{(2)}^1(p_{it}),v_{(2)}^2(p_{it})]$

pour $i\in N$.

2. Initialisation les ensembles des étiquettes

Soit : $v^1(p_{s\bar{i}})=0$ et $L_{ss}^P=L_{ss}^T=\left\{[v_{(1)}^1(p_{ss}),v_{(1)}^2(p_{ss})]\right\}$.

Pour $i\in N-\{s\}$,ensemble: $L_{si}^T=\phi$ et $L_{si}^P=\left\{[v_{(1)}^1(p_{si}),v_{(1)}^2(p_{si})],[v_{(2)}^1(p_{si}),v_{(2)}^2(p_{si})]\right\}$.

Soit $v^1(p_{\bar{j}t})=0$ et $L_{tt}^P=\ L_{tt}^T=\left\{[v_{(1)}^1(p_{tt}),v_{(1)}^2(p_{tt})]\ \right\}$.

Pour $j\in N-\{s,t\}$,ensemble: $L_{jt}^T=\phi$ et $L_{jt}^P=\left\{[v_{(1)}^1(p_{jt}),v_{(1)}^2(p_{jt})],[v_{(2)}^1(p_{jt}),v_{(2)}^2(p_{jt})]\right\}$.

Initialisation d'extension

1. Exécuter **initialisation de base** et soit
$C = \left\{ [v^1_{(1)}(p_{st}), v^2_{(1)}(p_{st})], [v^1_{(2)}(p_{st}), v^2_{(2)}(p_{st})] \right\}$.

2. tant que $C \neq \phi$:

(a) sélectionner et supprimer un élément $([v^1(p'), v^2(p')], [v^1(p''), v^2(p'')])$ de C Calculer $\alpha = (v^2(p') - v^2(p'')) / (v^1(p'') - v^1(p') + v^2(p') - v^2(p''))$ et l'ensemble des coûts des arcs $\alpha v^1_{ij} + (1-\alpha) v^2_{ij}$, pour $(i, j) \in A$.

(b) pour $i \in N$, calculer et ajouter l'étiquette $[v^1(p_{si}), v^2(p_{si})]$ de plus court chemin p_{si} à L^P_{si} et l'étiquette $[v^1(p_{it}), v^2(p_{it})]$ de plus court chemin p_{it} à L^P_{it}. Soit $[v^1(p), v^2(p)]$ est l'étiquette de plus court chemin de s à t.

(c) Si $\alpha v^1(p) + (1-\alpha) v^2(p) < \alpha v^1(p') + (1-\alpha) v^2(p')$ ajouter $([v^1(p'), v^2(p')], [v^1(p), v^2(p)])$ et $([v^1(p), v^2(p)], [v^1(p''), v^2(p'')])$ à C.

Etape en avant

1. Sélectionner l'étiquette lexicographiquement petite $[v^1(p_{s\bar{i}}), v^2(p_{s\bar{i}})]$

Transférer l'étiquette $[v^1(p_{s\bar{i}}), v^2(p_{s\bar{i}})]$ de $\cup_{i \in N} L^T_{si}$ à $L^P_{s\bar{i}}$ et supprimer elle de $L^T_{s\bar{i}}$.

2. Extension de chemin $p_{s\bar{i}}$ sélectionné

(a) pour $[v^1(p), v^2(p)] \in L^P_{\bar{i}t}$, supprimer de L^T_{st} toutes les étiquettes dominées par $[v^1(p_{s\bar{i}}), v^2(p_{s\bar{i}})] + [v^1(p), v^2(p)]$ et ajouter cette nouvelle étiquette à L^T_{st} si elle est efficace par rapport à L^T_{st} et L^P_{st}.

(b) si $v^1(p_{s\bar{j}}) < v^1_{(2)}(p_{s\bar{i}})$ et au moins une étiquette $[v^1(p_{s\bar{i}}), v^2(p_{s\bar{i}})] + [v^1_k(p_{s\bar{i}}), v^2_{k+1}(p_{s\bar{i}})]$ pour $k = 1, 2..., \left| L^P_{\bar{i}t} \right| - 1$, est efficace par rapport à L^T_{st} et L^P_{st}, alors, pour $(\bar{i}, j) \in A$: supprimer de L^T_{sj} toutes les étiquettes dominées par $[v^1(p_{s\bar{i}}), v^2(p_{s\bar{i}})] + (v^1_{\bar{i}j}, v^2_{\bar{i}j})$ et ajouter cette nouvelle étiquette à L^T_{sj} si elle est efficace par rapport à L^T_{sj} et L^P_{sj}.

Etape en arrière

1. Sélectionner l'étiquette lexicographiquement petite $[v^1(p_{\bar{j}t}), v^2(p_{\bar{j}t})]$

   Transférer l'étiquette $[v^1(p_{\bar{j}t}), v^2(p_{\bar{j}t})]$ de $\bigcup_{i\in N} L_{it}^T$ à $L_{\bar{j}t}^P$ et supprimer elle de $L_{\bar{j}t}^T$

2. Extension de chemin $p_{\bar{j}t}$ sélectionné

(a) pour $[v^1(p), v^2(p)] \in L_{s\bar{j}}^P$, supprimer de L_{st}^T toutes les étiquettes dominées par $[v^1(p), v^2(p)] + [v^1(p_{\bar{j}t}), v^2(p_{\bar{j}t})]$ et ajouter cette nouvelle étiquette à L_{st}^T si elle est efficace par rapport à L_{st}^T et L_{st}^P.

(b) si $v^1(p_{s\bar{i}}) < v_{(2)}^1(p_{s\bar{j}})$ et au moins une étiquette $[v_k^1(p_{s\bar{j}}), v_{k+1}^2(p_{s\bar{j}})] + [v^1(p_{\bar{j}t}), v^2(p_{\bar{j}t})]$ pour $k = 1, 2, ..., \left|L_{s\bar{j}}^P\right| - 1$, est efficace par rapport à L_{st}^T et L_{st}^P, alors, pour $(i, \bar{j}) \in A$: supprimer de L_{it}^T toutes les étiquettes dominées par $(v_{i\bar{j}}^1, v_{i\bar{j}}^2) + [v^1(p_{\bar{j}t}), v^2(p_{\bar{j}t})]$ et ajouter cette nouvelle étiquette à L_{it}^T si elle est efficace par rapport à L_{it}^T et L_{it}^P.

Figure 4.3. Algorithme d'étiquetages à double parcours.

4.3. Algorithmes de résolution du problème de recherche d'un plus court chemin multicritère d'un sommet source s à tous les autres sommets du réseau

4.3.1 Algorithme de Martins modifié

D'après la remarque 4.1, l'algorithme1 de la figure 4.1 donné au paragraphe précédent peut être utilisé pour la résolution de ce problème. Tandis que l'algorithme2 de la figure 4.2 ne peut être utilisé qu'après de légères modifications, telles qu'elles ont été faites par Abbas et Khodja [33], l'algorithme2 sera noté algorithme3 de la figure 4.4.

Etape0:

(1) Déterminer une arborescence T lexicographiquement petite.

(2) Calculer $[\Pi_i^1(T),\ldots,\Pi_i^r(T)]$ pour tout $i \in N$.

(3) Placer T dans LIST.

Etape1: Si LIST est vide alors fin d'algorithme.

Sinon supprimer de la LIST l'arborescence lexicographiquement petite T'.

Calculer $[\overline{v}_{ij}^1(T'),\ldots,\overline{v}_{ij}^r(T')]$ pour tout $(i,j) \notin T'$.

Etape2: Si T' n'est pas dans LIST1, alors placer T' dans LIST1.

Etape3: Tant qu'il existe un arc $(i,j) \notin T'$ tel que l'une des deux conditions suivantes est satisfaite:

$[\overline{v}_{ij}^1(T'),\ldots,\overline{v}_{ij}^r(T')] >^L 0$ et $\overline{v}_{ij}^k < 0$ pour certain $k \in \{1,\ldots,r\}$

ou $\overline{v}_{ij}^k = 0$ pour tout $k \in \{1,\ldots,r\}$ faire:

(1) $T' = T \cup \{(i,j)\} - \{(h,j)\}$ où $(h,j) \in T'$

(2) Calculer $[\Pi_i^1(T),\ldots,\Pi_i^r(T)]$ pour tout $i \in N$.

(3) Placer T dans LIST si elle n'existe pas.

Etape4: Supprimer de LIST toute arborescence dominée.

Etape5: Aller à l'étape 1.

Figure 4.4. Algorithme3 de Martins modifié.

4.3.2. Généralisation de l'algorithme de Dijkstra au contexte bicritère

Pour déterminer l'ensemble des chemins efficaces d'origine s dans *R,* Gabrel [32] a proposé un algorithme qui généralise au contexte bicritère celui de Dijkstra. Le principe de cet algorithme est le suivant : à chaque sommet $h \in N$ on associe un ensemble d'étiquettes, noté $L(h)$. Chaque étiquette l de $L(h)$ décrit de façon classique un chemin non-dominé allant de s à *h* et, se trouve dans l'un des deux états suivants : soit il s'agit d'une étiquette permanente car l représente un

chemin dont on a démontré qu'il s'agit d'un chemin efficace (on note $L^P(h)$ le sous-ensemble des étiquettes permanentes de $L(h)$), soit il s'agit d'une étiquette temporaire car l représente un chemin dont l'efficacité reste à prouver (on note $L^T(h)$ le sous-ensemble des étiquettes temporaires de $L(h)$). A chaque itération de l'algorithme, on s'attache d'une part, à sélectionner une étiquette temporaire pour la rendre définitive et, d'autre part, à la prolonger vers ses successeurs pour énumérer des nouveaux chemins représentés par des étiquettes temporaires.

Le critère choisit pour sélectionner une étiquette temporaire représentant un chemin efficace utilise l'ordre lexicographique.

Définition 4.2.

L'étiquette l représentant le chemin p est inférieure à l'étiquette l' représentant le chemin p' dans l'ordre (υ^1,υ^2)-*lexicographique*, noté $\upsilon(p) \prec_{\upsilon^1,\upsilon^2} \upsilon(p')$, si et seulement si $\upsilon^1(p) < \upsilon^1(p')$ ou, $\upsilon^1(p) = \upsilon^1(p')$ et $\upsilon^2(p) < \upsilon^2(p')$.

Définition 4.3.

L'étiquette l représentant le chemin p est inférieure à l'étiquette l' représentant le chemin p' dans l'ordre (υ^2,υ^1)-*lexicographique*, noté $\upsilon(p) \prec_{\upsilon^2,\upsilon^1} \upsilon(p')$, si et seulement si $\upsilon^2(p) < \upsilon^2(p')$ ou, $\upsilon^2(p) = \upsilon^2(p')$ et $\upsilon^1(p) < \upsilon^1(p')$.

Ainsi, chaque itération de l'algorithme d'étiquetage se décompose en deux étapes :

Etape de sélection. Sélectionner, dans $\bigcup_{i\in N} L^T(i)$ l'étiquette $l^*_{\upsilon^1,\upsilon^2}$ minimale dans l'ordre (υ^1,υ^2)-*lexicographique*, (on pose $l^*_{\upsilon^1,\upsilon^2} \in L^T(h^*_{\upsilon^1,\upsilon^2})$) ainsi que, l'étiquette $l^*_{\upsilon^2,\upsilon^1}$ minimale dans l'ordre (υ^2,υ^1)-*lexicographique* (on pose $l^*_{\upsilon^2,\upsilon^1} \in L^T(h^*_{\upsilon^2,\upsilon^1})$). Ajouter $l^*_{\upsilon^1,\upsilon^2}$ et $l^*_{\upsilon^2,\upsilon^1}$ à $L^P(h^*_{\upsilon^1,\upsilon^2})$ et $L^P(h^*_{\upsilon^2,\upsilon^1})$ respectivement et, supprimer $l^*_{\upsilon^1,\upsilon^2}$ et $l^*_{\upsilon^2,\upsilon^1}$ de $L^T(h^*_{\upsilon^1,\upsilon^2})$ et $L^T(h^*_{\upsilon^2,\upsilon^1})$ respectivement ;

Etape de génération. Considérer chacune des étiquettes l choisie en phase de sélection, avec $l \in L(h)$. Pour chaque successeur *j* de *h* dans *G*, générer une nouvelle marque l' représentant un chemin p' allant de s à j constitué du chemin *p*, représenté par l, suivi de l'arc (*h*, *j*), de valeur $\upsilon(p') = \upsilon(p) + \upsilon_{hj}$. Inclure l' dans $L^T(j)$ si $\upsilon(p')$ n'est pas dominé par un chemin de $L^T(j)$ et, supprimer de $L^T(j)$ toute étiquette représentant un chemin dominé par p'.

A l'initialisation, une seule étiquette temporaire incluse dans $L^T(s)$ de valeur (0,0) est créée. L'algorithme prend fin lorsque $\bigcup_{i \in N} L^T(i)$ est égal à l'ensemble vide.

Comme les valeurs sur les arcs sont positives, les chemins efficaces arrivant en un sommet quelconque *h* de N sont découverts dans l'ordre croissant de υ^1 et décroissant de υ^2 avec l'ordre (υ^1, υ^2)-*lexicographique* et, dans l'ordre décroissant de υ^1 et croissant de υ^2 avec l'ordre (υ^2, υ^1)-*lexicographique*. A chaque itération, pour tout $p \in P_{s,h^*_{\upsilon^1,\upsilon^2}}$, on a:

- $\upsilon^1(p^*_{\upsilon^1,\upsilon^2}) \geq \upsilon^1(p)$ et $\upsilon^2(p^*_{\upsilon^1,\upsilon^2}) \leq \upsilon(p)$ si p est un chemin représenté dans $L^P(h^*_{\upsilon^1,\upsilon^2})$.
- $\upsilon^1(p^*_{\upsilon^1,\upsilon^2}) \leq \upsilon^1(p)$ et $\upsilon^2(p^*_{\upsilon^1,\upsilon^2}) \geq \upsilon(p)$ si p est un chemin qui n'est pas représenté dans $L^P(h^*_{\upsilon^1,\upsilon^2})$.

en conséquence de quoi, $p^*_{\upsilon^1,\upsilon^2}$ ne peut pas être dominé par p, pour tout $p \in P_{s,h^*_{\upsilon^1,\upsilon^2}}$ avec $\upsilon(p^*_{\upsilon^1,\upsilon^2}) \neq \upsilon(p)$, et $p^*_{\upsilon^1,\upsilon^2}$ est donc bien efficace.

De façon similaire, pour tout $p \in P_{s,h^*_{\upsilon^2,\upsilon^1}}$, on a :

- $\upsilon^2(p^*_{\upsilon^2,\upsilon^1}) \geq \upsilon^2(p)$ et $\upsilon^1(p^*_{\upsilon^2,\upsilon^1}) \leq \upsilon^1(p)$ si p est un chemin représenté dans $L^P(h^*_{\upsilon^2,\upsilon^1})$.

- $v^2(p^*_{v^2,v^1}) \leq v^2(p)$ et $v^1(p^*_{v^2,v^1}) \geq v^1(p)$ si p est un chemin qui n'est pas représenté dans $L^P(h^*_{v^2,v^1})$.

en conséquence de quoi, $p^*_{v^2,v^1}$ ne peut pas être dominé par p, pour tout $p \in P_{s,h^*_{v^2,v^1}}$ avec $v(p^*_{v^2,v^1}) \neq v(p)$, et $p^*_{v^2,v^1}$ est donc bien efficace.

En outre, durant l'étape de génération, les tests de dominance engendrés par l'insertion d'une nouvelle étiquette peuvent être limités car ils ne doivent concerner que les étiquettes dont les valeurs appartiennent à $[v^1(p^*_{v^1,v^2}), v^1(p^*_{v^2,v^1})]$ et $[v^2(p^*_{v^2,v^1}), v^2(p^*_{v^1,v^2})]$.

CHAPITRE 5
REVISION DE L'ALGORITHME DE MARTINS POUR LES PROBLEMES DE PLUS COURT CHEMIN MULTICRITERE AVEC FONCTION COUT MAX-MIN

5.1. Introduction

Deux types de fonctions critères sont considérés dans les problèmes de plus court chemin multicritère: *fonctions linéaires* (de type *somme*) et *fonction du type max-min*, notés respectivement par (S-type) et (M-type). Par exemple, le coût est une fonction critère du type S-type et la qualité de transport public est une fonction du type M-type. Dans le cas général, la notation (σ-S$|\mu$-M) signifie que le problème concerne respectivement σ critères du premier type et μ critères du second type.

Il existe deux classes d'algorithmes pour la résolution des problèmes de plus court chemin multicritère du type S-type: les algorithmes de marquage (d'étiquetage) et les algorithmes de rangement des chemins. Lorsque toutes les fonctions critères sont du type S-type, l'ensemble des chemins efficaces peut être déterminé en utilisant l'algorithme de Martins [11].

Dans la suite de ce chapitre, nous commençons par donner des définitions (section 5.2). Par la suite, nous présentons une version révisée de l'algorithme de Martins pour les problèmes de plus court chemin multicritère (σ-S$|$1-M) (section 5.3). En suite, nous proposons une version révisée de l'algorithme de Martins pour les problèmes de plus court chemin multicritère (σ-S$|$2-M) (section 5.4).

5.2. Définitions

Nous considérons un réseau orienté multicritère R= (N, A,υ), avec N = {1,…, n}, dans lequel chaque arc a = (i, j) est valué par le vecteur $\upsilon_{ij} = (\upsilon_{ij}^1, ..., \upsilon_{ij}^r)$.

Soit p un chemin de s à i dans R. L'espace de décision est désigné par $P_{s,i}$, l'ensemble de tous les chemins de s à tous les autres sommets $i \in N - \{s\}$ dans R. Notons par $v^k(p)$ la valeur d'un chemin p par rapport à un critère $k, k = 1,...,r$, avec $r = \sigma + \mu,$ le nombre des critères. Le vecteur $v(p) = (v^1(p),...,v^r(p))$ représente le vecteur des performances du chemin $p \in P_{s,i}$ dans l'espace des critères $v(P_{s,i})$. La formulation des critères est $v^k(p) = \sum_{(i,j)\in p} v_{ij}^k$ pour les fonctions du S-type, et $v^k(p) = \min_{(i,j)\in p} v_{ij}^k$ pour les fonctions du type M-type. Ils correspondent respectivement au problème $(1-S)$ $\min \sum_{(i,j)\in p} v_{ij}^k$ et au problème $(1-M)$ $\max \min_{(i,j)\in p} v_{ij}^k$.

Soient p et p' deux chemins, $v(p)$ et $v(p')$ leur vecteurs performances. Nous supposons que tous les critères sont à minimiser. Nous introduisons dans la suite quelques définitions utiles.

Définition 5.1.

On dit que $v(p)$ *domine strictement* $v(p')$ si et seulement si, pour tout k = 1,…, r, $v^k(p) < v^k(p')$

Définition 5.2.

Un vecteur de performances $v(p)$, avec $p \in P_{s,i}$, est dit *faiblement non-dominé* s'il n'existe pas un autre chemin $p' \in P_{s,i}$ tel que $v(p')$ domine strictement $v(p)$.

Définition 5.3.

Un chemin $p \in P_{s,i}$ est un chemin *faiblement efficace* si et seulement si son vecteur de performances $v(p)$ est faiblement non-dominé.

5.2.1. Ensemble minimal / Ensemble maximal complet: des chemins efficaces

L'image de l'ensemble des chemins efficaces se réfère à l'espace des critères. Une solution appartient à cet ensemble si elle n'est dominée par aucune autre solution réalisable. La version du problème de plus court chemin multicritère considéré ici est la suivante:

$$\min_{p \in P_{s,i}} (\upsilon^1(p), ..., \upsilon^r(p)), \ \forall \ i \in N - \{s\}$$

Notons par $E(P_{s,i})$ l'ensemble maximal complet des chemins efficaces dans $P_{s,i}$ pour un sommet origine s, i.e., l'ensemble de tous les chemins de $P_{s,i}$ dans lequel chaque chemin correspond à un point non-dominé. Plusieurs chemins efficaces p_1, p_2, p_3 peuvent correspondre au même point non-dominé $\upsilon(p_1) = \upsilon(p_2) = \upsilon(p_3)$ dans l'espace des critères. Dans ce cas, les chemins p_1, p_2, p_3 sont dits équivalents dans l'espace des critères. L'ensemble réduit (ensemble minimal complet) des chemins efficaces est un sous ensemble de $E(P_{s,i})$ dont toutes les solutions sont non équivalentes, et pour chaque chemin $p \in E(P_{s,i})$, il existe un chemin p' dans l'ensemble réduit tel que p et p' sont équivalents.

5.2.2. Extension de l'algorithme de Martins pour le problème de plus court chemin multicritère (σ-S | μ-M)

D'après la remarque 4.1 du chapitre 4, l'algorithme de Martins peut être utilisé pour la résolution du problème de plus court chemin de s à tout autre sommet du réseau.

L'algorithme de Martins (figure 5.1) détermine l'ensemble maximal complet des chemins efficaces de s à tout autre sommet $i \in N - \{s\}$ et utilise plusieurs critères S-type. Ces caractéristiques ont permit d'étendre cet algorithme pour optimiser simultanément les critères S-type et M-type [34].

Etape 0: Associer l'étiquette temporaire $[0,...,0,*,*]_1$ au sommet s.

Etape1:

(1) Si l'ensemble des étiquettes temporaires est vide, aller à étape 3.
Sinon, déterminer parmi les étiquettes temporaires
lexicographiquement petite. Désignons la par la $h^{ème}$ étiquette
associée au sommet i et considérons la comme une étiquette
permanente.

(2) Tant qu'il existe un sommet j∈ N tel que (i, j) ∈ A, faire:

(i) $v^k(p_{sj}) = v_h^k(p_{si}) + v_{ij}^k$, *pour tout* $k \in \{1,...,r\}$,

et soit $[v^1(p_{sj}),...,v^r(p_{sj}), i, h]_\beta$ la nouvelle étiquette

temporaire associée au sommet j.

(ii) supprimer toutes les étiquettes temporaires du sommet j
dominées par la nouvelle étiquette,
supprimer la nouvelle étiquette si elle est dominée.

Etape 2: Aller à l'étape 1.

Etape 3: Trouver les chemins efficaces de s à tout autre sommet $i \in N - \{s\}$.

Etape 4: Fin de l'algorithme.

Figure 5.1. Algorithme de Martins

Chaque étiquette permanente correspondra à un chemin efficace unique. Pour déterminer tous les chemins efficaces, on choisit une étiquette permanente associée au sommet j et on suppose que (i,h) est l'étiquette qui lui est associée. Donc i est le sommet juste avant j dans le chemin efficace et le sommet juste avant i est donné par la $h^{éme}$ étiquette. Alors, par une remontée, le premier sommet s du chemin est retrouvé.

Exemple 5.1.

L'exemple suivant illustre l'algorithme de la figure 5.1 On examine le réseau de la figure 5.2 dans le but de chercher tous les chemins efficaces pour le problème

(4 - S) du sommet s =1 aux sommets destinations i = {2, 3, 4}, avec tous les critères sont à minimiser:

1. l'étiquette temporaire [0,0,0,0,∗,∗] est associée au sommet 1. Cette étiquette est sélectionnée et rendue permanente. Les successeurs du sommet1 sont marqués, par les deux étiquettes temporaires: [5,5,5,5,1,1] sur le sommet 2 et [3,4,5,6,1,1] sur le sommet 3.

2. parmi ces étiquettes temporaires, [3,4,5,6,1,1] est lexicographiquement petite. Cette étiquette est sélectionnée et devient permanente. Les successeurs du sommet 3 sont marqués, alternativement par les deux étiquettes temporaires: [5,5,10,16,3,1] sur le sommet 2 et [6,6,10,10,3,1] sur le sommet 4. La nouvelle étiquette du sommet 2 est dominée par la précédente, [5,5,5,5,1,1], et est par conséquent supprimée.

3. parmi ces étiquettes temporaires existantes, l'étiquette, [5,5,5,5,1,1], est lexicographiquement petite. Une fois sélectionnée, cette étiquette est devenue permanente. Les successeurs du sommet 2 sont marqués, encore par les deux étiquettes temporaires: [6,7,15,15,2,1] sur le sommet 3 et [6,6,10,10,2,1] sur le sommet 4. La nouvelle étiquette du sommet 3 est dominée par la précédente [3,4,5,6,1,1], et est par conséquent supprimée.

4. les étiquettes temporaires [6,6,10,10,2,1] et [6,6,10,10,3,1] sont lexicographiquement égales. Si l'étiquette [6,6,10,10,2,1] est sélectionnée, elle devient permanente, et le sommet 4 n'a aucun successeur, et ainsi aucune nouvelle étiquette temporaire n'est créée.

5. l'étiquette [6,6,10,10,3,1], la seule étiquette temporaire reste dans la liste d'étiquettes temporaires, cette dernière est sélectionnée et devient permanente. Puisque le sommet 4 n'a aucun successeur, ainsi aucune étiquette temporaire n'est créée. La liste d'étiquettes temporaires est vide, on conclut la phase de marquage.

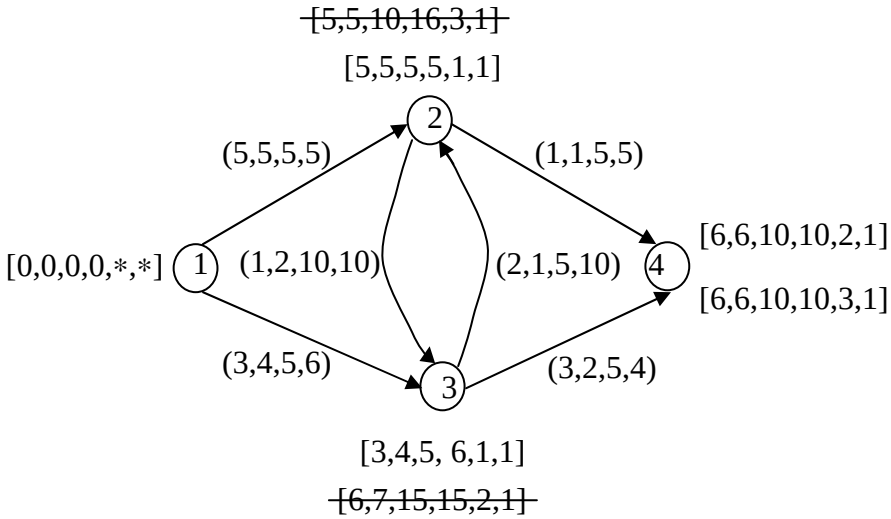

Figure 5.2. Exemple pour le problème (4-S). Les étiquettes barrées sont supprimées.

Tableau 5.1. Chemins efficaces concernent l'exemple (4-S).

de s = 1 à	chemins $p \in E(P_{s,i})$ sont	performances $v(p)$
i = 2	1 ⟶ 2	(5, 5, 5, 5)
i = 3	1 ⟶ 3	(3, 4, 5, 6)
i = 4	1 ⟶ 2 ⟶ 4 1 ⟶ 3 ⟶ 4	(6, 6, 10, 10) (6, 6, 10, 10)

La version révisée présentée dans la section suivante fait usage de (σ-S | μ-M), avec σ ≥1 et μ = 1, et cherche l'ensemble maximal complet des chemins efficaces $\{p \mid p \in P_{s,i}\}$

5.3. Version révisée de l'algorithme de Martins pour le problème (σ-S | 1-M)

Dans ce paragraphe, nous présentons une version révisée de l'algorithme de Martins pour le problème de type (σ-S | 1-M) proposée par Gandibleux et *al*. [34]. Le changement principal concerne le test de dominance pour s'assurer que l'ensemble des chemins efficaces pour un problème de type (σ-S | 1-M) est maximal complet. Pour une meilleure lisibilité, les indices seront notés 1,…, σ pour les fonctions critères de type S-type et μ (μ = 1) correspond aux fonctions critères de type M-type. Soit un sommet i connecté à un sommet j .

La valuation de l'arc (i, j) est donnée par $(\upsilon_{ij}^{1},...,\upsilon_{ij}^{\sigma}, \upsilon_{ij}^{\mu})$. Considérons deux étiquettes temporaires $[\upsilon_{i,h}^{1},...,\upsilon_{i,h}^{\sigma}, \upsilon_{i,h}^{\mu},*,*]$ $(h = 1, 2)$ associées au sommet i. Lorsque $\upsilon_{i,1}^{q} = \upsilon_{i,2}^{q}$ $\forall$ q = 1,...σ et $\upsilon_{i,1}^{\mu} > \upsilon_{i,2}^{\mu}$ l'étiquette 2 est *faiblement non-dominée* par l'étiquette 1. Selon le test de dominance de l'algorithme de Martins, l'étiquette 2 doit être supprimée. Considérons maintenant les étiquettes sur le sommet j. Deux cas se présentent, selon la valeur de υ_{ij}^{μ}. Si $\upsilon_{ij}^{\mu} \leq \min(\upsilon_{i,1}^{\mu}, \upsilon_{i,2}^{\mu})$, alors les deux étiquettes sont équivalentes et, par conséquent, l'étiquette 2 ne doit pas être supprimée. Dans le cas contraire, l'étiquette 2 reste *faiblement non-dominée*. Donc, le test de dominance de la version originale de l'algorithme de Martins ne peut pas identifier tous les chemins efficaces dans cette situation.

Pour résumer, la révision affecte trois aspects dans l'algorithme original:

- elle remplace la valeur 0 dans l'étiquette initiale par ∞ pour la fonction critère max-min correspondante.
- elle étend le test de dominance sur les étiquettes.
- elle élimine les étiquettes permanentes faiblement non-dominées pour la détermination des chemins efficaces à la fin de la phase de marquage.

Remarque 5.1.

Si l'étiquette 1 est faiblement non-dominée par l'étiquette 2 on a le même raisonnement.

La version révisée présentée précédemment utilise un seul critère de type M-type, la version révisée de l'algorithme de Martins est capable de prendre en compte plus d'un de ces critères. Dans la section suivante nous proposons une version révisée de l'algorithme de Martins pour les problèmes (σ-S | 2-M), avec σ ≥1 et μ = 2, et chercher l'ensemble maximal complet des chemins efficaces { p | $p \in P_{s,i}$ }.

5.4. Version révisée de l'algorithme de Martins pour le problème (σ-S | 2-M)

Dans ce paragraphe, nous proposons une version révisée de l'algorithme de Martins pour les problèmes de type (σ-S | 2-M). Le changement principal concerne le test de dominance pour s'assurer que l'ensemble des chemins efficaces pour les problèmes de type (σ-S | 2-M) est maximal complet. La différence entre cette version et la précédente est le nombre des critères de la fonction critère de type M-type. Pour une meilleure lisibilité, les indices seront notés 1,…,σ pour les fonctions critères de type S-type et μ (μ = μ_1, μ_2) correspond aux fonctions critères de type M-type. Considérons maintenant un sommet i connecté à un sommet j (figure 5.3).

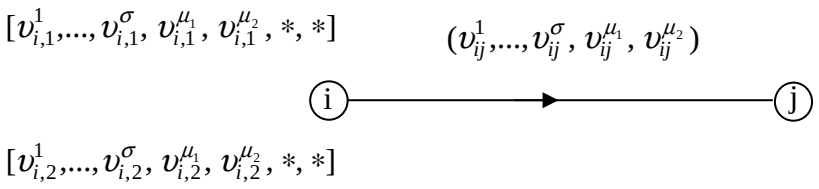

Figure 5.3. Représentation schématique de deux étiquettes.

La valuation de l'arc (i, j) est donnée par $(\upsilon_{ij}^{1},...,\upsilon_{ij}^{\sigma},\ \upsilon_{ij}^{\mu_1},\ \upsilon_{ij}^{\mu_2})$. Considérons deux étiquettes temporaires $[\upsilon_{i,h}^{1},...,\ \upsilon_{i,h}^{\sigma},\ \upsilon_{i,h}^{\mu_1},\ \upsilon_{i,h}^{\mu_2},\ *,\ *]$ ($h = 1, 2$) associées au sommet i. Les étiquettes faiblement non-dominées peuvent être découvertes dans les situations suivantes.

A). Lorsque $\upsilon_{i,1}^{q} = \upsilon_{i,2}^{q} \quad \forall\ q = 1,..., \sigma$

et $\left((\upsilon_{i,1}^{\mu_1} > \upsilon_{i,2}^{\mu_1}\ et\ \upsilon_{i,1}^{\mu_2} = \upsilon_{i,2}^{\mu_2})\ ou\ (\upsilon_{i,1}^{\mu_1} = \upsilon_{i,1}^{\mu_1}\ et\ \upsilon_{i,1}^{\mu_2} > \upsilon_{i,2}^{\mu_2})\right)$.

L'étiquette 2 est faiblement non-dominée par l'étiquette 1. Selon le test de dominance de l'algorithme de Martins l'étiquette 2 doit être supprimée. Considérons maintenant les étiquettes sur le sommet j. Deux cas se présentent, selon les valeurs de $\upsilon_{ij}^{\mu_1}$ ou $\upsilon_{ij}^{\mu_2}$. Si $\upsilon_{ij}^{\mu_1} \le \min(\upsilon_{i,1}^{\mu_1}, \upsilon_{i,2}^{\mu_1})$ ou $\upsilon_{ij}^{\mu_2} \le \min(\upsilon_{i,1}^{\mu_2}, \upsilon_{i,2}^{\mu_2})$, alors les deux étiquettes sont équivalentes et, par conséquent, l'étiquette 2 ne doit

pas être supprimée. Dans le cas contraire l'étiquette 2 reste faiblement non-dominée.

B). Lorsque $\upsilon_{i,1}^{q} = \upsilon_{i,2}^{q} \quad \forall\, q = 1,..., \sigma$

et ($\upsilon_{i,1}^{\mu_1} > \upsilon_{i,2}^{\mu_1}$ *et* $\upsilon_{i,1}^{\mu_2} > \upsilon_{i,2}^{\mu_2}$) .

L'étiquette 2 est faiblement non-dominée par l'étiquette 1. Selon le test de dominance de l'algorithme de Martins, l'étiquette 2 doit être supprimée. Considérons maintenant les étiquettes sur le sommet j. Deux cas se présentent, selon la valeur de $\upsilon_{ij}^{\mu_1}$ et $\upsilon_{ij}^{\mu_2}$. Si $\upsilon_{ij}^{\mu_1} \le \min(\upsilon_{i,1}^{\mu_1}, \upsilon_{i,2}^{\mu_1})$ et $\upsilon_{ij}^{\mu_2} \le \min(\upsilon_{i,1}^{\mu_2}, \upsilon_{i,2}^{\mu_2})$, alors les deux étiquettes sont équivalentes et, par conséquent, l'étiquette 2 ne doit pas être supprimée. L'étiquette 2 reste faiblement non-dominée dans les deux cas suivants:

1) si $(\upsilon_{ij}^{\mu_1} \ge \max(\upsilon_{i,1}^{\mu_1}, \upsilon_{i,2}^{\mu_1})$ *et* $\upsilon_{ij}^{\mu_2} \ge \max(\upsilon_{i,1}^{\mu_2}, \upsilon_{i,2}^{\mu_2}))$.

2) si $(\upsilon_{ij}^{\mu_1} \le \min(\upsilon_{i,1}^{\mu_1}, \upsilon_{i,2}^{\mu_1})$ *et* $\upsilon_{ij}^{\mu_2} \ge \max(\upsilon_{i,1}^{\mu_2}, \upsilon_{i,2}^{\mu_2}))$

ou $(\upsilon_{ij}^{\mu_1} \ge \max(\upsilon_{i,1}^{\mu_1}, \upsilon_{i,2}^{\mu_1})$ *et* $\upsilon_{ij}^{\mu_2} \le \min(\upsilon_{i,1}^{\mu_2}, \upsilon_{i,2}^{\mu_2}))$.

Donc, le test de dominance dans la version originale de l'algorithme de Martins ne peut pas identifier tous les chemins efficaces dans ces situations. De ce fait, les modifications peuvent être fournies de la manière suivante:

1. lorsque les deux étiquettes sont non-dominées ou équivalentes, les deux étiquettes ne doivent pas être supprimées.

2. si l'étiquette 2 est dominée par l'étiquette 1, alors elle doit être supprimée.

3. lorsque l'étiquette 2 est faiblement non-dominée par l'étiquette 1, deux cas se présentent:

   3.1. si $\exists\ q' \in \{1,...,\sigma\}$ tel que $\upsilon_{i,1}^{q'} < \upsilon_{i,2}^{q'}$, alors l'étiquette 2 est supprimée car, les coûts sont positifs, ces deux étiquettes ne peuvent pas être équivalentes dans les prochaines itérations.

3.2. si $v_{i,1}^{q} = v_{i,2}^{q} \quad \forall\, q = 1,....,\sigma$, alors les deux étiquettes ne peuvent pas être supprimées, selon la valeur de $v_{ij}^{\mu_1}$ et/ou $v_{ij}^{\mu_2}$, ces deux étiquettes peuvent être équivalentes sur le sommet j .

Etant donné que quelques étiquettes *faiblement non-dominées* ne peuvent être permanentes, il n'est pas sûr que toutes les étiquettes permanentes correspondent à un chemin efficace à la fin de l'algorithme. Cependant, les étiquettes *faiblement non-dominée*s ne peuvent pas être supprimées car elles contribuent à la détermination des chemins efficaces. Mais en même temps, ne peuvent pas être le sommet destination d'un chemin efficace car, par définition, il n'existe pas une autre étiquette sur ce sommet avec une performance meilleure. Les étiquettes *faiblement non-dominées* sont *cachées* dans la liste des étiquettes permanentes. Ces étiquettes seront utilisées pour déterminer des chemins efficaces sur un sommet destination. Dans cette version un chemin est déterminé de la même manière que l'algorithme de Martins.

Remarque 5.2.

Si l'étiquette 1 est faiblement non-dominée par l'étiquette 2 on a le même raisonnement.

Exemple 5.2.

L'étiquette faiblement non-dominée dans l'exemple de la figure 5.4 représente la situation suivante:

$$v_{2,2}^{1} = v_{2,1}^{1}\ ,\ v_{2,2}^{2} = v_{2,1}^{2}$$

$$\text{et}\ \ (v_{2,2}^{3} = v_{2,1}^{3}\ \ et\ \ v_{2,2}^{4} > v_{2,1}^{4}\ )$$

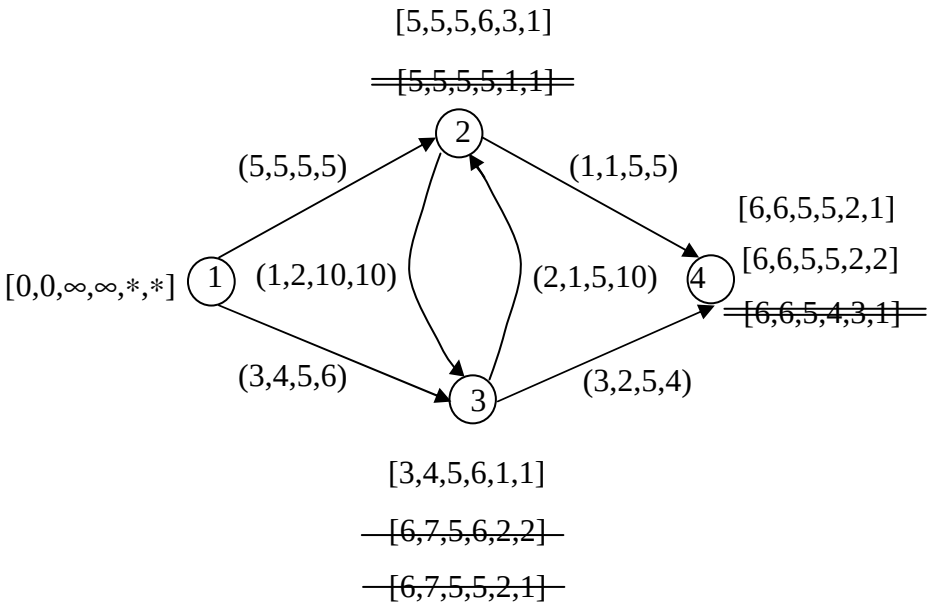

Figure 5.4. Exemple pour le problème (2-S|2-M). Les étiquettes barrées une fois sont supprimées. Les étiquettes barrées deux fois correspondent aux étiquettes cachées pour déterminer un chemin efficace sur un sommet de destination.

Afin d'illustrer la révision de test de dominance, l'exemple de la figure 5.4 est utilisé pour chercher les chemins efficaces pour le problème (2-S|2-M) du sommet source s = 1 aux sommets destinations i = {2, 3, 4}, les deux premiers critères sont à minimiser, les deux derniers sont à maximiser.

1. l'étiquette temporaire [0,0,∞,∞,*,*] associée au sommet s =1, est sélectionnée et devient permanente. Deux étiquettes temporaires sont calculées: [5,5,5,5,1,1] sur le sommet 2 et [3,4,5,6,1,1] sur le sommet 3.

2. parmi ces étiquettes temporaires, l'étiquette [3,4,5,6,1,1] est lexicographiquement petite. Elle est sélectionnée et devient permanente. Les successeurs du sommet 3 sont marqués, par les deux étiquettes temporaires: [5,5,5,6,3,1] sur le sommet 2 et [6,6,5,4,3,1] sur le sommet 4. l'étiquette [5,5,5,5,1,1] est faiblement non-dominée par l'étiquette [5,5,5,6,3,1], produisant une situation dans laquelle les composantes des

critères de S-type sont égales. Par conséquent, cette étiquette ne peut pas être supprimée.

3. l'étiquette [5,5,5,6,3,1] est la deuxième étiquette temporaire sélectionnée. Elle est devenue permanente. Les successeurs du sommet 2 sont marqués, par les deux étiquettes temporaires: [6,7,5,6,2,2] sur le sommet 3 et [6,6,5,5,2,2] sur le sommet 4. La nouvelle étiquette du sommet 3 est faiblement non-dominée par l'étiquette [3,4,5,6,1,1]. Cette situation correspond au cas 3.1 présenté précédemment. Par conséquent, l'étiquette [6,7,5,6,2,2] est supprimée.

4. Dans la prochaine itération, l'étiquette temporaire [5,5,5,5,1,1] est sélectionnée et rendue permanente. Les successeurs du sommet 2 sont marqués, par les étiquettes temporaires: [6,7,5,5,2,1] sur le sommet 3 et [6,6,5,5,2,1] sur le sommet 4. La nouvelle étiquette temporaire sur le sommet 3 encore faiblement non-dominée par l'étiquette [3,4,5,6,1,1], et par conséquent, est supprimée.

5. l'étiquette [6,6,5,5,2,1] et [6,6,5,5,2,2] sont lexicographiquement égales. Supposons que l'étiquette [6,6,5,5,2,1] est sélectionnée et devient permanente. Le sommet 4 n'a aucun successeur, et ainsi aucune étiquette temporaire n'est créée.

6. la liste contient une seule étiquette temporaire [6,6,5,5,2,2]. Cette étiquette est sélectionnée et devient permanente. Encore le sommet 4 n'a aucun successeur, et ainsi aucune étiquette temporaire n'est créée. La liste des étiquettes temporaires est vide, on conclut la phase de marquage.

Tableau 5.2. Chemins efficaces dans l'exemple (2-S | 2-M).

de s = 1 à	chemins $p \in E(P_{s,i})$ sont	performances $v(p)$
i = 2	1 → 3 → 2	(5, 5, 5, 6)
i = 3	1 → 3	(3, 4, 5, 6)
i = 4	1 → 2 → 4 1 → 3 → 2 → 4	(6, 6, 5, 5) (6, 6, 5, 5)

Le tableau 5.2 présente tous les chemins efficaces obtenus. La version proposée touche trois aspects de l'algorithme original:

1. elle remplace la valeur (0,0) dans l'étiquette initiale par (∞,∞) pour les fonctions critères max-min correspondantes.

2. elle étend le test de dominance sur les étiquettes.
3. elle élimine les étiquettes faiblement non-dominées pour la détermination des chemins efficaces à la fin de la phase de marquage.

Nous avons présenté une version révisée de l'algorithme de Matins pour les problèmes de plus court chemin multicritère avec (σ-S | 2-M) critères. Notre révision concerne l'étiquette initiale, le test de dominance sur étiquettes pendant la procédure de marquage et la procédure d'identification des chemins efficaces. Cette version révisée optimise simultanément σ critères de type S-type et 2 critères de type M-type.

CONCLUSION

Une littérature abondante existe pour traiter les divers aspects de différents problèmes de l'optimisation combinatoire monocritère; contrairement à l'optimisation combinatoire multicritère. Les problèmes d'optimisation multicritère présentent des difficultés spécifiques, qui résident dans la caractérisation de l'ensemble des solutions efficaces qui sont le plus souvent de cardinalité infinie.

Dans ce mémoire, nous nous sommes intéressés à un problème particulier de l'optimisation combinatoire multicritère: le problème de plus court chemin multicritère. La variante monocritère de ce problème est considérée comme étant pratiquement résolue. Mais, le cas multicritère est une variante NP-complète qui fait encore l'objet des travaux de recherche. Les algorithmes de résolution du problème de plus court chemin multicritère se basent, essentiellement, sur l'utilisation des algorithmes du problème classique de plus court chemin monocritère ou sur des versions légèrement modifiées.

En effet, dans un premier temps, nous avons étudié les conditions nécessaires et suffisantes d'existence de solutions efficaces pour le problème de plus court chemin d'un source s à un autre sommet puits t et le problème de plus court chemin d'un sommet source s à tout autre sommet dans un réseau bicritère. Puis dans le cas général de réseau multicritère.

Dans un second temps, nous avons donné un aperçu sur les algorithmes d'étiquetage multiples de résolution des deux problèmes étudiés.

Enfin, nous avons étudié la révision de l'algorithme de Martins pour le problème de plus court chemin multicritère avec fonction coût max-min et nous avons présenté une version révisée de l'algorithme de Martins. Cette dernière version permet de prendre en compte plusieurs fonctions critères de type S-type et deux fonctions critères de type M-type. Elle calcule tous les chemins efficaces d'un

sommet donné source, à tous les autres sommets du réseau. La version révisée proposée peut résoudre ainsi les problèmes dans lesquels les valuations des arcs du réseau sont toutes positives, et optimise plusieurs critères simultanément. Elle manipule deux fonctions critères de type max-min et plusieurs fonctions critères linéaires. La révision de l'algorithme de Martins touche le test de dominance et la procédure d'identification des chemins efficaces. On constate alors que le test de dominance de l'algorithme Martins ne peut identifie tous les chemins efficaces puisque cet algorithme supprime les étiquettes faiblement non-dominées. Dans notre version le test de dominance est révisé de façon que les étiquettes faiblement non-dominées ne soient pas supprimées car ces dernières contribuent à la détermination des chemins efficaces sur un sommet de destination. De plus, l'avantage de cette version est la détermination de l'ensemble maximal complet des chemins efficaces. Néanmoins, cette version révisée de l'algorithme de Martins est capable de prendre en compte plus de deux critères de type M-type.

Finalement, on pense aussi qu'il serait intéressant d'essayer de résoudre le problème de plus court chemin multicritère par des méthodes hybrides.

REFERENCES

1. Pallottino, S. et Scutellà, M.G., Shortest path algorithms in transportation models: classical and innovative aspects. In Equilibrium and Advanced Transportation Modelling, kluwer, p. 245-281, 1998.

2. Dijksta, E. W., A note on two problems in connexion with graphs. Numerische, Mathematik, 1 p.269-271, 1959.

3. Collette, Y et Siarry, P., Optimisation multi-objectif, Editions Eyrolles, Paris, 2002.

4. Vincke, P., Problème multicitères. Cahiers du Centre d'Etudes de Recherche Operationelle, 16, p.425-439, 1974.

5. Haimes, Y., Ladson, L. et D. Wismer., On a bicriterion formulation of the problems of integrated system identification and system optimisation. IEEE Transaction on System, Man and Cybernetics, 1, p.396-397, 1971.

6. Steuer, R.E. Mmltiple criteria optimisation: Theory, Computation and Application.Johna,Wiley and Sons, Inc.n New York, 1986.

7. Fourman, M. P., Compaction of symbolic layout using genetic algorithms. In Proceedings of the first international conference on genetic algorithms (ICGA), pages 141-153, 1985.

8. Gondran, M. and Minoux, M. 1985., Graphes et algorithmes. Editions EYROLLES. 537 pages. 1985. Paris.

9. Lacomme, P., Prins, C et Sevaux, M., Algorithmes de graphes. EYROLLES. 405 pages. 2003.

10. Werra, D. Eléments de programmation linéaire avec application aux graphes. Première édition. p. 150-153, 2001.

11. Martins, E.Q.V., On multicriteria shortest path problem. European Journal of Operational Research, 16: 236-245, 1984.

12. Meyer, U., Average-case complexity of single-source shortest-paths algorithms: lower and upper bounds, Journal of Algorithms, Vol 48, Issue1, p. 91-134,2003.

13. Boussedjra, M., Contribution a la résolution du problème du plus court chemin multiobjectif par algorithme évolutionnistes: application aux systèmes de transport intermodal. Thèse de Doctorat, Spécialité Automatique et informatique. Université de Technologie de Belfort-Montbéliard, UTBM, 2005.

14. Bertsekas, D.P., A Simple and Fast Lable Correcting Algorithm for Shortest Paths. Appeared in Networks, Vol.23, p. 703-709, 1993.

15. Skriver, A.J.V., Andersen, K.A., A lable correting approach for solving bicriterion shortest-path problems. Computation & Operations Research 27, p.507-524, 200.

16. Brumbaugh-Smith, J et Shier, D., An empirical investigation of some bicriterion shortest path algorithms, European of Operational Research, 43,216-224, 1989.

17. Hansen, P., Jaumard, B and Vovor, T., Solving the bicriterion shortest path problem from both ends. Technical Report G-98-17, GERAD, 1998.

18. Climaco, J.C.N and Martins, E.Q.V., On the determination of the nondominated path in a multiobjective network problem.Methods in Operations Research, 40 p.255-258, 1981.

19. Sakllariou, R., A computation srudy of parallel algorithms for the all-pairs shortest path problem. In E.A. Lipitakis (ed.) Proceedings of 2nd Hellenic-European Conference on Mathematics and Informatics. Hellenic Mathematical society, ISSN 1105-9737, p. 839-846, 1994.

20. Bertsekas, D.P, Guerriero, F and Musmanno, R., Parallel Asynchronous Label-correcting Methods for Shortest Paths. Journal of Optimisation Theory and Applications, vol.88, n°2, p.297-320,199 6.

21. Florian, M. Nguyen, M. Pallottino, S., A dual simplex algorithm for finding allshortest paths. Network, 11, p. 367-378, 1981.

22. Floyd, R.W. 1962. Algorithm 97, Shortest path. Comm.ACM, 5:345.

23. Pape, U., Implementation and Efficiency of Moore-Algorithms for the Shortest Path Problem, Math. Programming, Vol.7, p.212-222, 1974.

24. Ehrgott, M and Gandibleux, X., A survey and annotated bibliography of multicriteria combinatorial optimization. OR Spectrum 22.p.425-460,2002.

25. Botquélen, Y. Martineau, P. Billaut, J.C et Beptiste, H., Plus rapide chemin bicritère: un problème d'aménagement. Conférence Francophone de MOdélisation et SIMulation, MOSIM04, p. 2004.

26. Cherkassky, B.V., Goldberg, A.V., Radzik,T., Shortest Path Algorithms: Theory and Experimental Evaluation.Proceedings of the Fifth Annual ACM-SIAM Symposim on Discrete Algorithms. p. 516-525, 1993.

27. Bellman, R.E., On a routing problem. In Quart. Appl. Math, 16, p. 87-90, 1958.

28. Ford, L.R. and Fulkerson, D.R., Flow in network. Princeton university.Press, Princeton, NJ, 1962.

29. Moore, E.F., The shortest path through a maze. Proc.Int.Symp. On theory of switching, II, April 2-5:285- 292, 1959.

30. Glover .G. F, Glover.R and Klingman.D, " The Threshold Shortest Path Algorithm", Network, Vol.14 n°1, 1986.

31. Hansen, P., Bicriterion Path Problems in: G. Fandel and T. Gal, editors, Multiple Criteria Decision Making Theory and Application, volume 177 of Lecture Notes in Economics and Mathmatical Systems , pages 109-127. Springer Verlag, Berlin, 1979.

32. Gabrel V., Thiongane, et Vanderpooten D., Un nouvel algorithme de marquage multiple pour déterminer tous les chemins efficaces dans graphe bicritère. In 5ème journée du groupe de travail Programmation Mathématiques Multiobjectif (PM2O), Angers, France, mai 2002.

33. Abbas, M. et Khodja, K., Contribution aux problèmes de cheminement multicritère. Thèse de Magister, Spécialité recherche opérationnelle. Université de USTHB, Alger 2000.

34. Gandibleux. X, Beugnies.F, Randriamasy.S., Martins' algorithms revisited for multi-objctive shortest path problem with a MaxMin cost function. 4OR: Quaterly journal of Operational Research, 4(1): 47-59, 2006.