N°d'ordre : 07/2010 – M/MT

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE
HOUARI BOUMEDIENE
FACULTE DES MATHEMATIQUES

MEMOIRE

Présenté pour l'obtention du diplôme de **Magister**

En : MATHEMATIQUES

Spécialité : **Recherche Opérationnelle, Mathématiques de Gestion**

Par : CHEIKH ASMA

Thème

Etude de la Robustesse et de la Flexibilité pour la Recherche d'un Plus Court Chemin

Soutenu publiquement :20/10/2010 devant le juré composé de :

KHELLADI	Abdelkader	Professeur à l'U.S.T.H.B	Président
ABBAS	Moncef	Professeur à l'U.S.T.H.B	Directeur de mémoire
BOUDHAR	Mourad	Professeur à l'U.S.T.H.B	Examinateur
CHERGUI	Mohamed Elamine	M A A à l'U.S.T.H.B	Invité

Remerciements

Je tiens tout particulièrement à remercier Mr. ABBAS Moncef pour l'encadrement de ce mémoire, leur disponibilité au quotidien, leur patience, leurs précieux conseils et leurs commentaires constructifs tout au long de la rédaction de ce travail.

Je voudrais remercie également les membres du jury :

Mr. KHELLADI Abdelkader, professeur à l'USTHB, qui m'a fait l'honneur de présider ce jury ;

Mr. BOUDHAR Mourad, professeur à l'USTHB, qui a bien voulu examiner mon travail et être membre de ce jury ;

Mr. CHERGUI M^{ed} Amine , Maitre Assistant classe A, pour avoir accepté notre invitation et participé à ce jury

Mais surtout, un tout grand merci à mes parents et à mon mari, qui m'ont donné l'opportunité d'accomplir ces études dans les meilleures conditions et qui m'ont toujours soutenue et encouragée.

Enfin, merci à mes frères, mes amis, mes collègues de travail et toute personne qui, de près ou de loin, a contribué à l'accomplissement de cette mémoire.

Dédicaces

Je dédie ce modeste travail à :

A mes parents ;

A mon mari ;

A ma petite fille AYA ;

A mes frères ;

A ma belle famille.

Asma CHEIKH.

Table des matières

Table des figures

Introduction Générale

La construction des modèles qui protègent contre l'incertitude des données, comme la variabilité des informations, l'imprécision, le manque de connaissance, est un des défis majeurs dans l'optimisation robuste.

L'incertitude affecte plusieurs secteurs, le transport, la telecommunication, les finances et la gestion de stocks. Par exemple, le problème de maximiser la valeur future d'un portefeuille d'investissement est un cas classique où l'incertitude surgit dans la finance et puisque les taux(tarifs) de rentabilité sont incertains, nous faisons face à un programme linéaire avec la fonction objectif incertaine. L'incertitude peut alors être modélisée de deux facons : soit à l'aide de probabilités (on parle alors d'optimisation stochastique) soit sans probabilité (on parle alors d'optimisation **robuste**). Dans des réseaux de telecommunication, le modelé d'un réseau utilise généralement un diagramme pondéré dans lequel les coûts d'arcs sont associés aux retards de transmission. Puisque de tels retards de transmission sont dans le cas général incertain. Un certain nombre de problèmes de plus courts chemin avec l'incertitude dans les coûts d'arcs doivent être résolus pour guider les communications.

L'incertitude peut concerner les différentes parties d'un problème d'optimisation, à savoir : les coefficients de la fonction objectif, les contraintes ou tous les deux. De plus, l'ensemble d'incertitude peut être modélisé de plusieurs manières, comme les sous-ensembles compacts et convexes de $\Re^n$, les produits Cartésiens d'intervalles.

Une des approches possibles à faire face aux problèmes d'optimisation où l'incertitude survient, est de considérer une analyse de pire de cas, comme cela proposé par Kouvelis et Yu (1997) [47]. La motivation pour cette approche est que nous voudrions trouver une solution qui est relativement bonne dans chacune des réalisations possibles des paramètres

(appelée scenario) incluant le scénario catastrophe. Plus précisément, cela consiste dans la recherche de la meilleure solution dans le cas de la pire situation . Un exemple d'une telle approche est, le problème de trouver une solution avec le cas du pire regret. Dans ce cas, résolvant un problème sous l'incertitude est traduit comme la découverte d'une solution qu'en terme de la valeur de la fonction objectif, dévie le moins possible de la solution optimale dans tous les cas. Une telle solution est appelée *la solution de regret maximum.*

Sous l'incertitude de l'intervalle, les versions du regret maximum de plusieurs problèmes d'optimisation combinatoires polynômiaux sont NP-difficile, voir Aissi (2005) [6]. Ainsi, dans la résolution de ces problèmes, le défi de réduire l'espace de solutions devient une question importante. Dans ce contexte, pour savoir quand un élément du problème, représenté par une variable, fait toujours ou partiellement ou ne fait jamais partie d'une solution optimale pour toute la réalisation de données, (des variables strictement forts et non faibles respectivement), constitue une facon de réduire la taille du problème. Par exemple, dans la gestion de projet, un projet peut être modélisé par un graphe acyclique orienté où les arcs représentent des activités. Les longueurs d'arcs dénotent des temps pour compléter des activités individuelles et le chemin le plus long du nœud de départ s au nœud de fin (puis) t fournit le temps nécessaire de compléter le projet en entier. Si les temps d'activité de tâches sont incertains, le défi est de déterminer un ensemble d'activités qui appartiendront toujours ou jamais au plus long chemin. Ces informations sont utiles pour traiter le problème en enlevant les variables non faibles et mettant les variables strictement fort. Un des buts principaux de ce mémoire doit examiner ces questions pour quelques problèmes d'optimisation combinatoires sous l'incertitude.

Nous examinons aussi la version du critère de regret maximum des problèmes de programmation linéaires sous l'incertitude polyédrique dans les coefficients de fonction objectifs et nous donnons des algorithmes pour trouver les solutions exactes.

Nous étudions des problèmes d'optimisation combinatoires sous l'incertitude. Spécifiquement, nous considérons les versions du critère de regret maximum du problème de plus court chemin. Nous examinons le problème de trouver les arcs strictement fort et les arcs non faibles pour ce problème sous l'incertitude d'intervalle et nous donnons les procédures qui réduisent considérablement la taille des formulations du problème.

Le chapitre1 présente des définitions et des généralités sur les graphes, les chemins et les plus courts chemins. Dans le chapitre 2, nous donnons une vue générale sur l'optimization robuste et l'analyse de la robustesse.

Dans le Chapitre 3, nous donnons une courte vue générale de littérature concernant l'optimisation combinatoire robuste. Nous présentons les notions de problème d'optimisation combinatoire incertain, la solution robuste et nous illustrons ces notions avec le problème de plus court chemin sous l'incertitude d'intervalle.

Dans le Chapitre 4, nous étudions les versions incertaines du problème de plus court chemin qui consistant en la découverte d'un chemin de poids minimal connectant deux nœuds indiqués 1 et le m. Nous étudions aussi la classification des arcs et nous considérons ce problème sur des graphes orientés finis où les longueurs d'arcs sont des intervalles non négatifs. Dans le contexte du problème de plus court chemin incertain, nous donnons des conditions suffisantes pour qu'un arc ne pourra jamais être ou soit toujours sur un plus court chemin de 1 à m .

Dans le Chapitre 5, nous étudions la complexité en général du problème de plus court chemin robuste tel que les longueurs des arcs sont définis sur des intervalles, et nous parlons par la suite de la réduction du problème en un problème d'application d'algorithmes polynômiaux qui permettent la recherche des arcs appartenant toujours ou jamais au plus court chemin robuste.

Chapitre 1

Généralités sur les graphes et les chemins

1.1 Introduction

La théorie des graphes est née en 1736 quand Euler démontra qu'il était impossible de traverser chacun des sept ponts de la ville russe de Konigsberg [49] une fois exactement et de revenir au point de départ. Les ponts enjambent les bras de la Pregel qui coulent de part et d'autre de l'ile de Kneiphof. La théorie des graphes constitue un domaine des mathématiques qui, historiquement, s'est aussi développé au sein de disciplines diverses telles que la chimie (modélisation de structures), la biologie (génome), les sciences sociales (modélisation des relations) ou en vue d'applications industrielles (problème du voyageur de commerce). Elle constitue l'un des instruments les plus courants et les plus efficaces pour résoudre des problèmes discrets posés en Recherche Opérationnelle (RO). De manière générale, un graphe permet de représenter simplement la structure, les connexions, les cheminements possibles d'un ensemble complexe comprenant un grand nombre de situations, en exprimant les relations, les dépendances entre ses éléments (réseau de communication, réseaux ferroviaire ou routier, diagramme de succession de tâches en gestion de projet, ...). En plus de son existence purement mathématique, le graphe est aussi une structure de données puissante pour l'informatique.

1.2 Définition et concepts de base

1.2.1 Concepts orientés

Dans beaucoup d'applications, les relations entre éléments d'un ensemble sont orientées, ie, un élément x peut être en relation avec un autre y sans que y soit nécessairement en relation avec x. On parle alors de graphe orienté. Un graphe $G = (V, A)$ est déterminé par :

- un ensemble $V = \{v_1, v_2, \ldots, v_n\}$ dont les éléments sont appelés sommets ou nœuds (ce dernier terme est plutôt dans le contexte des réseaux) ;
- un ensemble $A = \{a_1, a_2, \ldots, a_m\}$ du produit cartésien $V \times V$ dont les éléments sont appelés arcs.

Pour un arc $a = (v_i, v_j)$, v_i est l'extrémité initiale, v_j l'extrémité finale (ou bien origine et destination). L'arc a part de v_i et arrive à v_j . Graphiquement, l'arc a se représente de la manière suivante 1.1 (diagramme sagittal) :

Figure 1.1 – Arc $a = (v_i, v_j)$

Application multivoque :

v_j est successeur de v_i si $(v_i, v_j) \in A$; l'ensemble des successeurs de v_i est noté $\Gamma(v_i)$. v_i est prédécesseur de v_j si $(v_i, v_j) \in A$; l'ensemble des prédécesseurs de v_j est noté $\Gamma^{-1}(v_j)$. L'application Γ qui, à tout élément de V, fait correspondre une partie de V est appelée une *application multivoque*.

1.2.2 Concepts non orientés

Lors de l'étude de certaines propriétés, il arrive que l'orientation des arcs ne joue aucun rôle. On s'intéresse simplement à l'existence d'arc(s) entre deux sommets (sans en préciser l'ordre). Un arc sans orientation est appelé arête. V est constitué non pas de couples, mais

de paires de sommets non ordonnés. Pour une arête $a = (v_i, v_j)$, on dit que a est incidente aux sommets v_i et v_j .

1.2.3 Notion de complexité des algorithmes

Plusieurs problèmes de graphes se rattachent à la classe des problèmes d'optimisation combinatoire. Ces problèmes se répartissent en deux catégories : ceux qui sont résolus d'une manière optimale par des algorithmes efficaces (rapides), et ceux dont la résolution peut prendre un temps exponentiel sur des instances de grandes tailles. On parle respectivement d'algorithmes *polynomiaux et exponentiels.*

Pour évaluer et classer les divers algorithmes disponibles pour un problème de graphes, il nous faut utiliser une mesure de performance indépendante du langage et de l'ordinateur utilisés. Ceci est obtenu par la notion de complexité d'un algorithme, qui consiste à mettre en évidence les possibilités et les limites théoriques du processus calculatoire, en évaluant le nombre d'opérations caractéristiques de l'algorithme dans le pire des cas. Elle est notée O (exemple,$O(n^2)$ pour une fonction qui augmente suivant le carré de la taille des données).

1.2.4 Connexité dans les graphes

Chaine - Cycle

Une chaine est une séquence d'arcs telle que chaque arc ait une extrémité commune avec le suivant. Un cycle est une chaine qui contient au moins un arc, telle que tous les arcs de la séquence sont différentes et dont les extrémités coincident.

Exemple :

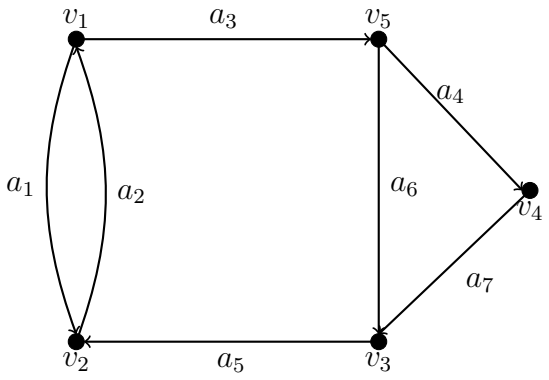

Figure 1.2 – $< a_2, a_5, a_6, a_4 >$ est une chaine de v_1 à v_4 et $< a_4, a_7, a_6 >$ est un cycle

1.2.5 Chemin – Circuit

Ce sont les mêmes définitions que les précédentes mais en considérant des concepts orientés. D'après la figure 1.2 $< a_2, a_3, a_4, a_7 >$ est un chemin de v_2 à v_3. $< a2, a3, a6, a5 >$ est un circuit. Le sous-ensemble de sommets atteints à partir d'un sommet donné, grâce à des chemins, est appelé fermeture transitive de ce sommet. Le terme de parcours regroupe les chemins, les chaines, les circuits et les cycles. Un parcours est :

- élémentaire : si tous les sommets qui le composent sont tous distincts.
- simple : si tous les arcs qui le composent sont tous distincts.
- hamiltonien : passe une fois et une seule par chaque sommet du graphe
- eulérien : passe une fois et une seule par chaque arc du graphe.
- préhamiltonien : passe au moins une fois par chaque sommet du graphe.
- préeulérien ou chinois : passe au moins une fois par chaque arc du graphe.

1.3 Le problème du plus court chemin

Les problèmes de cheminement dans les graphes (en particulier la recherche d'un plus court chemin) comptent parmi les problèmes les plus anciens de la théorie des graphes et les plus importants par leurs applications.

1.3.1 Définitions

Soit $G = (V, A)$ un graphe orienté ; on associe à chaque arc $a = (i, j)$ une longueur $l(a)$ ou l_{ij}. Le problème du plus court chemin entre i et j est de trouver un chemin $\mu(i, j)$ de i à j tel que :

$$l(\mu) = \Sigma_{a \in \mu} l(a)$$

soit minimale.

Interprétation de $l(\mu)$: coût de transport, dépense de construction, temps nécessaire de parcours,... etc.

Remarque :

la recherche du plus long chemin est similaire à la recherche du plus court chemin .

1.3.2 Principe d'optimalité

Le principe d'optimalite énonce le fait qu'un sous chemin d'un plus court chemin est un plus court chemin (la programmation dynamique repose sur ce principe fondamental). **Lemme** : Soient un graphe $G = (V, A)$ et une fonction de pondération $l : V \times V \longrightarrow \Re$, soit $C =< v_1, v_2, \ldots, v_k >$ un plus court chemin de v_1 à v_k et $\forall(i, j)$ tel que $1 \leq i \leq j \leq k$, soit $C_{ij} =< v_i, v_{i+1}, \ldots, v_j >$ un sous-chemin de C allant de v_i à v_j, alors C_{ij} est un plus court chemin de v_i à v_j .

1.3.3 Plus court chemin d'un sommet à tous les autres

Ce problème est aussi appelé le problème de recherche du plus court chemin à origine unique. Beaucoup d'autres problèmes peuvent être résolus par l'algorithme avec origine unique.

- plus court chemin à destination unique (inversion du sens de chaque arc du graphe) ;
- plus court chemin pour un couple de sommets donné ;

– plus court chemin pour tout couple de sommets (algorithme à origine unique à partir de chaque sommet).

Remarque :

Si dans le graphe il existe un circuit de longueur négative, la recherche d'un plus court chemin de l'origine au sommet considéré est sans objet. On peut utiliser le circuit une infinité de fois (présence d'un circuit absorbant qui entraine une diminution perpétuelle de la longueur du chemin).

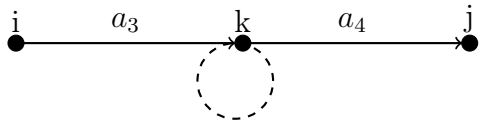

Figure 1.3 – circuit absorbant

1.4 Algorithme de recherche d'un plus court chemin

Nous présentons dans ce qui suit les algorithmes les plus connus pour la recherche du plus court chemin.

1.4.1 Algorithme de Kruskal

L'algorithme de Kruskal est le plus utilisé dans la théorie des graphes pour construire une arborescence de chemin minimal. Il peut être résumé par le pseudo-code suivant :

1. Calculer toutes les distances entre les sommets ;
2. Trier ces distances par ordre croissant, les mettre dans un tableau ;
3. Inclure la plus courte liaison entre deux sommets (i, j) dans l'arbre ;
4. Enlever cette liaison entre (i, j) du tableau ;
5. SI l'inclusion de la liaison (i, j) forme un cycle ALORS Enlever cette liaison (i, j) de l'arbre ;

6. SI l'arbre est connexe, ALORS Aller au 8 ;
7. Revenir au 3 ;
8. Fin.

Inconvénients :

Il se trouve que cet algorithme implique une connaissance totale des distances qui séparent les sommets deux à deux. Ce n'est pas tout car d'une part, à chaque itération il faut vérifier constamment si la nouvelle liaison que l'on vient d'insérer dans l'arbre ne forme pas un cycle. Et d'autre part, à l'étape 6, nous devons aussi vérifier si l'arbre n'est pas connexe. De plus, il faut ensuite parcourir tout le tableau de distances pour trouver le plus court chemin entre deux sommets. Dans tous les cas, cet algorithme nous demande une taille de mémoire importante pour stocker toutes ces informations.

Avantages :

Les plus court chemin entre deux sommets (i, j) quelconques sont connues à tout instant.

Complexité :

La complexité de l'algorithme est $\bigcirc(mn)$ avec $|A| = m$ le nombre d'arêtes et $|V| = n$ le nombre de sommets du graphe G.

1.4.2 Algorithme de Prim

Cet algorithme, comme celui de Kruskal, permet de construire une arborescence contenant le point de départ comme racine. L'arborescence ainsi obtenue nous donne le chemin minimal de la racine jusqu'à chacun des nœuds. Soit I l'ensemble de nœuds déjà inclus dans l'arbre en construction et NI l'ensemble de nœuds non encore inclus. L'algorithme de Prim peut être représenté par le pseudo-code suivant :

1. Initialisation $I = \emptyset$, $NI = \{\text{tous les nœuds}\}$;
2. Mettre le nœud de depart dans I ;

3. SI l'arbre est connexe, ALORS Aller à 6
4. Trouver un nœud b de NI dont la distance a un nœud quelconque de I est minimal. On relie ce nœud a à b et faire $I = I \cup b$, $NI = NI \setminus \{b\}$;
5. Aller à 3
6. Fin

Inconvénients :

De plus, à chaque itération on doit vérifier si l'arbre est connexe pour arrêter l'algorithme. Cela nous mène à le faire fonctionner sur tous les nœuds connus, ce qui va coûter un grand temps de calcul.

Avantages :

Le chemin minimal entre deux sommets quelconques est connu une fois pour toute comme dans le cas de l'algorithme de Kruskal.

Complexité :

La complexité de l'algorithme est $\bigcirc(m + n * log(n))$ avec $|A| = m$ le nombre d'arêtes et $|V| = n$ le nombre de sommets du graphe G.

1.4.3 Algorithme de Bellman-Ford-Moore

Cet algorithme peut être vu comme une version améliorée de l'algorithme proposé par Ford. Cette version est aussi appelée l'algorithme à correction d'étiquette. Ci-dessous se trouve son pseudo-code :

1. Initialisation pour tout nœuds
   (a) distance = infinie
   (b) père = NULL
2. Initialisation du nœud de départ
   (a) distance = 0

(b) père = NULL

3. Enfiler(L, nœud de départ)
4. Nœud à traiter a = défiler(L)
5. Tant que a non NULL Faire
   (a) Pour chaque nœud b relié à a Faire
      i. distance d = distance de a + longueur de l'arc (a, b)
      ii. Si distance actuelle de b > d Alors
         A. distance de b = d
         B. père de b = a
         C. enfiler(L,b)
      iii. Fin Si
6. Fin Tant que

Inconvénients :

A partir du nœud de départ, cet algorithme se réitère jusqu'à ce qu'il ne trouve plus de nœud dont la distance (ou étiquette) peut être améliorée. Dans de nombreux cas, il examine plusieurs fois un nœud déjà visité. Ceci pose aussi un problème de performance non négligeable. De plus, l'inconvénient d'un tel algorithme est le fait de trouver un cycle dont la somme des distances (ou étiquettes) est négative. Puisque l'algorithme compare la distance de chaque nœud, cette dernière se trouve être améliorée continuellement dans un tel cycle.

Avantages :

En insérant astucieusement une instruction de comparaison entre l'étape 1 et 2, on peut terminer l'algorithme dès que l'on tombe sur le nœud d'arrivé. Ceci permet d'éviter de parcourir tous les nœuds pour trouver un chemin. Un autre avantage non négligeable est sa capacité à travailler avec les arcs de poids négatifs. Néanmoins, l'algorithme doit être capable de détecter les cycles d'arcs de poids tous négatifs comme cité précédemment.

Complexité :

La complexité de l'algorithme est $\bigcirc(n*m)$ avec $|A| = m$ le nombre d'arêtes et $|V| = n$ le nombre de sommets du graphe G.

1.4.4 Algorithme de Dijkstra

Cet algorithme ressemble fortement à celui de Bellman-Ford-Moore. Cependant, Dijkstra a eu l'idée de marquer chaque nœud visité et empêché ainsi son algorithme de revenir en arrière. Ci-dessous ce trouve le pseudo-code de l'algorithme de Dijkstra

1. Initialisation de toutes les cases ;
   (a) La case incidente est indéterminée
   (b) La longueur du trajet vaut plus infini
2. Initialisation de la case de départ
   (a) La case incidente est indéterminée
   (b) La longueur du trajet vaut 0
   (c) Insertion de cette case dans la liste A des cases à traiter
3. Tant que A est non vide ET non trouvé Faire
   (a) Trier la liste A
   (b) Sortir la case m ayant la longueur minimale
   (c) Si la case à traiter est le point d'arrivé Alors trouvé est vrai
   (d) Pour chaque case voisine v non marquée de la case à traiter Faire
      i. Nouvelle longueur l = longueur actuelle de m + le trafic en v
      ii. Si la nouvelle longueur l < la longueur actuelle de v Alors
         A. La longueur de v est maintenant l
         B. La case incidente de v est m
         C. Insérer v dans la liste A des cases à traiter
      iii. Fin Si

(e) Fin Pour

(f) Marquer la case m comme traitée

(g) Insérer m dans la liste B des cases déjà traitées

4. Fin Tant que
5. Construire le chemin minimal en parcourant B

Inconvénients :

Cet algorithme comme celui de Bellman-Ford-Moore est de type glouton. C'est pourquoi dans un certain nombre de cas ou les cases à traiter ont un même niveau de trafic l'algorithme n'est pas très performant car il les parcourt toutes. C'est pareil si nous prenons un autre critère de tri comme la distance entre la case à traiter et la case d'arrivée. Nous verrons plus loin comment on peut guider cet algorithme dans la recherche afin d'éviter de parcourir toutes les cases inutilement. Puisque chaque nœud traité est marqué, nous ne pouvons pas revenir en arrière. De ce fait, l'algorithme de Dijkstra est inadapté quand nous avons des arcs de poids négatifs. Dans ce genre de situation, l'algorithme doit pouvoir revenir en arrière pour modifier les étiquettes. C'est le cas de l'algorithme Bellman-Ford-Moore.

Avantages :

Avec cet algorithme, nous pouvons nous arrêter dès que le point d'arrivé devient la case à traiter puisque le chemin que nous avons construit jusque là est un chemin minimal. Le système de marque empêche l'algorithme de revenir en arrière. Il améliore ainsi sa performance.

Complexité :

La complexité de l'algorithme par l'utilisation de la structure des données "*Tas de Fibonacci*" est $\bigcirc(n + m \times \ln(n))$ avec $|A| = m$ le nombre d'arêtes et $|V| = n$ le nombre de sommets du graphe G. Cette structure de données est similaire à celle du tas binomial, mais avec un meilleur temps d'exécution amorti. Les tas de Fibonacci sont utilisés pour

améliorer le temps asymptotique de l'algorithme de Dijkstra, qui calcule les plus courts chemins dans un graphe.

Chapitre 2

Robustesse en recherche opérationnelle

2.1 Introduction

Le terme robuste apparait de plus en plus souvent en aide à la décision [9]. Même si ce qualificatif a généralement le sens de «qui résiste à l'à peu près», il recouvre pourtant des significations différentes et parfois obscures. Le recours à cette notion pour éclairer des décisions soulève, me semble-t-il, de multiples questions. Sans chercher à être exhaustif, je voudrais ici en aborder quelques-unes.

2.2 Robustesse de la solution

Il s'agit de la solution qui constitue le résultat d'une procédure ou d'un algorithme auquel l'auteur s'intéresse. Dans la mesure où cette solution est celle que l'aide à la décision a pour objet de préconiser, il importe qu'elle soit robuste dans un sens qui mérite pourtant d'être précisé. Voici deux brèves citations (choisies parmi beaucoup d'autres possibles) qui illustrent ce point de vue.

«Un des souhaits souvent formulés par les décideurs vis-à-vis de la méthode multicritère qu'ils utilisent est d'avoir une idée de la robustesse du résultat. C'est la raison pour laquelle l'étude de la sensibilité des méthodes multicritères fait partie des grands axes de recherche du domaine. Cette demande traduit la volonté de savoir dans quelle mesure une variation des données due par exemple à une erreur de mesure ou d'estimation risque d'affecter le

résultat donné par la méthode» (Durand et Trentesaux, 2000).
«Nous utiliserons le terme de robustesse pour caractériser la performance d'un algorithme ou plutôt d'un processus complet de construction d'un ordonnancement en présence d'aléas» (Sanlaville, 2002).
Un décideur et, plus généralement, une quelconque partie prenante dans un processus de décision attend souvent, de l'aide à la décision, autre chose qu'une ou quelques solutions résultant de l'application d'un algorithme ou d'une procédure. Dans la mesure où l'aide à la décision ne vise pas uniquement à préconiser des solutions, n'est-il pas nécessaire de reconnaître que la réponse à la question «robustesse de quoi ?» ne doit pas être restreinte aux seules solutions. Il n'est donc pas surprenant que certains auteurs se soient déjà préoccupés de la robustesse dans au moins deux autres directions : robustesse de méthodes, robustesse de conclusions.

2.3 Définition d'une méthode

Une méthode M est une classe bien définie de procédures [9], une procédure P étant une séquence d'instructions qui, appliquées à une instance (ensemble de données) E d'un problème (auquel la méthode est censée s'appliquer), fournit un résultat $R(P, E)$.
Le résultat dont il est question ici consiste le plus souvent en une solution S admissible et remarquable du problème pour l'instance E. Il peut aussi inclure, et parfois même se restreindre à des constats.

2.4 Définition d'une conclusion

Etant donné un ensemble d'instances d'un problème et un ensemble de procédures applicables à ce problème [9], nous appellerons conclusion toute assertion (de nature quelconque, vraie ou fausse) qui vise à tirer parti de certaines des informations contenues dans les résultats $R(P, E)$ relatifs à tout ou partie des couples (P, E) envisagés.

2.5 Robustesse vis-à-vis de quoi ?

Quel que soit l'objet en question (solution, méthode, conclusion, ...) [9], donner un sens au qualificatif «robuste» ne peut se faire sans expliciter les raisons et les facteurs générateurs de contingence, d'arbitraire et d'ignorance vis-à-vis desquels la robustesse est recherchée. Outre leur caractère subjectif, ces raisons, ces facteurs revêtent des formes très variées dont la présence et/ou l'importance dépend beaucoup du contexte décisionnel considéré. Leur présence émane, essentiellement de trois sources :

1. Le caractère imprécis, incertain et, plus généralement, mal connu, voire mal défini, de certaines spécificités ou grandeurs factuelles du problème.
2. Les conditions de mise à exécution de la décision qui sera arrêtée, lesquelles peuvent être influencées par ce que sera l'état de l'environnement au moment où elle sera mise à exécution si elle est ponctuelle ou par les états successifs de cet environnement si elle est séquentielle.
3. Le caractère flou, éventuellement lacunaire, et non nécessairement stable des systèmes de valeurs (et plus particulièrement de préférences) qui sont censés prévaloir pour apprécier la faisabilité et l'intérêt relatif des diverses actions potentielles en tenant compte des conditions envisagées pour leur mise à exécution.

2.6 Robustesse pourquoi ?

Les responsables voulant arrêter une décision ou, plus largement, influencer un processus de décision n'attendent pas, en général, de l'aide à la décision qu'elle leur dicte leur conduite mais, plus simplement, qu'elle leur apporte des informations utiles pour baliser leur champ de réflexions et d'actions. Que ces informations se présentent en termes de solutions, de méthodes ou de recommandations assises sur des conclusions, elles ne leur seront véritablement utiles que si la facon dont elles sont dépendantes ou encore conditionnées par la contingence, l'arbitraire et l'ignorance que recèlent les sources est prise en compte dans un cadre suffisamment large et explicité. Pour qu'il en soit ainsi, il importe donc que

ces informations exploitent non pas un résultat privilégié $R(P, E)$ mais tous ceux envisagés à partir des ensembles et découlant de l'analyse et du formalisme dont il vient d'être question. Pour être utile (répondre à des besoins effectifs), ce mode de prise en compte revêt inévitablement des formes très variées adaptées au contexte décisionnel considéré. Nous illustrons quelques préoccupations auxquelles l'analyse de robustesse peut vouloir chercher à répondre [9].

1. La décision a un caractère ponctuel et exceptionnel : Avec ces restrictions, considérons par exemple le cas où il s'agit de sélectionner une variante parmi un ensemble fini de possibilités pour réaliser un projet ou encore d'attribuer des valeurs numériques à diverses variables afin d'arrêter les caractéristiques structurelles d'une grande réalisation. L'attente des demandeurs d'aide à la décision peut, schématiquement, revêtir deux formes :

   (a) La mise en évidence d'une solution accompagnée des arguments qui conduisent à la préconiser : le plus souvent, cette solution sera mise à exécution et finalement jugée dans des conditions, dans un environnement et selon des systèmes de valeurs qui ne peuvent, au moment de l'étude, être appréhendés avec exactitude. Attendre de cette solution qu'elle soit robuste, c'est la vouloir telle qu'elle puisse, le moment venu, apparaître, autant que faire se peut, comme étant l'une des meilleures et, quelles que soient les circonstances, comme jamais très mauvaise. [47]

   (b) L'élaboration de recommandations balisant le champ des décisions à considérer sur la base de conclusions robustes : ces conclusions ont pour objet de mettre en évidence des décisions ou des fragments de décision dont les avantages et inconvénients sont explicités aussi bien en fonction d'options pouvant conditionner la procédure (pondération des critères, mode de prise en compte de l'attitude face au risque,...) que vis-à-vis d'hypothèses ou scénarios relatifs aux conditions de mise à exécution (par exemple la date) ou encore de certaines caractéristiques de l'environnement dans lequel la décision arrêtée prendra place et sera finalement jugée (nouvelles normes, nouveaux systèmes de prix,...). Le lecteur pourra

trouver des exemples précis de telles conclusions et recommandations dans [8]

2. La décision a un caractère séquentiel : Considérons par exemple ici le cas où la décision revêt la forme d'un plan qui s'étale dans le temps et qui, de ce fait, apparait comme une suite de fragments de décisions. Dans ce cas, le plan peut, en général, être révisé à chaque étape afin de tenir compte de l'évolution des conditions de mise à exécution, des caractéristiques de l'environnement et, le cas échéant, des systèmes de préférences. Dans la mesure où les décisions prises au cours des premières étapes du plan sont susceptibles d'avoir un impact aussi bien sur les possibilités de décision dans les étapes ultérieures que sur les conséquences que ces futures décisions pourront avoir, la robustesse du plan (tel qu'il peut être revu à chaque étape) s'analyse en termes de flexibilité. Ici, l'analyse de robustesse vise à mettre en évidence et à prendre en compte les possibilités d'adaptation et de réaction que la décision qui doit être arrêtée en chacune des étapes considérées préserve pour l'avenir. Quel que soit cet avenir, il s'agit d'arrêter à chaque étape des décisions qui ne rendent pas impossibles ou ne dégradent pas trop les meilleures possibilités de choix ultérieurs et minimisent le risque d'acculer le décideur à des résultats catastrophiques qui auraient pu être évités.

3. La décision concerne l'adoption d'une méthode destinée à être utilisée de facon répétitive dans des conditions et environnements (lieux, moments,...) susceptibles de varier : Considérons ici le cas fréquent où la méthode fait intervenir une classe de procédures définies chacune par un jeu de valeurs précises attribuées à divers paramètres (certains d'entre eux pouvant être des paramètres techniques sans signification concrète bien claire) ainsi que par la facon dont la place et/ou le rôle de certaines règles de procédures ont été fixés (notamment rôle des critères et des contraintes). Ces méthodes ne se réduisent pas aux seules méthodes multicritères usuelles dont l'objet est d'opérer une sélection, un tri ou un rangement sur un ensemble d'actions. Elles concernent également celles qui visent à déterminer, de facon périodique, certaines conditions de fabrication ou de réapprovisionnement.

Conclusion

Pour conclure, nous aimerons attirer l'attention sur la grande diversité des préoccupations qui peuvent se cacher derrière le mot robustesse [49]. Afin d'en mieux comprendre la polysémie, il faudrait chercher à typer les principales situations contextuelles à partir notamment des distinctions suivantes :

- les données introduites font ou ne font pas intervenir l'état de l'environnement au-delà d'un très court terme ;
- la mise à exécution de la décision est appelée à être jugée dans un très court terme ou, au contraire, seulement à moyen ou long terme ;
- la décision implique une mise à exécution immédiate, différée ou progressive.

Chapitre 3

État de l'art sur le problème de plus court chemins robustes

3.1 Introduction

Dans plusieurs études de robustesse dans l'optimisation combinatoire, une solution robuste est une solution qui est acceptable dans une grande majorité de scenarios, et qui n'est jamais trop mal. Cette caractérisation donne place à plusieurs interprétations possibles, et ainsi cède place aux diverses approches de robustesse. Ces approches diffèrent en fonction des modèles utilisés pour la représentation des facteurs d'incertitudes et d'imprécision pour le problème de décision considéré. Deux modèles d'incertitude sont distingués : celui dit par intervalle et celui dit par scenarios. Différentes mesures et approches de la robustesse sont présentées : celles issues de la théorie de decision, celles issues de l'analyse multicritère et celles issues de la programmation mathématique. Elles donnent lieu à plusieurs versions distinctes du problème de plus court chemin robuste dont les complexités et les résolutions sont présentées.

C'est donc au niveau de l'optimisation robuste que nous situons nos travaux [46]. Le choix du modèle correspondant à un problème d'optimisation robuste est également double ; c'est-à-dire qu'il est envisageable de traiter le problème robuste au moyen de scénarios ou bien au moyen **d'intervalles**. Nous faisons le choix du modèle par intervalles. Par ailleurs,

les problèmes robustes rencontrés dans la littérature ou dans le domaine industriel sont généralement modélisés comme des programmes linéaires en variables continues (P) et relèvent principalement de l'incertitude au niveau des coefficients (c) de la fonction objectif. Dans ce contexte, on trouve dans la littérature deux mesures de l'incertitude permettant de résoudre, plus ou moins "facilement" le problème robuste associé : (i) critère du pire cas, (ii) critère du regret. L'approche (ii) semble à ce jour être la plus intéressante car elle permet le traitement du problème robuste en prenant en considération la sensibilité des solution contrairement à l'approche (i) rapporte des solutions très conservatrices basées sur l'attente que le pire cas pourrait bien arriver.
Dans ce chapitre, nous présentons des méthodes qui ont été étudiées pour le problème de plus court chemin robuste. Dans la section 2, on propose les modèles étudiés, dans la section 3, on présente les différentes mesures de robustesse. Et en fin, dans les sections 4 et 5, les complexités et les résolutions des différentes versions du problème de plus court chemin robuste sont présentées.

3.2 Notations et Définitions

Les problèmes d'optimisation combinatoires sont une classe de problèmes d'optimisation où les variables sont binaires. Considérons le graphe orienté $G = (V, A)$ tel que $V = \{1; \ldots, n\}$ est l'ensemble des sommets, et A est l'ensemble d'arcs tel que : $\mid A \mid = n$.
Soit à considérer la classe des problèmes d'optimization combinatoire :

$$\begin{array}{l} Minimiser \sum_{i \in V} l_i x_i \\ \text{tel que } \ x \in X \subseteq \{0,1\}^n \end{array} .$$

Assumons que les coefficients de la fonction objectif sont incertains. Nous appellerons cette classe de problèmes *Problème d'Optimisation Combinatoire Incertain* et nous dénoterons par l le vecteur de Cœfficients de la fonction objectif incertain.

Dans notre travail nous modèlisons l' incertitude en utilisant le concept de scénario. Un scénario est une attribution de valeurs possibles à chaque paramètre incertain du problème.

Définition 3.1. *Un scénario s est une réalisation des longueurs d'arcs, i.e, $l_{ij}^s \in [\underline{l}_{ij}, \bar{l}_{ij}]$ est fixée $\forall (i,j) \in A$.*

Définition 3.2. *Un scénario peut être considéré comme étant la situation du réseau, lorsque le plus court chemin robuste est un chemin qui garantie une bonne performance sous n'importe quelle configuration des longueurs d'arcs.*

Notons par S l'ensemble de tous les scénarios possibles et par D l'ensemble de toutes les valeurs possibles des cœfficients incertains que nous appellerons l'ensemble d'incertitude. Pour chaque $s \in S$, on note par $l^s = (l_1^s, \ldots, l_n^s) \in D$ le vecteur correspondant au scénario s, tel que l_i^s est la valeur du cœfficient i dans ce scénario. Dans ce cas pour chaque $c^s \in D$ correspond un problème d'optimisation combinatoire déterminé :

$$\begin{array}{l} Minimiser \sum_{i \in V} l_i^s x_i \\ \text{tel que } \ x \in X \subseteq \{0,1\}^n \end{array} .$$

Le modèle à intervalle : Dans le modèle à intervalle à chaque l arc est associé un intervalle, noté $[\underline{l}_{ij}, \bar{l}_{ij}]$ tel que $l_{ij} \in [\underline{l}_{ij}, \bar{l}_{ij}]$. Cet intervalle représente l'ensemble des valeurs possible de l'arc (i,j) .

L'ensemble de scenarios S est infini et il est défini comme étant le produit cartésien de m intervalles $[\underline{l}_{ij}, \bar{l}_{ij}]$. Dans ce contexte un chemin p peut prendre n'importe quelle valeur appartenant à l'intervalle $[\underline{l}_{ij}, \bar{l}_{ij}]$ avec $\underline{l}(p) = \sum_{(i,j) \in p} \underline{l}_{ij}$ et $\bar{l}(p) = \sum_{(i,j) \in p} \bar{l}_{ij}$.

Le modèle d'ensemble discret de scenarios : Dans ce modèle, l'ensemble de scenarios S est un ensemble fini. Donc, soit q scenarios donnés, chaque arc (i,j) est associé à un ensemble fini de q valeurs possibles $\{l_{ij}^1, \ldots, l_{ij}^q\}$.

Notons que contrairement aux modèles d'intervalle, n'importe quelle combinaison de valeurs l_{ij} ne constitue pas un scenario. Par conséquent, chaque chemin p est associé à un vecteur $V(p)$ tel que $V(p) = (l^1(p), \ldots, l^q(p))$ tel que : $l^s(p) = \sum_{(i,j) \in p} l_p^s$ tel que $s \in \{1, \ldots, q\}$.

Exemple sur le modèle d'intervalle : Soit à considérer le chemin $p = (1,2,4,5,7)$ dans le graphe G de la figure (3.1) dont les arcs sont estimés par des intervalles. Sous n'importe

quel scénario s, ce chemin a pour valeur $l_s(p)$ appartenant à l'intervalle $[5, 24]$.

Ce modèle est considéré pour le problème de plus court chemin dans [[21],[1],[18],[38],[39]]

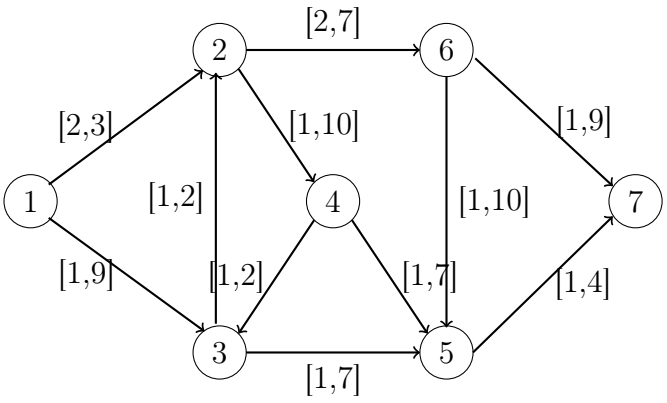

Figure 3.1 – Exemple du modèle d'intervalle

Exemple sur le modèle d'ensemble discret de scénarios :

Soit le graphe 3.2 tel que les poids d'arcs appartiennent à 3 scenarios :

Les chemins élémentaires de 1 à 7 avec leurs valeurs dans les 3 scenarios sont présentés

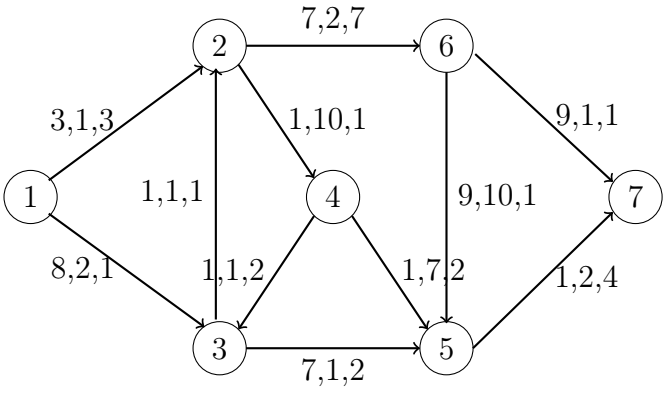

Figure 3.2 – Exemple du modèle d'ensemble discret de scénarios

dans la table 1 :

p	$l^1(p)$	$l^2(p)$	$l^3(p)$
$p_1 = (1.2.6.7)$	19	4	6
$p_2 = (1.2.6.5.7)$	12	15	10
$p_3 = (1.2.4.5.7)$	6	20	10
$p_4 = (1.2.4.3.5.7)$	13	15	12
$p_5 = (1.3.2.6.7)$	25	6	5
$p_6 = (1.3.2.4.5.7)$	12	22	9
$p_7 = (1.3.2.6.5.7)$	18	17	9
$p_8 = (1.3.5.7)$	16	5	7

Table 1 Chemin élémentaire de 1 à 7

Un modèle plus général est présenté dans [32] qui inclut les deux modèles précédents. Dans cette structure générale, la valeur d'arcs est déterminée par un ensemble de paramètres r. On ne connait pas exactement ces paramètres et on considère deux modèles : un ensemble discret de paramètres r-tuples et un sous-ensemble continu de R^r délimité par des contraintes (pour lequelles le modèle d'intervalle est un cas particulier).

3.3 Mesure de robustesse

Dans cette section, nous listons des modèles pour évaluer un chemin en terme de robustesse. Trois approches peuvent être distinguées : la première, venant de la théorie de décision, modélisant le concept de robustesse par un seul critère à être optimisé ; le deuxième utilise la programmation mathématique pour adresser l'incertitude de données ; et le troisième, venant de l'analyse de multicritères, essaye de caractériser les solutions robustes par un ensemble de conditions à être vérifié.

3.3.1 Critère venant à partir de la théorie de décision

Le critère de pire des cas :

Soit un chemin p donné. Le scenario à considérer est celui qui donne la plus mauvaise valeur à ce chemin. La valeur du chemin p, noté $V_{WOR}(p)$ est définie par :

$$V_{WOR}(p) = \max_{s \in S} l^s(p)$$

Exemple :
Soit à considérer le chemin $p = (1, 2, 4, 5, 7)$ dans le graphe de la figure (3.1). Le cas du plus mauvais scenario est celui par lequel chaque valeur d'arc est égale à sa valeur supérieure, dans ce cas $V_{WOR}(p) = 24$.
Le problème est de déterminer le chemin p^*_{WOR} qui minimise V_{WOR} comme suit :

$$V_{WOR}(p^*_{WOR}) = \min_{p \in \zeta} V_{WOR}(p)$$

Sous n'importe quel scenario $s \in S$, p^*_{WOR} à une valeur inférieure ou égale à $V_{WOR}(p^*_{WOR})$. Ainsi $V_{WOR}(p^*_{WOR})$ peut être considéré comme une borne supérieure qui offre une garantie absolue.
Pour cela Kouvelis et Yu [47] ont appelés les solutions qui optimisent ce critère : les solutions robustes absolues. Dans la théorie de décision, ce critère modélise le comportement du decider contre le risque.

Le critère de regret maximum :

Soit un scénario s donné, le choix du chemin p qui n'est pas nécessairement un plus court chemin dans G^s, génère un regret noté $r^s(p) = l^s(p) - l^s(p^*_{G^s})$. Un chemin p est donc évalué par $v_{reg}(p)$ qui est basée sur le scénario de regret maximum

$$v_{reg}(p) = \max_{s \in S} r^s(p)$$

Exemple :

Soit à considerer le graphe de la figure (3.4.2), le chemin $p = (1, 2, 4, 5, 7)$ et le scénario $s^{\underline{l}}$ pour lequel chaque valeur d'arc de A est affectée à sa borne inférieure. Donc, comme le plus court chemin de $G^{s^{\underline{l}}}$ est $p^*_{G^{s^{\underline{l}}}} = (1, 3, 5, 7)$ on à : $r^{s^{\underline{l}}}(p) = 5 - 3 = 2$. Et le regret maximum associé à p est 19. Et donc $v_{reg}(p) = 19$. la solution optimale selon le critère maximum sera noté p^*_{reg} et

$$v_{reg}(p^*_{reg}) = \min_{p \in \zeta} v_{reg}(p)$$

Exemple :

Considérons de nouveau le graphe dans la figure(3.2) pour lequel trois scénarios sont pris en compte. Puisque le chemin le plus court dans chaque scénario est respectivement $p^*_{G^1} = (1, 2, 4, 5, 7)$ de valeur 6, $p^*_{G^2} = (1, 2, 6, 7)$ de valeur 4 et $p^*_{G^3} = (1, 3, 2, 6, 7)$ de valeur 5, nous pouvons déterminer les valeurs $v_W OR(p)$ et $v_{REG}(p)$ Pour n'importe quel chemin p, présenté dans la table 2. Tel que : $v_W OR(p)$ Est la valeur du chemin p selon le critère de pire des cas, et $v_{REG}(p)$ est la valeur du chemin p selon le critère de regret maximum.

p	$l_{WOR}(p)$	$l_{REG}(p)$
$p_1 = (1.2.6.7)$	19	13
$p_2 = (1.2.6.5.7)$	15	11
$p_3 = (1.2.4.5.7)$	20	16
$p_4 = (1.2.4.3.5.7)$	15	11
$p_5 = (1.3.2.6.7)$	25	19
$p_6 = (1.3.2.4.5.7)$	22	18
$p_7 = (1.3.2.6.5.7)$	18	13
$p_8 = (1.3.5.7)$	16	10

Table 2 Valeurs des chemins pour le critère classique

Des solutions optimales selon le critère du pire des cas sont des chemins p_2 et p_4 et la solution optimale selon le critère de regret maximal est le chemin p_8.

Dans le travail de Kouvelis, et Yu [47], une solution qui optimise le critère de regret maximum est appelée solution de déviation robuste. De plus, ils proposent un autre critère de robustesse en terme de déviation relatif qui consiste en la normalisation de la mesure de regret comme suit :

$$v_{REG'}(p) = \max_{s \in S} \frac{l^s(p) - l^s(p^*_{G^s})}{l^s(p^*_{G^s})}$$

Ce critère aussi doit être minimiser.

Les critères classiques venant de la théorie de décision évaluent des solutions sur la base des scénarios extrêmes. L'inconvénient majeur c'est de se concentrer sur un seul scénario, principalement le pire cas, sans prendre en compte tous les autres scénarios. D'autre part, ces critères offrent des garanties d'optimalité. Particulièrement Averbakh dans [23] propose une interprétation très éclaircie du critère de regret maximal. Il suppose, plutôt naturellement, qu'une solution est robuste si c'est une bonne solution sur tous les scénarios. Ainsi, pour un scénario s, des solutions robustes doivent appartenir à l'ensemble, dénoté par ζ^s_ϵ, de la vérification de solutions ϵ optimales :

$$l^s(p) - l^s(p^*_{G^s}) \leq \epsilon$$

L'ensemble, dénoté ζ^{ROB}_ϵ , des solutions robustes est alors l'intersection de tous les ensembles de solutions ϵ optimales, un par scénario, comme suit :

$$\zeta^{ROB}_\epsilon = \bigcap_{s \in S} \zeta^s_\epsilon$$

Il est clair, que si ϵ est trop petit, ζ^s_ϵ sera vide et la cardinalité de ζ^s_ϵ augmente avec ϵ. Autrement, si ϵ est trop grand, donc ζ^s_ϵ contiendra des solutions avec de mauvaises évaluations sur quelques scénarios. Par conséquent, il est particulièrement intéressant de décider que la valeur la plus petite de ϵ tel que ζ^s_ϵ contient au moins une solution. Averbakh montre que cette valeur est $v_{reg}(p^*_{reg})$. En effet, les solutions de ζ^s_ϵ sont tel que :

$$\forall p \in \zeta^{ROB}_\epsilon, \forall s \in S, l^s(p) - l^s(p^*_{G^s}) \leq \epsilon$$

$$\Rightarrow \forall p \in \zeta_{\epsilon}^{ROB}, \max_{s \in S}(l^s(p) - l^s(p^*_{G^s})) = v_{REG}(p) \leq \epsilon$$

Mais, comme $\forall p \in \zeta_{\epsilon}^{ROB}, v_{REG}(p) \geq v_{reg}(p^*_{reg})$, si $\epsilon \leq v_{reg}(p^*_{reg})$ alors ζ_{ϵ}^{s} est vide et autrement, ζ_{ϵ}^{s} contient au moins $v_{reg}(p^*_{reg})$. Par conséquent, $v_{reg}(p^*_{reg})$ est le plus petit ϵ comme cela existe au moins une solution qui est ϵ optimale pour tous les scénarios. Cette interprétation peut être adaptée au critère de déviation relatif en modifiant la définition de solution ϵ optimale comme suit : pour un scénario s, p Est ϵoptimal si et seulement si

$$\frac{l^s(p) - l^s(p^*_{G^s})}{l^s(p^*_{G^s})} \leq \epsilon$$

3.3.2 Méthodologie venant de la programmation mathématique :

Le problème de plus court chemin peut être représenté par un programme linéaire en nombre entier de la forme suivante :

$$(P) \begin{cases} \min \sum_{(i,j) \in A} l_{ij} y_{ij} \\ \text{tel que } \ y \in Y, Y = \{0,1\} \end{cases}$$

Dans les études de la robustesse dans la programmation mathématique, on peut distinguer ceux dont l'incertitude affecte uniquement les elements de la matrice de contrainte ; et ceux dont l'incertitude concerne uniquement les cœfficients de la fonction objectif.
Dans ce context ; une approche très intéressante introduite par Bertsimas et Sim est présentée dans [13], [14], ils ont considérés le modèle d'intervalles pour les cœfficients de la fonction objectif comme suit :

$$l_{ij} \in [\bar{l}_{ij}; \bar{l}_{ij} + \widehat{l}_{ij}]$$

Telle que $\bar{l}_{ij}$ est la valeur minimale pour (i,j) est $\widehat{l}_{ij}$ représentée la valeur de la déviation à partir de la valeur nominale du cœfficient $\bar{l}_{ij}$.
Seulement un sous ensemble du $\bar{l}_{ij}$ va changer pour affecter la solution.
Les auteurs ont donc introduits un paramètre Γ qui représente le nombre maximum des cœfficients qui peuvent dévier de la valeur nominal.
$\Gamma := 0$ veut dire qu'aucun cœfficient ne varie ; par contre $\Gamma := m$ ceci veut dire que tous

les cœfficients changent dans le sens du plus mauvais cas.
Donc Γ est interprété comme étant le niveau de robustesse.
La version du problème de plus court chemin robuste devient :

$$(P)\begin{cases} Min(\sum_{(i,j)\in U}\bar{l}_{ij}y_{ij} + Max\sum_{R/R\subseteq U;|R|\leq\Gamma}\widehat{l}_{ij}y_{ij}) \\ \text{tel que } \ y \in Y \end{cases}$$

3.3.3 Méthodologie venant à partir de la l'analyse multicritère

Dans l'analyse multicritère ; le problème de décision est défini en utilisant un ensemble de solutions ; un ensemble discret de critères et un modèle d'aggregation de ces critères.
La solution est donc décrite par son vecteur dévaluation contenant la valeur de la solution dans chaque critère. Soit à considérer un scénario comme étant un critère. Une partie du résultat obtenu en analyse multicritère peut être utilisé pour l'analyse de robustesse.
Dans ce cas l'analyse de robustesse est basée sur le choix d'un vecteur dévaluation associé à chaque solution, et de la définition d'un modèle d'aggrgation de ces vecteurs d''evaluation.

Vecteur du chemin d'evaluation :

Lorsque l'ensemble de scénarios est discret il est naturel de considérer le vecteur $V(p)$ qui représente toutes les valeurs de p dans différents scénarios comme étant un vecteur d'évaluation.
Des travaux plus récents appliquées à ce principe consistent à admettre que le vecteur $V(p) = (c^1(p), \ldots, l^q(p)$ est équivalent au vecteur $V(p) = (l^{\sigma(1)}(p), \ldots, l^{\sigma(q)}(p))$, pour n'importe quelle permutation $\sigma(i)$ de q scénarios.
A partir de ce principe, un chemin p peut être évalué par un vecteur obtenu en ordonnant les valeurs de $c^i(p)$ dans un ordre décroissant. Ce vecteur est noté $D(p) = (d^1(p), \ldots, d^q(p))$ tel que, $d^1(p) \geq, \ldots, \geq d^q(p)$ avec $d^i(p)$ représente la valeur d'ordre i du chemin p.
Dans d'autres études sur la robustesse, un autre principe est proposé.
Selon ce principe, une solution pour laquelle ces évaluations sont distribuées autour de la moyenne pour les différents scénarios est toujours préférée à une autre solution.
Pour cela une évaluation de la solution est basée sur le vecteur suivant qui est appellé la

généralisation du vecteur de Lorenz : $L(p) = (L^1(p), \ldots, L^q(p))$ avec $L^j(p) = \sum d^i(p)$

Exemple :

Pour le graphe de la figure (3.2) ; les valeur du vecteur de généralisation de Lorenz pour chaque chemin sont présentés dans la table qui suit :

p	$d^1(p)$	$d^2(p)$	$d^3(p)$	$L^1(p)$	$L^2(p)$	$L^3(p)$
$p_1 = (1.2.6.7)$	19	6	4	19	25	29
$p_2 = (1.2.6.5.7)$	15	12	10	15	27	37
$p_3 = (1.2.4.5.7)$	20	10	6	20	30	36
$p_4 = (1.2.4.3.5.7)$	15	13	12	15	28	40
$p_5 = (1.3.2.6.7)$	25	6	5	25	31	36
$p_6 = (1.3.2.4.5.7)$	22	12	9	22	34	43
$p_7 = (1.3.2.6.5.7)$	18	17	9	18	35	44
$p_8 = (1.3.5.7)$	16	5	7	16	23	28

Table 3 Vecteur d'évaluation des chemins de 1 à 7

Modèles d'aggregation pour la robustesse :

Dans l'analyse multicritère, la relation de dominance est définie comme suit :

Définition 3.3. *Soit une famille donnée $F = (f^1, \ldots, f^q)$ de q critères, un chemin p domine un chemin p' si et seulement si $f^i(p) \leq f^i(p')$ pour $i = 1, \ldots, q$, avec au moins une inégalité stricte.*
Le chemin p' est donc dominé par le chemin p, et on note $p\Delta_F p'$

Définition 3.4. *Un chemin p n'est pas dominé si et seulement s'il n'existe pas un autre chemin $p' \in C$ tel que $p'\Delta_F p$.*

Dans le modèle d'intervalle il n'est pas possible d'associer à chaque chemin un nombre fini de vecteurs d'évaluation à cause du nombre infini de scénarios.
Toutefois la relation de dominance reste définie dans ce modèle comme suit :

Définition 3.5. *Un chemin p domine un chemin p' si et seulement si $c^s \leq c^s(p')\forall s \in S$, avec au moins une inégalité stricte.*

Le chemin $p^{'}$ est donc dominé par p et on note $p\Delta p^{'}$.
Quelque soit le vecteur d'evaluation, la solution robuste appartient naturellement à l'ensemble de solutions non dominées. Ainsi, le problème au départ est de déterminer toutes les solutions non dominées.

Exemple :

Pour le graphe de la figure 2, et concernant le vecteur d'évaluation $V(p)$. (Dans table 1), toutes les solutions sont non-domines sauf $p4$ qui est dominé par p_2. Les solutions non dominées sont p_1, p_2, p_8. Et concernant le vecteur le Lorenz. Les solutions non dominées sont p_2 et p_8.
Nous remarquons que le chemin p_4 qui est optimal pour le critère de pire des cas est un chemin dominé.

La principale difficulté vient à partir du nombre immense de chemins non dominées possible. Hansen, à défini un graphe bicritère pour lequel le nombre de chemins non dominées est une fonction exponentielle de l'ordre du graphe.
Perny et Spanjaard [50] ont montré que l'ensemble de solutions non dominées concernant les vecteurs d'évaluation de Lorenz est un sous ensemble de l'ensemble des solutions non dominées concernant les vecteurs $V(p)$.
Ainsi, dans le graphe proposée par Hansen, seulement deux chemins restent non dominées accordant au vecteurs d'evaluation de Lorenz.
Toutefois, un graphe avec un nombre exponentiel de chemins non dominés concernant le vecteur d'evaluation de Lorenz est proposé. Par conséquent, la détermination de toutes les solutions non dominées n'est pas possible dans un cas g'general.
Autres procédures doivent être définis pour l'aggregation du vecteur d'évaluation dans l'ordre de fournir un ensemble de petite taille de solutions robustes soit à mentionner une approche alternative basée sur une procédure lexicographique.
Cette approche est appliquée à un vecteur de problème de location pour le 1-median.
Ces approches sont complètement différentes. Les critères classiques venant à partir de la théorie de décision prennent en considération uniquement les scénarios extrêmes.
La méthodologie introduite par Bertsimas et Sim représente une manière pour qualifié l'analyse du pire des cas en considérant que uniquement un sous ensemble de cœfficients

variant dans le mauvais sens.
Dans l'approche de l'analyse multicritère, le concept de robustesse n'est pas caractérisé par l'optimalité sur un seul critère, mais par une liste de principes et de conditions qui aide à déterminer un ensemble de solutions robustes (cet exemple peut être vide).

3.4 Complexité et résolution des problèmes de plus court chemins robustes dans le modèle de données d'intervalles

3.4.1 Cas du critère de pire des cas :

Cette version du problème de plus court chemin robuste, noté **ROBINTWOR** est étudié dans [18]. Soit un chemin p qui à la valeur $v_{wor} = max_{s\in S} l^s(p)$, le problème est de déterminer le chemin optimal qui minimise cette valeur dans tous les chemins.
Soit un chemin p de 1 à n, le scénario qui maximise $l^s(p)$ est celui pour lequel chaque arc $p \in A$ à la plus grande valeur possible qui est noté $\bar{l}p$.
Ainsi, ce scenario de pire des cas noté $s^{\bar{l}}$, est unique. A travers ces définitions, le graphe d'evaluation par ce scénario de pire des cas est considéré, et à déterminer le plus court chemin qui correspond à p^*, ainsi $v_{wor}(pwor^*) = min_{p\in p}\bar{l}(p)$ Par conséquent :

Théorème 3.1. *[18] Le problème **ROBINTWOR** est polynomial.*

Exemple :
Dans le graphe de la figure (3.1), le chemin optimal correspondant au critère de pire des cas est $p^* = (1, 2, 6, 7) = p_1$, et qui à la valeur 19.

3.4.2 Cas du critère de regret maximum :

Cette version du problème de plus court chemin, dénoté par ROBINTREG, est étudiée dans [[22],[18][36],[38],[39],[41]]. La plupart des résultats sont basés sur la propriété suivante,

établis par Karasan, Pinar et Yaman. Laissez-nous vous rappeler d'abord que, dans cette version du problème, un chemin p a la valeur $v_{reg(p)} = \max_{s \in S}(l^s(p) - l^s(p^s_{G^s}))$.

Propriété 3.1. *[18] Soit un chemin p de 1 à n, le scénario noté $s(p)$, qui maximise $r^s(p)$ est celui pour lequel chaque arc (i, j) appartenant à p à une valeur égale à sa plus grande valeur possible, autrement dit $l_{ij}^{s(p)} = \bar{l}_{ij}$, et chaque arc (k, h) n'appartenant pas à p à une valeur égale à sa valeur inférieure autrement dit $l_{kh}^{s(p)} = \underline{l}_{kh}$.*

Le scénario $s(p)$ est appelé le scénario induit par p. Le problème de déterminer le chemin robuste, selon le critère de regret maximal, est de trouver p^*_{REG} tel que :

$$v_{reg}(p^*_{REG}) = \min_{p \in \zeta}(l^{s(p)}(p) - l^{s(p)}(p^*_{G^{s(p)}}))$$

Exemple : Pour le graphe de la figure , la table 4 présente la valeur de regret maximale de chaque chemin.

p	$l^{s(p)}(p)$	$l^{s(p)}(p^*_{G^{s(p)}})$	$v_{reg}(p)$
$p_1 = (1.2.6.7)$	19	3	16
$p_2 = (1.2.6.5.7)$	24	6	18
$p_3 = (1.2.4.5.7)$	24	5	19
$p_4 = (1.2.4.3.5.7)$	26	5	21
$p_5 = (1.3.2.6.7)$	27	5	22
$p_6 = (1.3.2.4.5.7)$	32	5	27
$p_7 = (1.3.2.6.5.7)$	32	8	24
$p_8 = (1.3.5.7)$	20	5	15

Table 4 Valeur du regret maximum pour chaque chemin de 1 à 7

Le chemin p_8 est optimal pour le critère de regret maximal (p_8 est aussi le chemin optimal pour le critère du meilleur cas.

Complexité

Ce problème à été prouvé NP-difficile pour les graphes orientés par Zielinski [36] avec une réduction à une version particulière d'un problème de partitionnement qui est connu pour être NP-difficile.
En parallèle, Averbakh et Lebedev [22] ont établi que le problème ROBINTREG est fortement NP-difficile pour les graphes non orientés. En utilisant une réduction à un problème du chemin d'hamilton noté HAMIL. Ils ont notés que leurs résultats peut être facilement utilisé pour le cas des graphes orientés acycliques avec une structure de couches (niveaux). Un graphe admet une structure de niveaux s'il est possible de partitioner l'ensemble des sommets V à des sous ensembles disjoints $V_1, \ldots, V_g$; appelés blocs, tel que chaque arc démarre à partir d'un bloc de sommets V_j vers un bloc de sommets suivant V_{j+1}.

Théorème 3.2. *[36] Le problème ROBINTREG est NP-difficile même pour les graphes orientés acycliques avec une structure en niveaux.*

Résolution

Plusieurs algorithmes pour résoudre ROBINTREG ont été proposés.
Dans ces études, ROBINTREG est écrit comme un programme linéaire en nombre entiers avec les variables y_{ij} qui représentent le chemin μ comme suit :

$$y_{ij} = \begin{cases} 1, \text{si} (i,j) \in \mu \\ 0, \text{sinon} \end{cases}$$

Le programme linéaire en nombre entier est noté P^0

$$(P^0) \begin{cases} MinMax(\sum_{(i,j)\in U} c_{ij}^s y_{ij} - x^s) \\ \text{tel que } \sum_{(i,j)\in U} y_{ij} - \sum_{(k,i)\in U} y_{ki} = \begin{cases} 1, \text{si } i = 1 \ \forall i \in X \\ -10, \text{si } i = n \\ 0, \text{sinon} \end{cases} \\ y_{ij} \in \{0,1\} \ \forall (i,j) \in U \end{cases}$$

tel que x^s est la longueur du plus court chemin de 1 à n dans G^s.

Selon la propriété 1, et pour un chemin μ donné représenté par y, le scénario qui accompli $max_{s\in S}\sum_{(i,j)\in U} c_{ij}^s y_{ij} - x^s$ est le scénario induit par $s(\mu)$.
Sous ce scénario la longueur de n'importe quel arc (i,j) peut être écrite en fonction de y_{ij} comme suit : $b_{ij} + (B_{ij} - b_{ij})y_{ij}$. Ainsi,en introduisant les variables x_i qui représentent la longueur d'un plus court chemin de 1 à i sous le scénario $s(\mu)$. Karasan à proposé une nouvelle formulation de ROBINTREG, noté P^1

$$(P^1)\begin{cases} Min\sum_{(i,j)\in U} B_{ij}U_{ij} - x_n \\ \text{tel que } \sum_{(i,j)\in U} y_{ij} - \sum_{(k,i)\in U} y_{ki} = \begin{cases} 1, \text{si } i = 1 \ \forall i \in X \\ -10, \text{si } i = n \\ 0, \text{sinon} \end{cases} \\ x_j \leq x_i + b_{ij} + (B_{ij} - b_{ij})y_{ij} \ \forall (i,j) \in U \\ x_1 = 0 \\ y_{ij} \in \{0,1\} \ \forall (i,j) \in U \\ y_{ij} \in \{0,1\} \ \forall i \in X \end{cases}$$

P^1 est ainsi un programme linéaire mixte. Karasan et al ont résolus le problème en utilisant un programme linéaire solver. Toutefois, quand, G est une grande taille. P^1 ne peut pas être résolu avec exactitude.
La difficulté vient à partir de la contrainte (2) ; si cette contrainte est supprimé ;le problème P^1 devient un problème de plus court chemin simple. Aussi, les auteurs proposent d'appliquer une procédure de prétraitement qui supprime quelques contraintes (2) ;ceux associés avec quelques chemins qui ne peuvent pas appartenir au chemin optimal.
En diminuant la taille de P^1, cette procédure donne possibilité d'augmenter la taille du problème résolu avec exactitude.

3.4.3 Cas de l'analyse multicritère

Dans le modèle d'intervalle, un chemin p domine un chemin p', et on note $p\Delta p'$.
La complexité de ce teste de dominance vient à partir de l'infinité du nombre de scénario à considérer. Dias et Climaco [34] ont étudié ce problème pour des modèles plus généraux

(pour lesquels le modèle d'intervalle est un cas particulier). Ils ont introduits une relation binaire pour comparer l'évaluation des vecteurs des chemins sur la base de deux scénarios extrêmes seulement, le scénario optimal (meilleur), et le scénario pire (plus mauvais).
Dans le cas de scénario optimal ;noté $s^{\underline{l}}$, la valeur de chaque arc est $u \in A$ est égal au plus petit cout : $c_p^{s^{\underline{l}}} = \underline{l}_u$. Et ils ont définis la relation binaire suivante :

Relation 1 : $p \ll p^{'}$ si et seulement si $l^{s^{\bar{l}}} < l^{s^{\underline{l}}(p^{'})}$
Relation 2 : $p < p^{'}$ si et seulement si $[l^{s^{\underline{l}}}(p) = l^{s^{\underline{l}}}(p^{'})\ et\ c^{s^b}(p < l^{s^{\bar{l}}}(p^{'})]$ où $[l^{s^{\underline{l}}}(p) < l^{s^{\underline{l}}}(p^{'})\ et\ l^{s^{\underline{l}}}(p) = l^{s^{\bar{l}}}(p^{'})]$
Relation 3 : $p <\approx p^{'}$ si et seulement si $[c^{s^{\underline{l}}}(p) \leq l^{s^{\underline{l}}}(p^{'})\ et\ c^{s^b}(p) \leq c^{sB}(\mu^{'})]$ on se basant sur ces trois relations, ils ont établit les rśultats suivants dans la comparaison de deux chemins p et p' de C.

- $p \ll p^{'} \Rightarrow p \Delta p^{'}$
- $c^{s^{\underline{l}}}(p^{'}) < c^{s^b}(p) \Rightarrow p$ ne peut pas dominer $p^{'}$
- $l^{s^{\bar{l}}}(p^{'}) < l^{s^{\bar{l}}}(p) \Rightarrow p$ ne peut pas dominer $p^{'}$
- $p \Delta p^{'} \Rightarrow p <\simeq p^{'}$, avec le corollaire suivant :

Corollaire :

[18] Si p n'est pas $<\approx$ dominé alors p n'est pas dominé.

Avec ces résultats ils ont proposé un algorithme basé sur l'énumération de k plus court chemins, pour déterminer l'ensemble des chemins non dominés.
Dans cet algorithme, les chemins sont énumérés dans un ordre croissant de leurs valeurs sur le scénario $s^{\underline{l}}$, alors les chemins $<\approx$ dominés et les chemins $\ll$ dominés sont supprimés. Et finalement, le test de dominance est accompli seulement sur les chemins restants dans le but d'éliminer ceux qui sont dominés.
Dans un cas plus prćis du modèle d'intervalle, le test de dominance complet ne peut pas être accompli.
La propriété suivante tient compte uniquement des critères extrêmes, le critère optimal et le critère de pire des cas.

Soit à indiquer au debut que $p \setminus p^{'}$ est l'ensemble des arcs qui appartiennent à p et n'appartiennent pas à $p^{'}$.

Propriété 3.2. *[18] p domine $p^{'}$ si et seulement si $l^{s^{\bar{l}}}(p \setminus p^{'} \leq l^{s^{b}}(p^{'} \setminus p)$.*

Remarque

Si p et $p^{'}$ n'ont pas d'arcs communs, alors la propriété 2 devient : $p\Delta p^{'} \Leftrightarrow c^{s^{\bar{l}}}(p) \leq l^{s^{\underline{l}}}(p^{'})$

Exemple

Pour le graphe présenté dans la figure 4, $p = (a, b, d, e)$ domine $p^{'} = (a, c, d, e)$. Et on vérifie que $l^{s^{\bar{l}}}(p \setminus p^{'} = 5) < l^{s^{b}}(p^{'} \setminus p) = 6$ avec $l^{s^{\bar{l}}} = 11 > c^{s^{\underline{l}}}(p^{'}) = 10$.

A partir de la propriété, il est possible de proposer un algorithme par dias et climaco [32] basé sur l'énumération des k plus courts chemins. Ils doivent énumérer des chemins en ordre croissant de leur valeur sur le meilleur scénario possible et, en cas d'égalité, par l'ordre croissant de leur valeur sur le scénario catastrophe. Par conséquent, le ième chemin énuméré, dénoté μ_i ne peut pas être dominer par p_K si $i < k$. Autrement dit, p_i domine p_k si et seulement si $l^{s^{\bar{l}}}(p_i \setminus p_K) \leq l^{s^{\underline{l}}}$. Ce test est le seul test à être effectué pour chaque chemin énuméré.

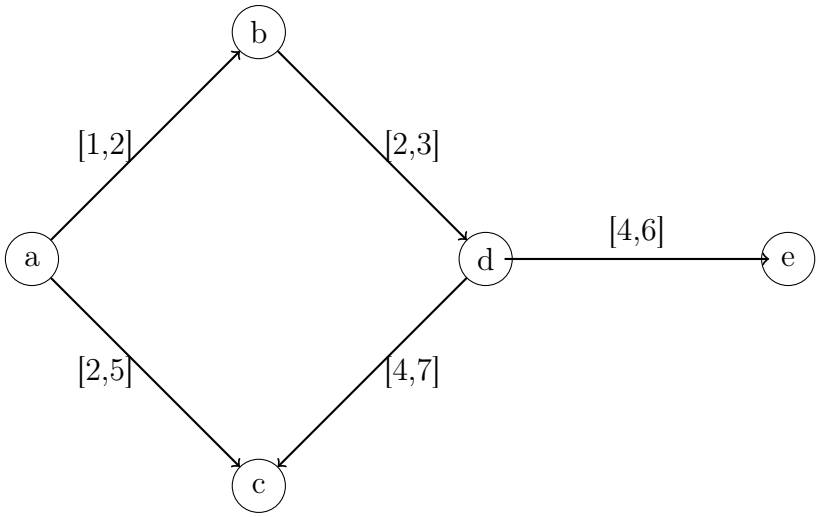

Figure 3.3 – Test de dominance pour les graphes d'intervalles

3.4.4 Cas de la méthodologie de la programmation mathématique

Les auteurs ont montrés que le problème P_{ROB} présenté précédemment (page 42) peut facilement être résolu en utilisant au plus $m+1$ problème de plus court chemins.
En fait, la valeur de la solution optimal de P_{ROB} noté Z^* est donnée par $Z^* = min_{k=1,\dots,m} Z^k$ tel que :

$$Z^k = \Gamma_{\widehat{c}_k} + min_{y \in Y}(\sum_{(i,j)\in U} \overline{c}_{(i,j)\in U} y_{ij} + \sum_{h=1}^{k} (\widehat{c}_h - \widehat{c}_k) y_h)$$

Là il est supposer que les indices des arcs sont ordonnés tel que, $\widehat{c}_1 > \widehat{c}_2 > \dots > \widehat{c}_m$
Par conséquent, dans cette approche, la version robuste présente la même complexité que celle du nominale. Ainsi dans le cas du problème de plus court chemin, la version robuste peut être résolue d'une manière polynomial ce qui rend l'approche très intéressante.

3.5 Complexité et résolution du problème de plus court chemin robuste dans un modèle de scénario discret

3.5.1 Cas du critère de pire des cas :

Complexité :

Cette version du problème du chemin robuste, noté ROBDISWOR, est étudié par Yu [[16],[47]]et Yang qui ont montré que ce problème est faiblement NP-difficile si le nombre de scénarios est borné par une constante et fortement NP-difficile sinon. En effet, ils montrent dans [16] que le problème à 2-partition est réduit au problème ROBDISWOR.

Résolution

Yu et Yang ont proposé un algorithme exacte pour résoudre ROBDISWOR lorsque le nombre de scénarios est borné. Il est basé sur le principe de la programmation dynamique avec un temps de complexité pseudo-polynomial.
En particulier, pour les layered graphes, la compléxitést $0(|U|(L_{max})^q)$ tel que $L_{max} = max_{s\in S} l_s$ avec L_s est la longueur du plus long chemin de 1 à n pour le scénario s. Yu et

Yang ont proposé aussi un algorithme approximative noté H. Pour les graphes de Layered, la complexité est0($|U|_q$).Cet algorithme consiste à calculer le plus court chemin pour chaque scénario et,pour un scénario fictif qui est une moyenne de q scénarios.
Pour déterminer le chemin qui minimise le critère du pire des cas.
Donnons μ^h_{wor} le chemin déterminé par H, les auteurs ont montrés que :

$$\frac{v_{wor}(\mu^H_{wor})}{v_{wor}(\mu^*_{wor})} \leq \frac{q\frac{max_{s\in S}s^s(\mu^H_{wor})}{min_{s\in S}s^s(\mu^H_{wor})}}{\frac{max_{s\in S}s^s(\mu^H_{wor})}{min_{s\in S}s^s(\mu^H_{wor})}+q-1}$$

3.5.2 Cas du critère de regret maximum :

Yu et Yang ont montré que le problème de 2-partition est réduit au problème de plus court chemin rosute dans un modèle de scénario discret selon le critère de regret maximum noté ROBDISREG, pour lequel le problème est de décider s'il existe un chemin μ de 1 à n tel que $v_{reg}(\mu) = v$. La réduction présenté dans ce qui précède reste valide comme $v_{reg}(v) = v_{wor}(\mu)$.Puisque $c^1(\mu^*_{G_1}) = c^2(\mu^*_{G^2}) = 0$.
En plus les algorithmes exactes et approximatives pour résoudre ROBDISWOR peuvent être appliqués, après quelques modifications pour résoudre ROBDISREG.

3.5.3 Cas de la méthodologie multicritère :

Dans le modèle d'ensemble discret de scénarios, le problème de détermination de tous les chemins non dominés peut être résolu en appliquant des algorithmes définis en analyse multicritère.
Dans la literature, deux types d'algorithmes sont proposés pour déterminer tous les chemins non-dominés dans un graphe multicritère : Algorithmes de graphs : qui pour les cas multicritère et ceux basés sur l'énumération de k plus court chemin. L'efficacité de cette approche décroˆquand le nombre de scénarios augmente lorsque le nombre de chemins non dominés croit significativement avec le nombre de scénarios.

3.6 Conclusion

Les problèmes de chemins robustes ont beaucoup été étudiés. Pour la plupart des modèles considérés, la version robuste du problème de plus court chemin robuste devient NP-difficile, sauf pour des pires des cas avec le modèle d'intervalle qui est resoluble en temps polynomial. Toutefois, l'analyse du pire des cas suppose que le scénario qui sera réalisé va affecter la solution.

Ces hypothèses ne seront pas significatifs dans quelques contextes de décision.

Dans telles situations, des nouveaux modèles de robustesse doivent être proposés.

Chapitre 4

Etude de la complexité du problème

4.1 Introduction

Dans ce chapitre, nous voulons examiner la complexité de la version robuste du problème de plus court chemin, où les longueurs d'arc sont donné par des intervalle. Chaque longueur d'arc peut prendre n'importe quelle valeur, n'importe quelle réalisation, dans son intervalle.

Pour cette représentation d'incertitude le critère d'optimisation qui est adopté est celui de la robustesse relative (le critère du regret maximum). Ce critère est discuté dans le livre [47], qui est entièrement consacré à l'optimisation discrète robuste.

Le problème, considéré dans ce chapitre, consiste dans la recherche parmi tous les chemins celui, que sur toutes les réalisations d'intervalles de longueurs d'arcs, réduit au minimum l'écart(la déviation) maximal de la longueur du chemin de la longueur du chemin le plus court dans la réalisation correspondante.

Notre étude n'est pas la première. Kouvelis et Yu [47] a étudié le chemin robuste relatif le plus court avec un ensemble de scénarios discret. Chaque scénario représente une réalisation possible des longueurs d'arc. Ils ont prouvé que le problème est NP-difficile avec un nombre limité de scénarios et il est fortement NP-difficile avec un nombre illimité de scénarios. Ils ont conjecturé qu'aussi le problème avec des données d'intervalle, considérées ici, est NP-difficile.

Malheureusement, dans Montemanni et Gambardella [38] et Karasan et al [18], qui

contiennent la même conjecture que le problème de chemin robuste relatif le plus court avec des données d'intervalle est difficile, sans donner aucune preuve.

Dans ce chapitre nous donnons une réponse complète à la question de la complexité du problème de plus court chemin robuste relatif avec des données d'intervalle, à savoir que le problème est NP-difficile.

4.2 Notation et définitions

Nous gardons les mêmes notations définis dans le chapitre précédent, soit $G = (V, A)$, graphe orienté, on note par $V(G)$ et $A(G)$, l'ensemble des nœuds et des arcs respectivement, on suppose que :$|V(G)| = m$, $|A(G)| = n$. On considère deux nœuds spécial de G le nœud original 1 et le nœud terminal m.

Soit S l'ensemble des scénarios pour les longueurs d'arcs de G et soit D l'ensemble des données incertaines. On note par l_{ij}^s les longueurs d'arcs (i, j) non négatifs dans le scénario s $l_{ij} \geq 0, \forall (i, j) \in A$et on assume que D est le produit cartésien des intervalles, ceci veut dire que, chaque l_{ij}^s peux prendre une valeur arbitraire dans l'intervalle $[\underline{l}_{ij}, \bar{l}_{ij}]$. ie $l_{ij}^s \in [\underline{l}_{ij}, \bar{l}_{ij}]$.

Soit P l'ensemble de tous les chemins de G de 1 à m. On note la longueur du chemin $p \in P$ dans le scénario s par $l_{ij}^s = \sum_{(i,j)\in P} l_{ij}^s$.

Rappelons quelques notions concernant le problème de plus court chemin robuste.

Définition 4.1. *La déviation robuste d'un chemin $p \in P$ dans le scénario s, noté par d_p^s, est la différence entre la longueur du chemin p dans s et la longueur du plus court chemin $p^*(s) \in p$ dans le scénario s, ie $d_p^s = l_p^s - l_{p^*(s)}^s$.*

Définition 4.2. *Un chemin $p^r \in P$ est un plus court chemin relativement robuste, s'il admet la plus petite (à travers tous les chemins de P) maximum (à travers tous les scénarios de S) déviation robuste, ie $s_p \in argmax_{s\in S}(l_p^s - l_{p^*(s)}^s)$.*

Définition 4.3. *Pour un chemin donné $p \in P$, un scénario s_p qui remet la déviation robuste pour p maximal est appelé le pire scénario relatif s_p, ie $s_p \in argmax_{s\in S} d_{p^*}^s$.*

Proposition 4.1. *[18]. Pour tous $p \in P$ le scénario de déviation robuste (un scénario relativement pire) s_p peut être obtenu comme suit :*

$$l_{ij}^{s_p} = \begin{cases} \bar{l}_{ij} \ si(i,j) \in P, \\ \underline{l}_{ij} \ si(i,j) \notin P, \end{cases}$$

4.3 Complexité du problème de plus court chemin de déviation robuste

Dans cette section, nous prouvons que le problème de plus court chemin de déviation robuste est NP-difficile en montrant qu'un problème de décision correspondant est NP-complet. Le problème de décision du plus court chemin de déviation robuste, est défini comme suit :

Input : Un graphe orienté $G = (V, A)$, des longueurs sur les arcs $(i, j) \in A$ sont déterminés par des d'intervalles $[\underline{l}_{ij}, \bar{l}_{ij}]$, avec $l_{ij} \geq 0$, deux nœuds 1 et m sont distingués comme l'origine et le nœud de destination, respectivement et un entier non négatif D.

Question : Existe t'il un chemin p de 1 à m dans G tel que $d_p^{s_p} \leq D$?

On montre que le problème de décision du plus court chemin de déviation robuste que nous allons noté par la suite Décision-RRSPP est NP-complet en lui réduisant un certain problème de PARTITION modifié, appelé MPARTITION.

Le problème MPARTITION est défini comme suit :

Input : Un ensemble fini d'entiers positifs $A = \{a_1, \ldots, a_q\}$, ayant la somme totale de $2b$, et un entier positif $K < q$.

Question : Existe t'il un sous-ensemble $A' \subset A$ qui résume exactement à b et $\mid A' \mid = K$?

Théorème 4.1. *[36] Décision-RRSPP est NP-complet.*

Preuve. Nous prétendons qu'une instance de MPARTITION est polynomiallement réductible à un cas de Décision-RRSPP.

La réduction se procède comme suit. à chaque instance de MPARTITION, on associe un graphe $G^{'} = (V^{'}, A^{'})$ avec $4q + 3$ nœuds notés $1, 2, \ldots, 4q + 3$. Le nœud $2i$ $i = 1, \ldots, q$, est adjacent au nœuds $2i - 1, 2i + 1$ et $2(2q - i + 2)$. Les arcs $(2i - 1, 2i), (2i, 2i + 1)$ et $(2i, 2(2q-i+2)), i = 1, \ldots, q$, admettent des longueurs d'intervalles $[0, M], [M + a_i, M + a_i]$ et $([q-i)M + \sum_{j=i}^{q} a_j, 4qM]$, respectivement. M est un nombre tel que $M > 2b$. Posons $M = 2b+1$. Les intervalles à un point ont été présentés dans la figure 4.1 comme des nombres précis. le nœud $2(2q-i+2), i = 1, \ldots, q$ est adjacent au nœuds $2(2q-i+2)-1$ et $2(2q-i+2)+1$. Les arcs $(2(2q - i + 2) - 1, 2(2q - i + 2))$ et $(2(2q - i + 2), 2(2q - i + 2) + 1), i = 1, \ldots, q$, admettent des longueurs d'intervalles$[M + a_i, M + a_i]$ et $[0, M]$ respectivement. Entre les nœuds $2i - 1$ et $2i + 1$ respectivement, et d'une manière similaire entre, $2(2q - i + 2) - 1$ et $2(2q-i+2)+1$ il existe des arcs $(2i-1, 2i+1)$ et $(2(2q-i+2)-1, 2(2q-i+2)+1)$ avec des longueurs d'intervalles $[0, 2M]$ pour $i = 1, \ldots, q$. Les arcs $(2q + 1, 2q + 2)$ et $(2q + 2, 2q + 3)$ les deux admettent des longueurs d'intervalles $[0, 4qM]$. L'arc $(2q + 1, 2q + 3)$ admet la longueur d'intervalle $[0, 0]$. Les arcs $(1, 2q + 2)$ et $(2q + 2, 4q + 3)$ admettent des longueurs d'intervalles $[(q - K)M + b, 4qM]$ et $[KM + b, 4qM]$, respectivement. Ceci complète la définition du graphe $G^{'}$. La construction de $G^{'}$ est faite dans un temps polynomial dans la taille de MPARTITION. Le reste des input de Décision-RRSPP est défini comme suit. Le nœud $1 \in V^{'}$ est le nœud 1, et le nœud $m \in V^{'}$ est le nœud $4q + 3$, et le paramètre D est posé $3qM$.

Nous montrons maintenant, qu'il existe un $p \in P$ dans $G^{'}$ tel que $d_p^{s_p} \leq 3qM$ si et seulement s'il existe un sous ensemble $A^{'} \subset A$ qui résulte exactement à b et $A^{'} \subset A, \mid A^{'} \mid = K$.

$\Rightarrow$ Soit p un chemin tel que $d_p^{s_p} \leq 3qM$. Le scénario s_p est le scénario de déviation robuste pour p déterminé par la proposition 1. Nous déterminons un sous ensemble $A^{'} \subset A, \mid A^{'} \mid = K$, et nous montrons qu'il resume exactement à b. $\diamondsuit$

D'abord, nous donnons deux propriétés du chemin p.

Propriété 4.1. *[36] Le chemin p doit traverser l'arc* $(2q+1, 2q+3)$.

Dans le but de démontrer la propriété 1, nous supposons contrairement que, p avec $d_p^{s_p} \leq 3qM$ utilise un des arcs parallèle à $(2q+1, 2q+3)$. Remarquons que ces arcs ont des intervalles de longueur avec une borne supérieur égal à $4qM$. Il est assez grand d'interdire au chemin p de traverser ces arcs. On le voit facilement que chaque chemin de l'ensemble P traversant un de ces arcs parallèles admet la déviation robuste dans son scénario catastrophe relatif d'au moins $4qM + 2M$. Cela contredit notre supposition.

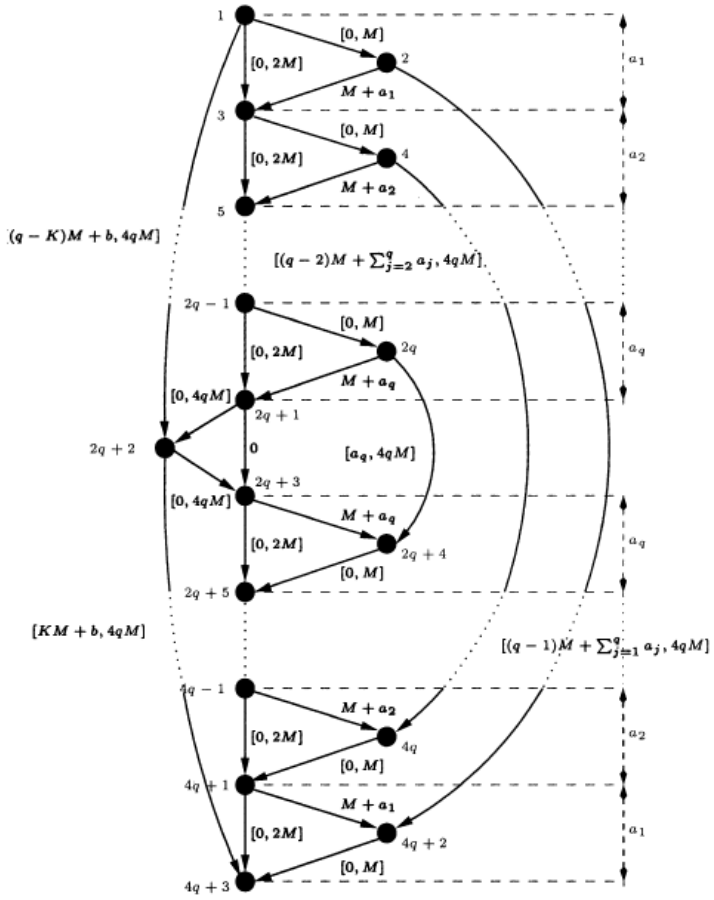

Figure 4.1 – Le graphe G'

Propriété 4.2. *[36] Le chemin p doit utiliser l'un des arcs* $(2i-1, 2i)$, $(2i, 2i+1)$ *(la partie droite du triangle qui procède l'arc* $(2q+1, 2q+3)$*) et l'arc* $(2(2q-i+2)-1, 2(2q-i+2)+1)$ *(la partie gauche du triangle qui succède à l'arc* $(2q+1, 2q+3)$*) ou l'arc* $(2i-1, 2i+1)$ *(la*

partie gauche) et les arcs $(2(2q-i+2)-1, 2(2q-i+2)), (2(2q-i+2), 2(2q-i+2)+1)$ *(la partie droite), pour* $i = 1, \ldots, q$.

Pour démontrer la propriété 2, nous assumons le contraire, qu'il existe i, $1 \leq i \leq q$, tel que le chemin p avec $d_p^{s_p} \leq 3qM$ traverse l'un des arcs $(2i-1, 2i+1)$ et $(2(2q-i+2)-1, 2(2q-i+2)+1)$ simultanément.

Considérons le cas ou il n'existe pas de i tel que le chemin p utilise les arcs $(2i-1, 2i+1)$ et $(2q-1+2)-1, 2(2q-i+2)+1)$, simultanément. Donc, il existe aumoins un i tel que le chemin p utilise les arcs $(2i-1, 2i), (2i, 2i+1), (2(2q-i+2)-1, 2'2q-i+2))$ et $(2(2q-i+2), 2(2q-i+2), 2(2q-i+2)+1)$, simultanément. Choisissons un chemin $p^{'} \in P$, qui traverse l'arc $(2q+1, 2q+3)$ avec des longueurs d'arcs sur $p^{'}$ appartenant à leurs bornes inférieurs dans s_p. Ce chemin admet uniquement un arc en commun avec p, qui est $(2q+1, 2q+3)$. Le chemin $p^{'}$ peut être déterminé d'une manière unique. Nous arrivons immédiatement à une contradiction, tant que $l_p^{s_p} > 4qM + 2b$ et $l_{p^{'}}^{s_p} > 4qM + 2b$ ainsi, $d_p^{s_p} > 3qM$.

Considérons le cas tel qu'il existe aumoins un i tel que le chemin p utilise les arcs $(2i-1, 2i+1)$ et $(2(2q-i+2)-1, 2(2q-i+2)+1)$, simultanément. Nous dénotons par $i^{'}$ plus petit que i. Nous choisissons un chemin $p^{'} \in P$ qui traverse l'arc $(2i^{'}, 2(2q-i^{'}+2))$ parallèle à l'arc $(2q+1, 2q+3)$, avec des longueurs d'arcs sur $p^{'}$ posés à leurs bornes inférieurs dans s_p. Le chemin $p^{'}$ à des arcs disjoints de p et peut être déterminé uniquement. Ainsi, $l_p^{s_p} \geq 4qM$ et $l_{p^{'}}^{s_p} \leq (q-1)M+2b$, la différence entre $l_p^{s_p}$ et $l_{p^{'}}^{s_p}$ est aumoins de $3qM+M-2b$. Par définition, $2b < M$ et par conséquent $d_p^{s_p} > 3qM$. Ceci contredit notre supposition et donc la démonstration de la proposition 2 est complète.

Nous retournons à la preuve principale. A partir des Propriétés 1 et 2, nous déduisons que le chemin p doit traverser l'arc $(2q+1, 2q+3)$, q fois la partie droite et la partie gauche des triangles de $G^{'}$. Donc, pour le scénario s_p est de longueur $4qM + 2b$. Ainsi $d_p^{s_p} \leq 3qM$ et $l_p^{s_p} = 4qM + 2b$, la longueur des arcs du plus court chemin, dans s_p est aumoins de $qM + 2b$. Un chemin de l'ensemble P qui traverse l'arc $(2q+1, 2q+3)$ avec des longueurs d'arcs appartenant à leurs bornes inférieurs dans s_p dans l'un des chemins d'une longueur exactement égal à $qM+2b$. Nous le notons par $p^*(s_p)$. Si p utilise la partie droite du triangle

$p^*(s_p)$ utilise la partie gauche, et de la même manière, si p utilise la partie gauche $p^*(s_p)$ utilise la partie droite. Le chemin $p^*(s_p)$ admet uniquement un arc en commun avec p, qui est l'arc $(2q+1, 2q+3)$. Et donc, nous obtiendrons la remarque suivante :

Remarque :

Le chemin $p^*(s_p)$ satisfait aussi les propriétés 1 et 2.

On montre que le sous chemin de $p^*(s_p)$ à partir du nœud 1 à $2q+1$ admet une longueur $l^{'}$ égal à à $(q-K)M+b$. Nous assumons, le contraire que $l^{'} < (q-K)M+b$. Et donc, le sous graphe de $p^*(s_p$ de $2q+1$ à $4q+3$ est de longueur $l^{''} > KM+b$ Ceci implique l'existence d'un chemin plus court que $p^*(s_p)$. Il est composé des sous graphe de $p^*(s_p)$ de 1 à $2q+1$ et des arcs $(2q+1, 2q+2), (2q+2, 4q+3)$, De même, nous assumons que $l^{'} > (q-K)M+b$. Le sous graphe de $p^*(s_p)$ de $2q+1$ à $4q+3$ est de longueur $l^{''} < KM+b$, il s'ensuit qu'il existe un chemin plus court que $p^*(s_p)$. Ce chemin contient les arcs $(1, 2q+2), (2q+2, 2q+3)$ et le sous graphe de $p^*(s_p)$ de $2q+3$ à $4q+3$.

Comme le chemin $p^*(s_p)$ est de longueur $qM+2b$, le sous graphe de $p^*(s_p)$ à partir du nœud $2q+3$ à $4q+3$ est de longueur $KM+b$.

Déterminons un sous ensemble $A \setminus A^{'}$. Si le sous graphe de $p^*(s_p)$ à partir de 1 à $2q+1$ traverse l'arc $(2i, 2i+1)$ (la partie droite du triangle), donc a_i est inclue dans $A \setminus A^{'}, i = 1, \ldots, q$. Sinon, (il traverse $(2i-1, 2i+1)$) a_i est inclue dans $A^{'}$. Ainsi le sous graphe est de longueur $(q-K)M+b, M > \sum_{i=1}^{q} a_i$, et uniquement les arcs $(2i, 2i+1)$ admettent les longueurs $M+a_i, i = 1, \ldots, q$ (le reste des arcs du sous graphe sont de longueur 0 dans s_p), Il doit utiliser $q-K$ fois la partie la partie droite du triangle et k fois la partie gauche. Ceci donne $\mid A \setminus A^{'} \mid = q-K$ et $\sum_{i \setminus a_i \in A \setminus A^{'}, 1 \leq i \leq q} a_i = b$. De cela et du fait que les sommes jusqu'à $2b$, nous pouvons conclure que : $\sum_{i \setminus a_i \in A^{'}}, 1 \leq i \leq q a_i = b$ et $\mid A^{'} \mid = K$.

$\Leftarrow$ Assumons qu'il existe un sous ensemble $A^{'} \subset A$ qui résume exactement à b et $\mid A^{'} \mid = K$. Nous allons construire un chemin $p \in P$ avec $d_p^{s_p} = 3qM$.

Nous observons que chaque element $a_i \in A, i = 1, \ldots, q$ correspond aux deux triangles $(2i-1, 2i, 2i+1)$ et $(2(2q-i+2)-1, 2(2q-i+2), (2q-i+2)+1))$ relié pr l'arc $(2i, 2(2q-i+2))$ voir 4.1. Si $a_i \in A^{'}$, donc nous incluons les arcs $(2i-1, 2i), (2i, 2i+1)$

(la partie droite du premier triangle) dans le chemin p, sinon ($ai \in A \setminus A^{'}$), nous incluons les arcs $(2i-1, 2i+1)$ (la partie gauche du premier triangle) et $(2(2q-i+2)-1, 2(2q-i+2), (2'2q-i+2), 2(2q-i+2)+1)$ (la partie droite du second triangle) dans le chemin p. Pour compléter p nous ajoutons les arcs $(2q+1, 2q+3)$.

Considérons le scénario s_p déterminé selon la Proposition 1. Notons que chaque element $a_i, i = 1, \ldots, q$ appartenant soit à $A^{'}$ ou bien à $A \setminus A^{'}$. Donc le chemin construit p utilise la partie droite du premier triangle, et la partie gauche du second si $a_i \in A^{'}$ et la partie gauche du premier triangle et la partie droite du second si $a_i \in A \setminus A^{'}$ pour chaque i. Le sous ensemble $A^{'}$ resume à b et $\mid A^{'} \mid = K$. Donc le chemin p utilise k fois les parties droites et $q-K$ fois les parties gauches des triangles, qui précède l'arc $(2q+1, 2q+3)$. Ainsi la longueur du sous chemin de p de 1 à $2q+1$ est égal à $2qM+b$ dans s_p. De même autant que le sous-ensemble $A \setminus A^{'}$ est concerné, qui résume exactement à b et $\mid A \setminus A^{'} \mid = q-K$, le chemin p doit utiliser $q-K$ fois les parties droites et K fois les parties gauches des triangles qui succèdent d'arc $(2q+1, 2q+3)$. La longueur du sous chemin de p à partir de $2q+3$ à $4q+3$ est égal à $2qM+b$ sans s_p. Ainsi, la longueur $l_p^{s_p} = 4qM+2b$. Le reste des chemins de l'ensemble P dans $G^{'}$ sont d'une longueur aumoins de $qM+2b$. En conséquence, le plus court chemin $p^*(s_p) \in P$ sans s_p admet une longueur $qM+2b$ ie $l_{p^*(s_p)}^{(s_p)} = qM+2b$. Ainsi, $d_p^{s_p} = l_p^{s_p} - l_{p^*(s_p)}^{(s_p)} = 3qM$.

Il reste pour montrer que la Décision-RRSPP est dans NP. C'est à dire nous vérifions que si p est un chemin de 1 à m dans G et que $d_p^{s_p} \leq D$ dans le scénario de déviation robuste s_p déterminé par la Proposition 1. La vérification peut être effectué dans un temps polynômial par rapport à la taille de G donc il revient à résoudre le problème de plus court chemin dans G où les longueurs d'arc sont déterminées par le scénario s_p. Ainsi, la Décision-RRSPP est NP-Complète.

A partir du Théorème 1 nous obtenons immédiatement la complexité du problème de plus court chemin de déviation robute RRSPP.

Corollaire 4.1. *[36] Le problème de plus court chemin de déviation robuste RRSPP est NP-Difficile.*

4.4 Conclusion

Dans ce chapitre, nous avons montré que le problème de plus court chemin de déviation robuste d'un graphe orientés, tel que les longueurs d'arcs sont spécifiés par des intervalles, est NP-Difficile.

Chapitre 5

Etude critique et application

5.1 Introduction

Dans ce chapitre, nous considérons les versions incertaines du problème de plus court chemin, qui consiste dans la découverte d'un chemin de longueur minimale connectant deux nœuds indiqués 1 et m. Nous considérons le problème sur des graphes orientés finis où les longueurs d'arcs appartiennent aux intervalles non négatifs. Nous acceptons l'existence des cycles et des longueurs d'arcs tel que, les bornes inférieurs et supérieures de quelques intervalles peuvent avoir la même valeur. Ces problèmes ont des applications importantes dans le transport et à la telecommunication, où il n'est pas facile d'évaluer des coûts d'arcs d'une manière exacte. D'autres applications peuvent être trouvées dans Montemanni et all (2004) [38].

Le problème que nous considérerons est le problème du plus court chemin incertain sur un graphe orienté fini sous l'intervalle des longueur d'arcs incertaines. Kouvelis et Yu (1997) [47] ont étudié ce problème sous l'incertitude discrète et prouvé que (*le le problème de plus court chemin robuste absolue*) et (*Le problème de plus court chemin de déviation robuste*) sont NP-complet pour un nombre limité de scenarios. De plus les auteurs ont prouvé que le problème devient fortement NP-difficile pour un nombre illimité de scenarios, voir la Table 3.1. Dans ce chapitre, nous modélisons l'incertitude de données en traitant les longueurs d'arcs comme des intervalles non-négatifs, c'est-à-dire chaque longueur d'arc peut prendre

n'importe quelle valeur dans son intervalle.

Averbakh et Lebedev (2004) [21] ont prouvé que le problème de plus court chemin de déviation robuste sous l'incertitude d'intervalle est NP-difficile même si le graphe est orienté, acyclique et admet une structure en niveau (en couches) et a montré que ce problème est polynomialement resoluble dans le cas où le nombre d'arcs avec des longueurs incertaines est fixé ou est limité par le logarithme d'une fonction polynomial du nombre total d'arcs. Indépendamment, Zielinski (2004) [36] a montré que ce problème est NP-difficile et reste NP-difficile même quand un graphe est limité pour un graphes orienté, acyclique, planaire et régulier de degré trois. Ainsi dans la résolution de ce problème, la réduction de l'espace de solution devient une question importante.

Karasan et al (2001) [18] ont étudié le problème de plus court chemin de déviation robuste avec incertitude d'intervalle. Les auteurs ont considéré des graphes acycliques sous l'intervalle d'arcs incertain non négatif et non-dégénéré et proposent une formulation d'un programme en nombre entier mixte avec un prétraitement pour résoudre ce problème. Les auteurs ont proposé une condition suffisante pour qu'un arc ne soit jamais sur un plus court chemin de 1 a m, ils ont présenté une procédure polynômial pour déterminez si un arc est d'un tel type. Le prétraitement consiste à enlever ces arcs qui ne sont jamais dans les chemins les plus courts. Les résultats informatiques ont montré que le prétraitement dans de telles sortes de graphes est efficace.

Montemanni et al (2004) [38] ont fourni un algorithme de branch and bound pour résoudre le problème de chemin de regret maximum sous l'incertitudes de l'intervalle mais ils n'ont pas mis en oeuvre le prétraitement proposé par Karasan et al [18] parce qu'en pratique il peut être utilisé seulement pour des graphes en niveau (en couche) acycliques. Montemanni et Gambardella (2004) [39] a présenté un algorithme exact pour résoudre le problème de plus court chemin de déviation robuste sous l'incertitude d'intervalle. Kasperski et Zielinski a examiné le problème de plus court chemin de regret maximum sur un multi-graphe orienté séries-parallèle avec l'incertitude d'intervalle et a montré que ce problème est NP-difficile. Les auteurs ont présenté un algorithme pseudo-polynômial pour résoudre le problème.

Dans ce chapitre nous étudions le problème de détection des arcs qui ne sont jamais ou

qui sont toujours sur un plus court chemin de 1 à m. Nous considérons ce problème sur une plus grande classe de graphes orientés. Particulièrement nous donnons des conditions suffisantes pour qu'un arc peut être toujours ou jamais sur un plus court chemin de 1 à m (des arcs strictement forts et arcs non faibles, respectivement). Ces conditions nous permettent de donner des algorithmes polynômiaux pour trouver des arcs strictement forts et des arcs non faibles. Nous proposons une procédure de prétraitement consistant d'enlever les arcs non faibles dans le but de réduire le temps de recherche et rechercher dans un graphe réduit les arcs strictement forts.

Problème de plus court chemin sous l'incertitude de l'intervalle	
Critère de regret maximum	
Algorithmes et heuristiques	Théorie et complexité
Montemanni et al (2004) Montemanni et Gambardella (2004) Montemanni et Gambardella (2005a) Karasan et al (2001) Kaspersky et Zielinski (2004) Kaspersky et Zielinski (20046)	Zielinski (2004) Averbakh and Lebedev (2004)
Critère de pire des cas	
Algorithmes et heuristiques	Théorie et complexité
Karasan et al (2001)	Karasan et al (2001)

Problème de plus court chemin : Incertitude sur l'intervalle des données

5.2 Notations et définitions

Nous présentons ici quelques définitions [10] que nous trouvons utiles dans le reste du chapitre. Soit G un graphe fini orienté, on note par $V(G)$ et $A(G)$, l'ensemble des nœuds et des arcs respectivement, on suppose que :$|V(G)| = m$, $|A(G)| = n$. On considère deux nœuds spécial de G le nœud source 1 et le nœud terminal m. On note par (i, j) l'arc du nœud i vers le nœud j. Soit $A^{'}$ un sous ensemble non vide de $A(G)$, le sous graphe de G

avec ensemble des sommets $A(G)\backslash A^{'}$ est tous simplement écrit comme $G - A^{'}$; c'est le sous graphe obtenu à partir de G en supprimant des arcs dans $A^{'}$. Si $A^{'} = \{(i,j)\}$ on écrit $G - \{(i,j)\}$.

On considère le problème déterministe suivant appelée problème de plus court chemin dans les graphes orientés : Soit G un graphe orienté donné avec des arcs non négatifs $l_{i,j}$ associés à chaque arc $(i,j) \in A(G)$ on à besoin de trouver un plus court chemin orienté, noté $(1,m) - chemin$ connectant deux nœuds spécifiques, le sommet source 1 et le sommet puis m dans G, tel que la longueur du chemin est la somme des longueurs de ces arcs. Un algorithme efficace $\bigcirc(\mid V(G) \mid^2)$ pour résoudre ce problème à été donné par Dijkstra, qui permet de trouver un plus court chemin de 1 à m.

Dans le but de construire la version incertaine de ce problème, nous présenterons les concepts et les notations suivants [18]. Soit S l'ensemble des scénarios pour les longueurs d'arcs de G et soit D l'ensemble des données incertaines. On note par l_{ij}^s les longueurs d'arcs (i,j) non négatifs dans le scénario s et on assume que D est le produit cartésien des intervalles, ceci veut dire qure, chaque l_{ij}^s peux prendre une valeur arbitraire dans l'intervalle $[\underline{l}_{ij}, \bar{l}_{ij}]$. On note par $\underline{s}$ le scénario pour lequel tous $(i,j) \in A(G)$, $l_{ij}^{\underline{s}} = \underline{l}_{ij}$ et par $\overline{s}$ le scénario pour lequel tous $(i,j) \in (A(G)$, $l_{ij}^{\overline{s}} = \bar{l}_{ij}$.

Pour la simplicité, nous dénoterons par l_p^s la longueur $\sum_{(i,j)\in A(G)} l_{ij}^s x_{ij}$ du chemin p dans le scénario s. Si nous voulons rechercher un plus court chemin connectant deux sommets 1 et m, on défini :

$$f(p^{*s}, l^s) = l_{p^{*s}}^s = \min_{p \in P_{1m}(G)} l_p^s$$

. Tel que P_{1m} est l'ensemble des chemins de 1 à m. dans G.

5.3 Robustesse absolue

Un chemin robuste est défini pour être celui qui fonctionne(qui se produit) d'une manière satisfaisante quelque soient les données réalisées. Dans cette section, nous utilisons le critère de robustesse absolue pour trouver un chemin robuste. Ce critère choisira un chemin pour lequel la longueur de trajet maximum prise à travers toutes les réalisations possibles est aussi basse que possible. En d'autres termes, nous souhaitons trouver un chemin qui réduit au minimum la longueur du trajet maximum entre l'origine et le nœud terminal.

Kouvelis et Yu [47] ont montrés que le problème de chemin robuste absolu avec un ensemble de scénario est NP-complet même dans les réseaux de décomposition de largeur 2 et avec seulement 2 scénarios. De plus, ils ont prouvés que le problème peut être résolu dans un temps pseudo-polynômial pour les réseaux de décomposition avec l'ensemble lié de scénario et le problème est fortement NP-dur pour un nombre illimité de scénarios.

Dans notre cas, afin de trouver une solution au problème du chemin robuste absolu, il suffit de considérer le scénario unique où la longueur de tous les arcs sur le graphes sont placées à leurs bornes supérieures puisque la longueur de trajet maximum correspond à ce scénario unique. Alors nous pouvons trouver un chemin robuste absolu en trouvant un plus court chemin dans le graphe sous ce scénario. Par conséquent, le problème de chemin robuste absolu peut être résolu en temps polynomial.

Sous le critère de robustesse absolu, les solutions ne sont pas sensibles à la variation des réalisations des données.

L'utilisation de cette approche donne des solutions très conservatrices basées sur l'anticipation que le pire cas pourrait bien se produire. Dans la section suivante, nous considérerons l'approche de déviation robuste qui est plus sensible à l'incertitude sur les données.

5.4 Déviation robuste

Dans cette section, nous souhaitons trouver un chemin tels que la différence maximum entre la longueur de ce chemin et la longueur du plus court chemin sur toutes les réalisations des scénarios est la plus petite. C'est à dire, une solution qui montre la plus petite déviation de pire cas de l'optimalité sur tous les scénarios potentielles. Cette solution permet l'évaluation de l'exécution des décisions contre les meilleurs résultats sous n'importe quel ensemble de données.

Définition 5.1. *La **déviation robuste** d'un chemin p est définie comme étant la différence entre la longueur du chemin p et la longueur du plus court chemin dans le graphe pour une réalisation spécifique des longueurs d'arcs, ie pour un scénario fixée* $d_p = l_p - l_p^*$.

Définition 5.2. *Un chemin p est dit un chemin de déviation robuste s'il admet la déviation robuste minimale sur tous les chemins, ie un chemin de déviation robuste* $p^r \in argmin_{p\in P} max_{s\in S}(l_p^s - l_{p^*(s)}^s)$ *tel que* $p^*(s)$ *est un plus court chemin dans le scénario s.*

Dans le but de trouver dans le graphe un chemin de déviation robuste, on a besoin uniquement de considérer le scénario qui rend la déviation robuste maximum. Un tel scénario est donné par la proposition suivante :

Proposition 5.1. *La déviation robuste d'un chemin p est maximisée sur le scénario pour lequel les longueurs de tous les arcs de p sont à leurs bornes supérieurs et les longueurs de tous les autres arcs sont à leurs bornes inférieurs.*

Preuve. Soit d_p^* la déviation robuste maximale pour un chemin p qui est réalisé sur le scénario s_p. Soit s le scénario pour lequel les longueurs des arcs sur p sont à leurs bornes supérieurs et les longueurs des arcs restants sont à leurs bornes inférieurs. Donc :

$$d_p^* = l_p^{s_p} - l_{p^*(s_p)}^{s_p} = \sum_{(i,j)\in p\setminus p^*(s_p)} l_{ij}^{s_p} - \sum_{(i,j)\in p^*(s_p)\setminus p} l_{ij}^{s_p}$$

$$\leq \sum_{(i,j)\in p\setminus p^*(s_p)} l_{ij}^{s} - \sum_{(i,j)\in p^*(s_p)\setminus p} l_{ij}^{s} = l_p^s - l_{p^*(s_p)}^s \leq l_p^s - l_{p^*(s_p)}$$

Donc s est aussi un scénario qui maximise la déviation robuste du chemin p. Cette proposition implique qu'on a besoin de considérer uniquement un nombre fini de scénarios, à savoir le nombre de chemins dans le graphe. Cependant, le nombre de chemins dans le graphe peut se développer d'une manière exponentielle avec le nombre de nœuds dans le graphe.

Pour évaluer un chemin $p \in P_1(G)$ sous le scénario s on utilise une fonction

$$f : P_1(G) \times D \longrightarrow \Re : (p, l^s) \longrightarrow f(p, l^s)$$

Dans ce travail on assume que cette fonction est la somme des longueurs des arcs de p sous le scénario s.

Donc,

$$f(p, l^s) = \sum_{i \in I} l_i^s p_i$$

Une solution optimal p^{*s} correspondant au scénario s satisfait,

$$f(p^{*s}, l^s) = \min_{p \in P_{1m}(G)} f(p, l^s)$$

$\diamondsuit$

Bertsimas et al (2005) ont étudié "la persistance" d'une variable de décision, c'est à dire la probabilité qu'elle fait partie d'une solution optimale et puis ils ont donné un modèle pour calculer la persistance d'une variable de décision dans des problèmes d'optimisation discrets sous des informations probabilistes sur les coefficients objectifs. Inspiré par cette idée, nous présenterons les définitions suivantes [12] [51] [52].

5.5 Classification des arcs :

Définition 5.3. *Un chemin est dit* faible *si c'est un plus court chemin de* 1 *à* m *pour au moins une réalisation des longueurs d'arcs.*

Définition 5.4. *Un arc est dit* faible *s'il appartient à au moins un chemin faible.*

Définition 5.5. *Un arc est dit* fort *s'il appartient à au moins un plus court chemin de* 1 *à* m *pour tous les scénarios.*

Définition 5.6. *Un arc est dit* non-faible *s'il n'appartient à aucun un plus court chemin de* 1 *à* m *pour tous les scénarios.*

Définition 5.7. *Un arc est dit* strictement fort *s'il appartient à tous les plus court chemin de* 1 *à* m *pour tous les scénarios.*

Proposition 5.2. *La décision si un arc donné est faible est NP-complete.*

Preuve. [18] Nous examinons les problèmes d'arcs sur des graphes en niveaux (en couches). Un graphe en niveau est défini comme celui qui satisfait les propriétés suivantes. L'ensemble de nœud peut être divisé dans des sous-ensembles disjoints $V = \{1\} \cup V_1 \cup \ldots \cup V_m \cup m$, avec $V_i \cap V_j = \emptyset, i \neq j$. Le nœud 1 est le nœud d'origine et le nœud m est le nœud puit. Les arcs existent uniquement à partir de 1 à V_1, V_m à m et de V_k à V_{k+1} pour $k = 1, 2, \ldots, m-1$. Soit $w = \max\{\mid V_k \mid : k = 1, 2, \ldots, m\}$, w est appelé, la largeur du graphe en niveaux. La figure suivante montre un exemple d'un m-graphe en niveaux avec une largeur de 2. La concentration sur des graphes en niveaux n'est pas une analyse restrictive puisque n'importe quel graphe orienté acyclique peut être converti en un graphe en niveau en ajoutant des nœuds et des arcs. ♢

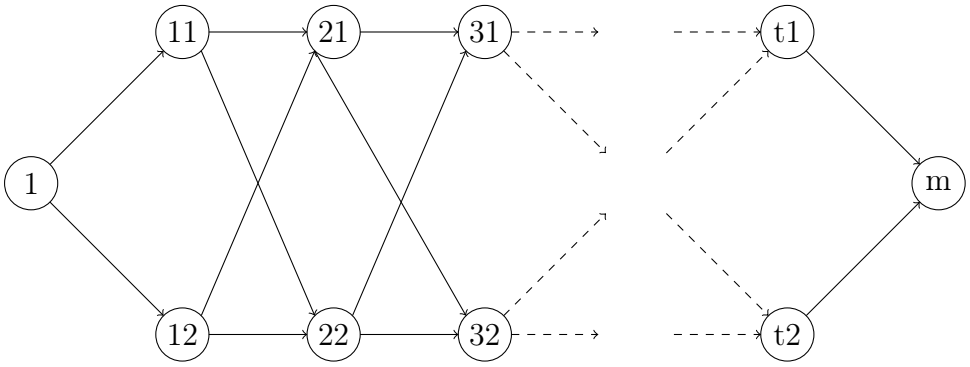

Figure 5.1 – Exemple d'un graphe m-niveaux

Proposition 5.3. *[18] On donne une condition nécessaire et suffisante pour qu'un chemin soit faible. On assume qu'un chemin p est un chemin faible. Donc, pour tous* $p^{'} \in P$ *on a :*

$$\sum_{(i,j)\in p\setminus p^{'}} l_{ij}^{s} + \sum_{(i,j)\in p\cap p^{'}} l_{ij}^{s} \leq \sum_{(i,j)\in p^{'}\setminus p} l_{ij}^{s} + \sum_{(i,j)\in p\cap p^{'}} l_{ij}^{s}$$

$$\Leftrightarrow \sum_{(i,j)\in p\setminus p^{'}} l_{ij}^{s} \leq \sum_{(i,j)\in p^{'}\setminus p} l_{ij}^{s}$$

$$\Rightarrow \sum_{(i,j)\in p\setminus p^{'}} \underline{l}_{ij} \leq \sum_{(i,j)\in p^{'}\setminus p} \bar{l}_{ij}$$

$$\sum_{(i,j)\in p\setminus p^{'}} \underline{l}_{ij} + \sum_{(i,j)\in p\cap p^{'}} \underline{l}_{ij} \leq \sum_{(i,j)\in p^{'}\setminus p} \bar{l}_{ij} + \sum_{(i,j)\in p\cap p^{'}} \underline{l}_{ij}$$

Et donc, p est un plus court chemin lorsque les longueurs de tous les arcs sur p sont à leurs bornes inférieurs et les longueurs de tous les arcs restants sont à leurs bornes supérieurs.

Proposition 5.4. *[18] Un chemin robuste absolue est un chemin faible.*

Soit p un chemin qui n'est pas faible. Soit $p^{'}$ un autre chemin qui est un plus court chemin lorsque les longueurs

Preuve. On se basant sur l'analyse des chemins cité ci dessus, on peux tirer le résultat suivant : un chemin robuste absolue est un chemin faible. Puisque un chemin robuste absolue est un plus court chemin sous le scénario pour lequel toutes les longueurs des arcs sont à leurs bornes supérieurs. ♢

Proposition 5.5. *[18] Un chemin de déviation robuste est un chemin faible.*

Preuve. de tous les arcs sur p sont à leurs bornes inférieurs et les longueurs des arc restants sont à leurs bornes supérieur.Donc, $l_p > l_{p'}$ pour toutes les réalisations des longueurs d'arcs. On considère un scénario s^*pour le chemin $p^{'}$ et pour lequel les longueurs de tous les arcs de $p^{'}$ son à leurs bornes supérieurs et les longueurs de tous les arcs restants sont à leurs bornes inférieurs, ie le scénario qui résulte est comme celui de la déviation robuste maximum.
On à donc

$$max_{s\in S}(l^s_{p'} - l_{p^*(s)}) = l^{s^*}_{p'} - l^{s^*}_{p^*(s^*)} < l^{s^*}_p - l^{s^*}_{p^*(s^*)} \leq max_{s\in S}(l^s_p - l_{p^*(s)})$$

Donc p ne peut pas avoir la plus petite déviation robuste maximum. ♢

Proposition 5.6. *Soit (k,r) l'arc tel que dans le graphe $G-(k,r)$ le nœud m est accessible à partir du nœud 1. Pour chaque $s \in S$ soit $p^*_s \in P^*_{1m}(G,s)$. L'arc (k,r) est un arc strictement fort de G si et seulement si pour tous $q \in P_{1m}(G-(k,r))$ et pour tous $s \in S, l^s_q > l^s_{p^*_s}$.*

Preuve. Soit (k,r) tel que dans le graphe $G-(k,r)$ le nœud m est accessible à partir du nœud 1. Si (k,r) est un arcs strictement fort de G et $q \in P_{1m}(G-(k,r))$, alors q n'est pas un plus court chemin de G de 1 à m pour chaque scénario s, et par conséquent pour tous $s \in S$,$l^s_q > l^s_{p^*_s}$. Maintenant, si pour tous $q \in P_{1m}(G-(k,r))$ et pour tous $s \in S, l^s_q > l^s_{p^*_s}$, donc q n'est jamais un plus court chemin de G à partir de 1 à m pour tous scénario s, donc pour tous $s \in S$ un plus court chemin de G à partir de 1 à m sous le scénario s contient l'arc (k,r), et (k,r) est donc un arc strictement fort. ♢

Remarque

Soit (k,r) tel qu'il existe $p \in P_{1m}(G)$ tel que $(k,r) \in A(p)$ et sur le graphe $G-(k,r)$ le nœud m est non accessible à partir du nœud 1, donc l'arc (k,r) est un arc strictement fort.
On peux observer que dans la dernière caractérisation des arcs strictement forts, le nombre de scénarios extrême peut être exponentiel, donc dans le théorème qui suit on donne une

condition suffisante facile à tester pour qu'un arc soit strictement fort. On va utiliser la proposition précédente dans la preuve.

Théorème 5.1. *Soit $G(V, A)$ un graphe orienté fini, avec un nœud original 1 et un nœud terminal m. On suppose que chaque nœud $j \in V(G)$ est accessible à partir du nœud 1. Soit p le plus court chemin de 1 à m trouver par l'algorithme de Djikstra sous le scénario $\overline{s}$. Soit $(k, r) \in A(p)$, et soit $s^{'}$ le scénario tel que, $l_{ij}^{s^{'}} = \bar{l}_{ij}$ si $(i, j) \in A(p)$, et $l_{ij}^{s^{'}} = \underline{l}_{ij}$ si $(i, j) \notin A(p)$. Si sur le graphe $G - (k, r)$ le nœud m est accessible à partir du nœud 1, soit $q^{'} \in P_{1m}^{*}(G - (k, r), s^{'})$ un plus court chemin de 1 à m sous le scńario $s^{'}$. Si $l_{q^{'}}^{s^{'}} > l_{p}^{s^{'}}$ donc l'arc (k, r) est un arc strictement fort.*

Preuve. Soit p le plus court chemin de 1 à m trouver par l'algorithme de Djikstra sous le scénario $\overline{s}$, soit $(k, r) \in A(p)$. Maintenant on considère le scénario $s^{'}$ tel que $l_{ij}^{s^{'}} = \bar{l}_{ij}$ si $(i, j) \in A(p)$, et $l_{ij}^{s^{'}} = \underline{l}_{ij}$ si $(i, j) \notin A(p)$. Soit $q^{'}$ le plus court chemin de $G - (k, r)$ à partir de 1 à m sous le scńario $s^{'}$. Si $l_{q^{'}}^{s^{'}} > l_{p}^{s^{'}}$ donc pour tout $q \in P_{1m}(G - (k, r))$

$$\sum_{(ij)\in q\setminus p} \underline{l}_{ij} = \sum_{(ij)\in q\setminus p} l_{ij}^{s^{'}} > \sum_{(ij)\in p\setminus q} l_{ij}^{s^{'}} = \sum_{(ij)\in p\setminus q} \bar{l}_{ij}$$

Et donc pour tout $s \in S$ on à :

$$\sum_{(ij)\in q\setminus p} l_{ij}^{s} \geq \sum_{(ij)\in q\setminus p} \underline{l}_{ij} > \sum_{(ij)\in p\setminus q} \bar{l}_{ij} \geq \sum_{(ij)\in p\setminus q} l_{ij}^{s}$$

Ceci implique que pour tout $s \in S$ et pour tout $q \in P_{1m}(G - (k, r))$, on à $l_q^s > l_p^s$ et si $p_s^* \in P_{1m}^*(G, s), l_q^s > l_p^s \geq l_{p_s^*}^s$. à partir de la proposition 1, (k, r) est un arc strictement fort. ♢

Le Théorème précédent nous permet de tirer un algorithme polynomial pour trouver les arcs strictement forts dans le graphe. Un algorithme pour détecter et éliminer les arcs non-faibles est présenté dans ce qui suit.

Dans ce qui précède, nous avons vu que le problème de chemin de déviation robuste semble être plus dure que le problème de plus court chemin absolue. En résolvant le pro-

blème du chemin de déviation robuste, la réduction de l'espace de solution devient une question importante. Dans cette section, nous réalisant une analyse des chemins selon les réalisations des longueurs d'arc. Ce concept a été défini par Yaman [18], et à été employé dans [18][52].

5.6 Algorithme de recherche d'arcs strictement forts

1. Appliquant l'algorithme de Djikstra à G sous le scénario $\overline{s}$ on obtient un plus court chemin p de 1 à m.
2. Choisir un arc $(k, r) \in A(p)$.
3. Construire un scénario $s^{'}$ pour lequel $l_{ij}^{s^{'}} = \overline{l}_{ij}$.si$(i, j) \in A(p)$ et $l_{ij}^{s^{'}} = \underline{l}_{ij}$ si $(i, j) \notin A(p)$
4. Appliquant l'algorithme de Djikstra à $G - (k, r)$ sous le scénario $s^{'}$. S'il existe un plus court chemin $q^{'}$ de 1 à m, Aller à (5). Sinon (k, r) est strictement fort.
5. Calculer $l_{q^{'}}^{s^{'}}$ et $l_{p}^{s^{'}}$.
6. Si$l_{q^{'}}^{s^{'}} > l_{p}^{s^{'}}$ alors l'arc (k, r) est un arc strictement fort.
7. Aller à l'étape (2). Choisir un autre arc dans $A(p)$ et continuer.

5.7 Version améliorée de l'algorithme

Dans ce chapitre nous somme sur le point de faire une classification d'arcs, ce qui nous permet d'améliorer la sixième étape de l'algorithme proposé par Martha [52] en intégrons une instruction dans l'algorithme qui permet de détecter la classe des arcs faible.

5.8 Algorithme de recherche d'arcs strictement forts et d'arcs faibles

1. Appliquant l'algorithme de Djikstra à G sous le scénario $\overline{s}$ on obtient un plus court chemin p de 1 à m.
2. Choisir un arc $(k, r) \in A(p)$.
3. Construire un scénario $s^{'}$ pour lequel $l_{ij}^{s^{'}} = \overline{l}_{ij}$.si$(i, j) \in A(p)$ et $l_{ij}^{s^{'}} = \underline{l}_{ij}$ si $(i, j) \notin A(p)$
4. Appliquant l'algorithme de Djikstra à $G - (k, r)$ sous le scénario $s^{'}$. S'il existe un plus court chemin $q^{'}$ de 1 à m, Aller à (5). Sinon (k, r) est strictement fort, aller à (2).
5. Calculer $l_{q^{'}}^{s^{'}}$ et $l_{p}^{s^{'}}$.

6. Si$l_{q'}^{s'} > l_p^{s'}$ alors l'arc (k, r) est un arc strictement fort **sinon si** $l_{q'}^{s'} \leq l_p^{s'}$ **alors** (k, r) **est un arc faible car il appartient à un plus court chemin trouvé dans un scénario précédent.**
7. Aller à l'étape (2). Choisir un autre arc dans $A(p)$ et continuer.

5.8.1 Justification de l'algorithme

l'idée est de classifier les arcs en deux classes :

– Arcs strictement forts.

– Arcs faibles.

Les arcs strictement forts : Ce sont des arcs qui sont toujours sur le plus court chemin pour toutes les variations des longueurs d'arcs ie pour tous les scénarios.

Ces arcs sont détectés en effectuant un test tel que l'étape (6) de l'algorithme l'indique : Si$l_{q'}^{s'} > l_p^{s'}$, (le scénario s' tel que $l_{ij}^{s'} = \overline{l}_{ij}$.si$(i, j) \in A(p)$ et $l_{ij}^{s'} = \underline{l}_{ij}$ si $(i, j) \notin A(p)$ est celui trouvé par le critère de regret maximum de plus un plus court chemin se trouve par l'application de l'algorithme de Dijkstra) alors l'arc (k, r) est un arc strictement fort.

Dans le cas ou l'arc testé n'assure pas la condition, ie $l_{q'}^{s'} \leq l_p^{s'}$ ceci veut dire que l'arc n'est pas un arc strictement fort. Par la suite, si $l_{q'}^{s'} \leq l_p^{s'}$ nous classifions cet arc comme étant un arc faible, car c'est un arc trouvé dans l'étape (1) de l'algorithme, appartenant à un plus court chemin trouvé sur un scénario fixé $\overline{s}$ ie c'est un arc qui appartient à aumoins un plus court chemin pour aumoin un scénario (aumoin une réalisation des longueurs d'arcs).

Nous passons par la suite dans les deux cas à l'étape (2) et nous choisissons un deuxième arc sur lequel nous effectuons les mêmes testes.

5.8.2 Complexité de l'algorithme

1. Application de Dijkstra à G sous $\overline{s}$, l'algorithme de Dijkstra pour un graphe orienté à m sommets et n arcs admet pour complexité $\bigcirc(n + m \times \ln(n))$ (Voir page (23))
2. Choix d'un arc $(k, r) \in A(p)$, une seul opération de choix et donc la complexité de cette instruction est de $\bigcirc(1)$.
3. Construction du scénario s' tel que :

- $l_{ij}^{s'} = \bar{l}_{ij}$ si $(i,j) \in A(p)$, cette instruction admet pour complexité $\mid p \mid$ opérations, ie le nombres d'arcs qui constituent le chemin p et donc sa complexité est de $\bigcirc(\mid p \mid)$
- $l_{ij}^{s'} = \underline{l}_{ij}$ si $(i,j) \notin A(p)$, cette instruction admet pour complexité $\mid A \mid - \mid p \mid$ opérations, ie le nombres d'arcs qui constituent le chemin p et donc sa complexité est de $\bigcirc(\mid A \mid - \mid p \mid)$

4. $\exists q'$ un plus court chemin de 1 à m, cette opération se termine par une seul affectation malgré le résultat de si et donc cette instruction admet pour complexité $\bigcirc(1)$.

Comme l'algorithme s'execute n fois et donc la complexité de l'algorithme est de n fois le nombre d'iterations de cet algorithme et donc l'algorithme est de complexité

$$\bigcirc(n^2) + \bigcirc(n(n + m \times \ln(n))) = \bigcirc(nmln(n))$$

5.9 Proposition d'un Algorithme de recherche des arcs non faibles :

1. Fixer les longueurs d'arcs à leurs bornes inférieurs ie le scénario $\underline{s}$.
2. Choisir un arc $(k,r) \in A(G)$.
3. On considère le chemin $p \in P_{1k}(G)$.
4. On considère le chemin $C \in P_{rm}(G)$.
5. Construire le scénario s' pour lequel $l_{kr}^{s'} = \underline{l}_{kr}, l_{ij}^{s'} = \underline{l}_{ij}$ si $(i,j) \in A(p)$ $l_{ij}^{s'} = \underline{l}_{ij}$ si $(i,j) \in A(C)$et $l_{ij}^{s'} = \bar{l}_{ij}$ si $(i,j) \notin A(p)$, $(i,j) \notin A(C)$ et $(i,j) \neq (k,r)$.
6. Appliquant l'algorithme de Djikstra à G sous le scénario s' on obtient un plus court chemin q' de 1 à m.
7. Calculer $l_{q'}^{s'}$, $l_p^{s'}$ et $l_C^{s'}$.
8. Si $l_{q'}^{s'} < l_p^{s'} + \underline{l}_{kr} + l_C^{s'}$ alors l'arc (k,r) est un arc non faible, éliminer (k,r) et travaillez sur le graphe réduit $G - (k,r)$, sinon (k,r) est un arc faible.

5.9.1 Justification de l'algorithme

Dans cet algorithme l'idée cette fois est de classifié les arcs en deux classes :

- Arcs non faibles.
- Arcs faibles.

Les arcs non faibles : Ce sont des arcs qui n'appartiendrons jamais au plus court chemin pour toutes les variations des longueurs d'arcs ie pour tous les scénarios.

Ces arcs sont détectés en effectuons un test tel que l'étape (8) de l'algorithme l'indique : Si $l_{q'}^{s'} < l_p^{s'} + \underline{l}_{kr} + l_C^{s'}$, (le scénario s' tel que $l_{kr}^{s'} = \underline{l}_{kr}, l_{ij}^{s'} = \underline{l}_{ij}$ si $(i,j) \in A(p)$ $l_{ij}^{s'} = \underline{l}_{ij}$ si $(i,j) \in A(C)$et $l_{ij}^{s'} = \bar{l}_{ij}$ si $(i,j) \notin A(p)$, $(i,j) \notin A(C)$ et $(i,j) \neq (k,r)$ est celui trouvé par le critère de regret maximum de plus, un plus court chemin se trouve par l'application de l'algorithme de Dijkstra) alors l'arc (k,r) est un arc non faible.

Dans le cas ou l'arc testé n'assure pas la condition, ie $l_{q'}^{s'} < l_p^{s'} + \underline{l}_{kr} + l_C^{s'}$ ceci veut dire que l'arc n'est pas un arc non-faible. Par la suite, nous classifions cet arc comme étant un arc faible.

Nous passons par la suite dans les deux cas à l'étape (2) et nous choisissons un deuxième arc sur lequel nous effectuons les mêmes testes.

5.9.2 Complexité de l'algorithme

1. Fixer l_{ij} sur $\underline{s}$, ie $l_{ij} = \underline{l}_{ij}, \forall (i,j) \in A(G)$. Cette étape depend du nombre d'arcs et donc elle admet pour complexité : $\mid A(G) \mid = n$ opération à effectuer donc la complexité de cette instruction est de : $\bigcirc(n)$.
2. Choix de $(k,r) \in A(G)$, cette instruction admet pour complexité $\bigcirc(1)$
3. Construction du chemin $p \in P_{1k}(G)$, cette instruction admet pour complexité $\mid p \mid$ opérations, ie le nombres d'arcs qui constituent le chemin p et donc sa complexité est de $\bigcirc(\mid p \mid)$
4. Construction du chemin $C \in P_{rm}(G)$, cette instruction admet pour complexité $\mid C \mid$ opérations, ie le nombres d'arcs qui constituent le chemin C et donc sa complexité est de $\bigcirc(\mid C \mid)$
5. Construction du scénario s' tel que :

- $l_{kr}^{s'} = \underline{l}_{kr}$, cette instruction est une affectation, elle admet pour complexité $\bigcirc(1)$
- $l_{ij}^{s'} = \underline{l}_{ij}$ si $(i,j) \in A(p)$, cette instruction admet pour complexité $\mid p \mid$ opérations, ie le nombres d'arcs qui constituent le chemin p et donc sa complexité est de $\bigcirc(\mid p \mid)$
- $l_{ij}^{s'} = \underline{l}_{ij}$ si $(i,j) \in A(C)$, cette instruction admet pour complexité $\mid C \mid$ opérations, ie le nombres d'arcs qui constituent le chemin C et donc sa complexité est de $\bigcirc(\mid C \mid)$
- $l_{ij}^{s'} = \bar{l}_{ij}$ si $(i,j) \notin A(p)$, cette instruction admet pour complexité $\mid A \mid - \mid p \mid$ opérations, et donc sa complexité est de $\bigcirc(n- \mid p \mid)$
- $l_{ij}^{s'} = \bar{l}_{ij}$ si $(i,j) \notin A(C)$, cette instruction admet pour complexité $\mid A \mid - \mid C \mid$ opérations, et donc sa complexité est de $\bigcirc(n- \mid C \mid)$
- $l_{ij}^{s'} = \bar{l}_{ij}$ si $(i,j) \neq (k,r)$, cette instruction admet pour complexité $\mid A \mid -1$ opérations, et donc sa complexité est de $\bigcirc(n-1)$

6. Application de Dijktsra à G sous s' pour l'obtention de q' plus court chemin de 1 à m, l'algorithme de Dijkstra pour un graphe orienté à m sommets et n arcs admet pour complexité $\bigcirc(n + m \times \ln(n))$ (Voir page (23))
7. Calculer $l_{q'}s', l_p^{s'}, l_c^{s'}$, ce sont trois opérations et donc la complexité de cette instruction est de $\bigcirc(3)$
8. Si $l_{q'}^{s'} < l_p^{s'} + \underline{l}_{kr} + l_c^{s'}$, cette instruction est composée de deux opérations une somme et une comparaison et donc elle admet une complexité de $\bigcirc(2)$
9. Si $l_{q'}^{s'} < l_p^{s'} + \underline{l}_{kr} + l_c^{s'}$ aller à 1 sinon aller à 1 et donc une seul opération et l'instruction admet pour complexité $\bigcirc(1)$

Comme l'algorithme s'execute n fois et donc la complexité de l'algorithme est de n fois le nombre d'iterations de cet algorithme et donc l'algorithme est de complexité

$$\bigcirc(n^2) + \bigcirc(n(n + m \times \ln(n))) = \bigcirc(nmln(n))$$

Exemple1 : on applique l'algorithme à l'exemple suivant donné dans la figure 5.2

Première étape :

1. On pose les longueurs de tous les arcs à leurs bornes inférieurs voir 5.3.

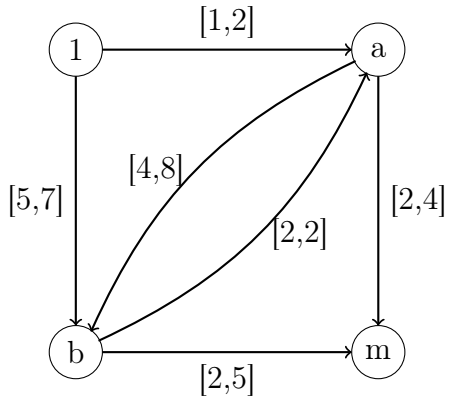

Figure 5.2 – Application de l'algorithme sur le graphe

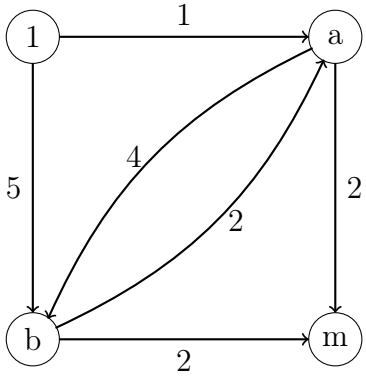

Figure 5.3 – Première étape

2. Choix de $(k, r) = (a, b)$.
3. Trouver le plus court chemin p_{1k} (de 1 à a) $p_{1k} = (1, a)$, $l_{p_{1a}} = 1$
4. Trouver le plus court chemin C_{rm} (de b à m) $C_{rm} = (b, m)$, $l_{p_{bm}} = 2$
5. Construire le scénario $s^{'}$ tel que $l_{ab} = 4, l_{1a} = 1, l_{bm} = 2$
6. Chercher $q^{'}$ le plus court chemin de 1 à m.
   $q^{'} = \{(1, a), (a, m)\}$ $l_{q'} = 5, l_p^{s^{'}} = 1, l_C^{s^{'}} = 2$ $l_p^{s^{'}} + l_C^{s^{'}} + \underline{l}_{kr} = 1+2+4 = 7$ $l_{q'} < l_p^{s^{'}} + l_C^{s^{'}} + \underline{l}_{kr}$ et donc (a, b) est un arc non faible, et donc (a, b) est à éliminer et je travaille sur le graphe réduit $G \setminus (b, a)$.

   Deuxième étape :
7. On pose les longueurs de tous les arcs à leurs bornes inférieurs.

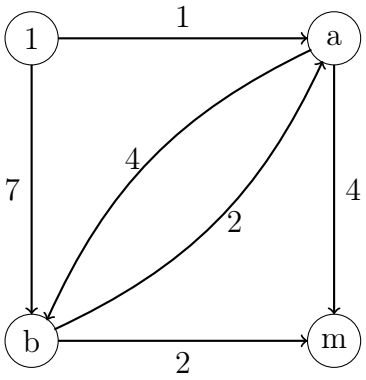

Figure 5.4 – Application de l'algorithme

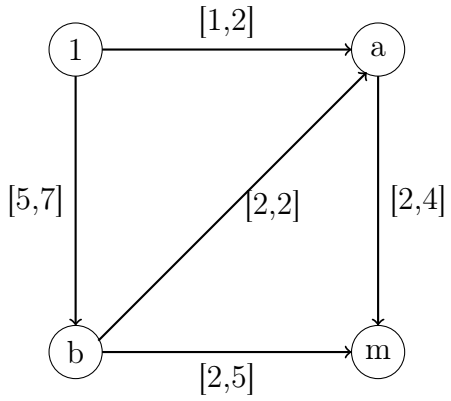

Figure 5.5 – Deuxième étape

8. Choix de $(k, r) = (b, a)$.
9. Trouver le plus court chemin p_{1k} (de 1 à b) $p_{1b} = (1, b)$, $l_{p_{1b}} = 5$
10. Trouver le plus court chemin C_{am} (de a à m) $C_{rm} = (a, m)$ $l_{C_{am}} = 2$
11. Construire le scénario $s^{'}$ tel que $l_{ba} = 2, l_{1b} = 5, l_{am} = 2$
12. Chercher $q^{'}$ le plus court chemin de 1 à m.
    $q^{'} = \{(1, a), (a, m)\}$ $l_{q'} = 4, l_p^{s^{'}} = 5, l_C^{s^{'}} = 2$ $l_p^{s^{'}} + l_C^{s^{'}} + \underline{l}_{kr} = 5+2+2 = 9$ $l_{q'} < l_p^{s^{'}} + l_C^{s^{'}} + \underline{l}_{kr}$ et donc (b, a) est un arc non faible, et donc (b, a) est à éliminer et je travaille sur le graphe réduit $G \setminus (a, b), (b, a)$.

    Troisième étape :
13. On pose les longueurs de tous les arcs à leurs bornes inférieurs.
14. Choix de $(k, r) = (a, m)$.

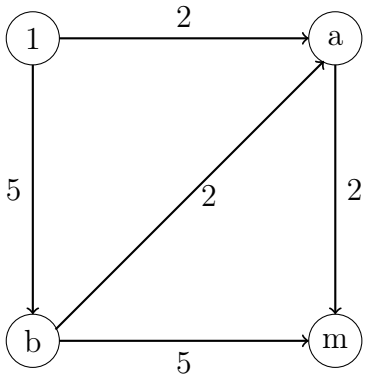

Figure 5.6 – Application de l'algorithme

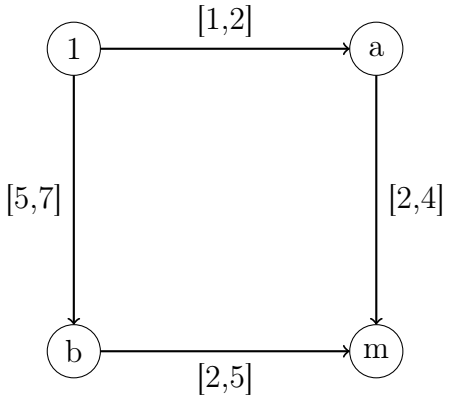

Figure 5.7 – Troisième étape

15. Trouver le plus court chemin p_{1k} (de 1 à b) $p_{1b} = (1, b)$, $l_{p_{1b}} = 5$
16. Trouver le plus court chemin C_{am} (de a à m) n'existe pas et donc (a, m) est un arc faible.

Quatrième étape :

17. On pose les longueurs de tous les arcs à leurs bornes inférieurs.
18. Choix de $(k, r) = (1, a)$.
19. Trouver le plus court chemin p_{1k} (de 1 à a) n'existe pas et donc $(1, a)$ est un arc faible.

Cinquième étape :

20. On pose les longueurs de tous les arcs à leurs bornes inférieurs.
21. Choix de $(k, r) = (1, b)$.

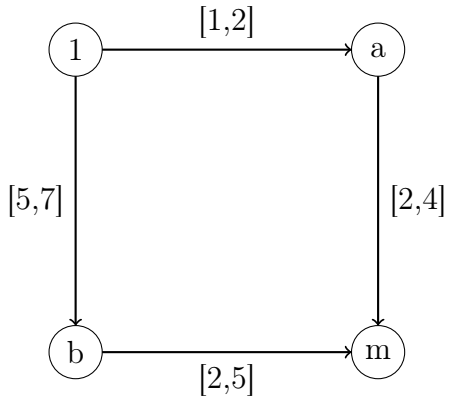

Figure 5.8 – Quatrième étape

22. Trouver le plus court chemin p_{1k} (de 1 à a) n'existe pas et donc $(1, b)$ est un arc faible.

    sixième étape :

23. On pose les longueurs de tous les arcs à leurs bornes inférieurs.

24. Choix de $(k, r) = (b, m)$.

25. Trouver le plus court chemin p_{1k} (de 1 à b) $p_{1b} = (1, b)$, $l_{p_{1b}} = 1$

26. Trouver le plus court chemin C_{mm} (de m à m) n'existe pas et donc (a, m) est un arc faible.

L'application de l'algorithme de recherche des arcs strictement forts sur le graphe réduit par l'application de l'algorithme de recherche et élimination des arcs non-faibles

Exemple Pour clarifier la procédure précédente, on l'applique sur le graphe donné dans la figure 5.9

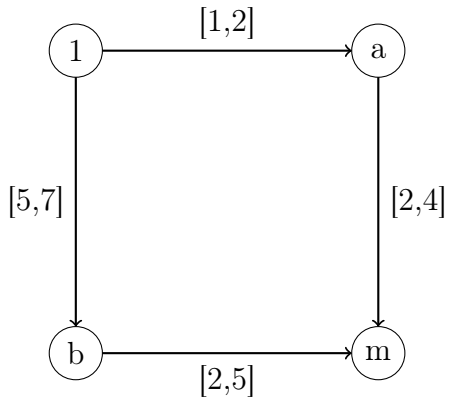

Figure 5.9 – Le sous graphe obtenu après l'application du premier algorithme

Première itération :

1. On pose la longueur de tous les arcs à leurs bornes supérieure, on trouve le plus court chemin p à partir du nœud 1 au nœud m, on présente ces chemins avec des traits foncés.

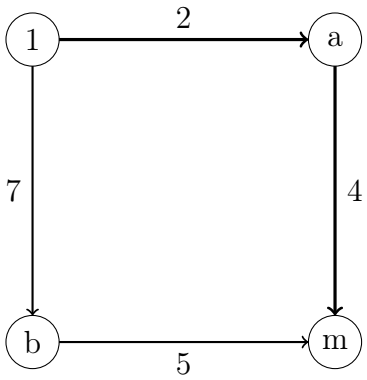

Figure 5.10 – première étape

2. $A(p) = \{(1, a), (a, m)\}$, je choisie $(k, r) = (1, a)$

3. Construction du scénario $s^{'}$, tel que, $l_{(1,a)} = \underline{l}_{(1,a)} = 2$, $l_{(a,m)} = \underline{l}_{(a,m)} = 4$ et $l_{(1,b)} = \overline{l}_{(1,b)} = 5$, $l_{(b,m)} = \overline{l}_{(b,m)} = 2$
4. Application de Dijkstra au graphe réduit $G \setminus (k, r)$ sous $s^{'}$ pour trouver le plus court chemin $q^{'}$ s'il existe.

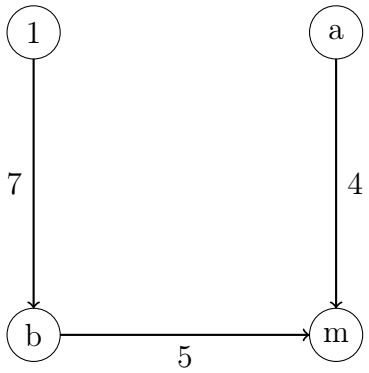

Figure 5.11 – Application de Dijkstra au graphe réduit $G \setminus (k, r)$

5. $q^{'} = \{(1, b), (b, m)\}$, $l^{s^{'}}_{q^{'}} = 7 + 5 = 12$, $l^{s^{'}}_{p} = 2 + 4 = 6$
6. $l^{s^{'}}_{q^{'}} > l^{s^{'}}_{p}$, et donc l'arc $(1, a)$ est un arc strictement fort
7. Revenir à la deuxième étape, et continuer en choisissant un autre arc.

Deuxième itération :

1. Choix de $(k, r) = (a, m)$
2. Construction du scénario $s^{'}$ pour lequel $l_{(1,a)} = \overline{l}_{(1,a)} = 2$, $l_{(1,b)=\underline{l}_{(1,b)}} = 5$, $l_{(b,m)} = \underline{l}_{(b,m)} = 2$
3. Application de Dijkstra au graphe réduit $G \setminus (k, r)$ sous $s^{'}$ pour trouver le plus court chemin $q^{'}$ s'il existe.
4. $q^{'} = \{(1, b), (b, m)\}$, $l^{s^{'}}_{q^{'}} = 5 + 2 = 7$, $l^{s^{'}}_{p} = 2 + 4 = 6$
5. $l^{s^{'}}_{q^{'}} > l^{s^{'}}_{p}$, et donc l'arc (a, m) est un arc strictement fort.

   Et donc les arcs $(1, a), (a, m)$ sont des arcs strictement forts et appartiennent forcement au plus court chemin de 1 à m.

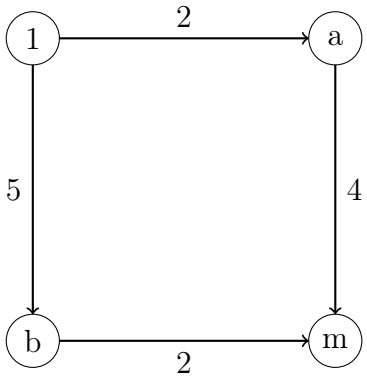

Figure 5.12 – Deuxième étape

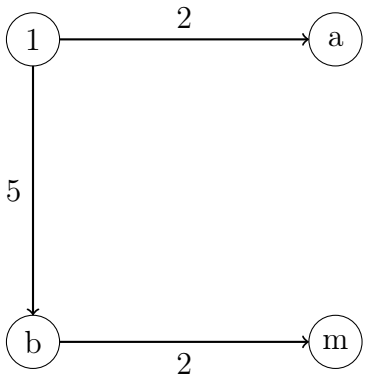

Figure 5.13 – Application de Dijkstra au graphe réduit $G \setminus (k, r)$

5.10 Conclusion

Dans ce chapitre, nous avons donné une condition nécessaire pour qu'un arc soit toujours sur un plus court chemin (arcs strictement forts) et pour qu'un arc ne soit jamais sur un plus court chemin (arcs non faibles), par la suite nous avons critiquer puis améliorer un algorithme qui permet de rechercher les arcs strictement forts dans le graphe, et nous avons aussi proposer un algorithme qui permet de rechercher puis éliminer les arcs non faibles dans le graphe.

Conclusion et perspectives

Conformément aux définitions classiques de la robustesse, la version robuste du problème de plus court chemin devient NP-Difficile. Par conséquent, l'étude de la réduction de l'espace des solutions devient une question importante.

Nous avons examiné le problème de plus court chemin, sur un graphe fini orienté tel que les longueurs d'arcs sont non-négatifs et appartiennent à des intervalles. Nous avons présentés des conditions suffisantes pour qu'un arc soit toujours sur un plus court chemin *(arcs strictement forts)* où ne soit jamais sur un plus court chemin *(arcs non-faibles)*. Basé sur ces résultats nous avons présenté un algorithme polynômial de recherche d'arc strictement forts et un autre algorithme polynômial pour la recherche et l'élimination des arcs non-faibles.

Des considérations futurs dans ce secteur de recherche impliquent les aspects question suivantes :

- Verifier la performance de nos algorithmes de recherche d'arcs strictement forts et d'arcs non-faibles.
- L'étude des conditions pour qu'une variable soit strictement forte ou non faible pour d'autres problèmes d'optimisation combinatoire.
- L'étude des propriétés d'une classe d'autres problèmes pour lesquels la décision doit résister aux changements répétitifs de conditions, définir de nouveaux critères de robustesse et construire des algorithmes pour résoudre de tels problèmes.

Bibliographie

[1] Adam Kasperskia, Pawel Zielinski. *The robust shortest path problem in series–parallel with interval data.*Operations Research Letters. 34 ;(2006)69 –76.

[2] André Rossi. *TOrdonnencement en milieu incertain ; mise en oeuvre d'une démarche robuste.*Thèse de doctorat de L'INPG Institut national polytechnique de greunoble.

[3] A. Kasperski *Minmax Regret Shortest Path*, Discrete Optimization with Interval Data, 228 ;81-112,2008.

[4] Aissi Hassene , Cristina Bazgan, Daniel Vanderpooten. *Min–max and min–max regret versions of combinatorial optimization problems : A survey.*European Journal of Operational Research 197 ;427-438,2009.

[5] Aissi Hassene , Cristina Bazgan, Daniel Vanderpooten. *Complexity of the Min–max and min–max regret versions of min cut problems.*Operational Research letters 33 ;634-640,2005.

[6] Aissi Hassene , Cristina Bazgan, Daniel Vanderpooten. *Complexity of the Min–max (regret) assignement problems : A survey.*Operational Research letters 33 ;634-640,2005.

[7] Antonio Sedeno-Noda, Carlos Gonzalez-Martin *On the K shortest path trees problem*, European Journal of Operational Research 202 ;628-635,2010.

[8] Bernard Roy *Robustness in operational research and decision aiding : A multi-faceted issue*, European Journal of Operational Research 200 ;629-938,2010.

[9] Bernard Roy *Robustesse de quoi et vis à vis de quoi mais aussi robustesse pourquoi en aide à la décision ?*, 2003.

[10] J.A. Bondy and U.S.R. Murty. *Graph Theory with applications. American* Elsevier, New York, 1976.

[11] Bruno Escoffier, Jerome Monnot et Olivier Spanjaard*Some Tractable Instances of Interval Data Minmax Regret Problems : Bounded Distance from Triviality*,Operations Research Letters 36 ;424-429,2008.

[12] Dimitris Bertsimas, Dessislava Pachamanova, Melvyn Sim *Robust linear optimization under general norms*, Operations Research Letters 32 ;510-516,2004.

[13] Dimitris Bertsimas · Melvyn Sim *Robust discrete optimization and network flows*, Math.Program 98 ;49-71,2003.

[14] Dimitris Bertsimas,Melvyn Sim. *The Price of Robustness.*Operations Research Letters. 52 ;(2004)35-53.

[15] Fabio Hernandes, Maria Teresa Lamata, Jose Luis Verdegay et Akebo Yamakami*The shortest path problem on networks with fuzzy parameters*,OFuzzy Sets and Systems 158 ;1561 – 1570,2007.

[16] Gang Yu et Jian Yang. *On the robust shortest path problem.* Comput. Oer. Res. 6 ;457-468,1998.

[17] Hande Yaman, Oya Ekin Karasan, Mustafa C.Pinar. *The robust spanning tree problem with interval data.*Operations Research Letters 29 ;(2001)31-40.

[18] Hande Yaman, Oya Ekin Karasan, Mustafa C.Pinar. *The robust Shortest path problem with interval data.*Comput. Oer. Res. (2001)

[19] H. Yaman, OE. Karasan, MC. Pinar *Restricted robust uniform matroid maximization under interval uncertainty*, Math.Program 110 ;(2007)431-441.

[20] Igor Averbakh, Vasilij Lebdev. *Interval data minmax reget network with uncertainty.*Operations Research Letters 138 ;(2004)289-301.

[21] Igor Averbakh. *Interval data minmax reget network optimization problems.*Discrete Applied Mathematics 138 ;(2004)289-301.

[22] Igor Averbakh. *Minmax regret solutions for minimax optimization problems.* Discrete Applied Mathematics 27 ;(2000)57-65.

[23] Igor Averbakh, *On the complexity of a class of combinatorial optimization problems with uncertainty.*Math.Program 90 ;263-273,2001.

[24] Igor Averbakh, *Minmax regret solutions for minimax optimization problems with uncertainty.*Operations Research Letters 27 ;57-65,2000.

[25] Igor Averbakh, *The minmax regret permutation flow-shop problem with two jobs.*European Journal of Operational Research 169 ;761-766,2006.

[26] Igor Averbakh, Vasilij Lebedev *On the complexity of minmax regret linear programming*, European Journal of Operational Research 160 ;227-231,2005.

[27] Igor Averbakh. *Computing and minimizing the relative regret in combinatorial optimization with interval data.*Discrete Optimization 2 ;(2005)273-287.

[28] Igor Averbakh. *Minmax regret linear resource allocation problems.* Operations Research Letters 32 ;(2004)174-180.

[29] IRavindra K. Ahuja, Thomas L. Magnanti, et James B. Orlin *Network Flows*, Elsevier Science Publishers 1989.

[30] John M. Mulvey and Robert J. Vanderbet *Robust Optimization of Large-scale Systems*, European Journal of Operations Research 43 ;264-281,1995.

[31] JA. Bondy et USR Murty. *Graph theory with applications.* American Elsevier, New York, and Macmillian, London.

[32] Luis C.Dias, Joao N.Climaco*Shortest path problems with partial information : Models and in a planar network with duration intervals*, European Journal of Operational Research 121 ;16-31,2000.

[33] M.Salazar-Neumann. *The robust minimum spanning tree problem : compact and convex uncertainty.*Operations Research Letters 35 ;17-22,2007.

[34] Melvyn Sim *Robust Optimization*, 2004.

[35] Ph. Vincke *Robust and neutral methods for aggregating preferences into an outranking relation*, European Journal of Operational Research 112 ;405-412,1999.

[36] Pawel Zielinski. *The computational complexity of the relative robust shortest path problem with interval data.* European Journal oh Operational Research 158 ;(2004)570-576.

[37] Pierre Lopez. *Cours de graphes.* http ://www.laas.fr/ lopez/cours/GRAPHES/graphes.html (2005).

[38] R.Montemanni, L.M. Gambardella, A.V.Donati. *A branch and bound algorithm for the robust shortest path problem with interval data.*Operations Research Letters 32 ;(2004)225-232.

[39] R.Montemanni, L.M. Gambardella, A.V.Donati. *An exact algorithm for the robust shortest path problem with interval data.*Computers and operations research 31 ;(2004)1664-1680.

[40] R.Montemanni, L.M. Gambardella. *Robust shortest path problems with uncertain costs.*Technical Report IDSIA-03 ;2008,.

[41] Roberto Montemanni and Luca Maria Gambardella. *The robust shortest path problem with interval data via Benders decomposition.*Computers and operations research 3 ;315-328,2005.

[42] R. Kalai, C. Lamboray. *Lαrobustesse lexicographique : une relaxation de la βrobustesse.* ANNALES DU LAMSADE N°7 Mai 2007.

[43] Tero Harju. *Graph theory.* 2007.

[44] Tomas Feder, Rajeev Motwani, Liadan Callaghan ,Chris Olston, Rina Panigrahy. *Computing shortest paths with uncertainty.*Journal of Algorithms 62 ;(2007)1-18.

[45] Thomas. l. Magnanti, Lorence A. Wolsey *Optimal Trees*, Elsevier Science Vol 7, 1995.

[46] V. Gabrel, C. Murat. *Robust shortest path problems.* ANNALES DU LAMSADE N°7 Mai 2007.

[47] P.Kouvelis, G.Yu. *Robust Discrete Optimization and its Application.* Kluwer Academic Publishers, Boston 1997.

[48] P.Penry and O.Spanjaard. *An axiomatic approach to robustness in search problems with multiple scenarios.* European Journal oh Operational Research. 62 ;(2003)469-476.

[49] Ph.Vincke . *Robust and neutral methods for aggregating preferences into an outranking relation.* European Journal oh Operational Research. 112 ;(1999)405-412.

[50] R.Hites ; Y.De Smet, N.Risse, M.Salazar-Neumann and P.Vincke. *About the applicability of MCDA to some robustness problems.* European Journal oh Operational Research. 174 ;322-332,2006.

[51] Martha Salazar Neumann.*Advances in robust combinatorial optimization and linear programming* Thèse de doctorat ,2009-2010.

[52] Martha Salazar Neumann.*The robust shortest path and the single-source shortest path problems : Interval data* ANNALES DU LAMSADE N°7 Mai 2007.

$\mathcal{R}$*ésumé*

Dans ce travail, on s'intéresse au problème de recherche d'un plus court chemin robuste. Un plus court chemin robuste est un chemin qui résiste à l'optimalité malgré les variations des longueurs d'arcs.

Nous avons alors cité quelques résultats de la littérature concernant ce problème.

Dans La première partie du mémoire, nous étudions la complexité du problème de plus court chemin robuste que nous montrons que c'est un problème NP-difficile.

La deuxième partie de notre travail consiste à proposer un algorithme de recherche d'arcs strictement forts (arcs appartenant au plus court chemin robuste) et un autre algorithme de recherche et d'élimination d'arcs non faibles (arcs qui n'appartiennent jamais au plus court chemin robuste).

Mots clés : Robustesse, plus court chemin, graphe, classification des arcs, complexité algorithmique.